

# **PEMROGRAMAN WEB LANJUT**

## **PERTEMUAN 10**

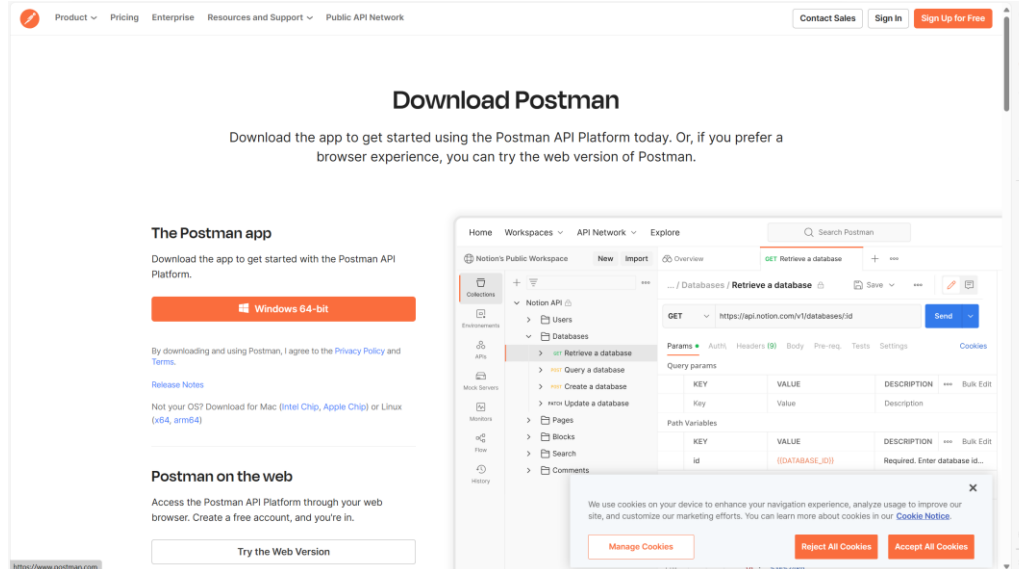


**MUHAMMAD IQBAL MAKMUR AL-MUNIRI**  
**TI-2C / 15 / 2241720099**  
**TEKNIK INFORMATIKA**  
**TEKNOLOGI INFORMASI**

## PRAKTIKUM 1 – Membuat Restful API Register

1. Sebelum memulai membuat REST API, terlebih dahulu download aplikasi Postman di. Aplikasi ini akan digunakan untuk mengerjakan semua tahap praktikum pada jobsheet ini.

Jawab:



2. Lakukan instalasi JWT dengan mengetikkan perintah berikut

Jawab:

```
MINGW64:/d/Polinema/Semester 4/Pemrograman Web Lanjut/pwl_pos
GalaAldebara@LAPTOP-OD4VKIRU MINGW64 /d/Polinema/Semester 4/Pemrograman Web Lanjut/pwl_pos (main)
$ composer require tymon/jwt-auth:2.1.1
./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 (2.3.0)
- Locking lcobucci/jwt (4.0.4)
- Locking stella-maris/clock (0.1.7)
- Locking tymon/jwt-auth (2.1.1)
Writing lock file
Installing dependencies from lock file (including require-dev)
Package operations: 4 installs, 0 updates, 0 removals
- Downloading stella-maris/clock (0.1.7)
- Downloading lcobucci/clock (2.3.0)
- Downloading lcobucci/jwt (4.0.4)
- Downloading tymon/jwt-auth (2.1.1)
- Installing stella-maris/clock (0.1.7): Extracting archive
- Installing lcobucci/clock (2.3.0): Extracting archive
- Installing lcobucci/jwt (4.0.4): Extracting archive
- Installing tymon/jwt-auth (2.1.1): Extracting archive
```

- Setelah berhasil menginstall JWT, lanjutkan dengan publish konfigurasi file dengan perintah berikut

Jawab:

```
INFO Publishing assets.

Copying file [D:\Polinema\Semester 4\Pemrograman Web Lanjut\pwl_pos\vendor\tymon\jwt-auth\config\config.php] to [D:\Polinema\Semester 4\Pemrograman Web Lanjut\pwl_pos\config\jwt.php] DONE

GalaAldebara@LAPTOP-OD4VKIRU MINGW64 /d/Polinema/Semester 4/Pemrograman Web Lanjut/pwl_pos (main)
$
```

- 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.

Jawab:

```
config > jwt.php
1  <?php
2
3  /*
4   * This file is part of jwt-auth.
5   *
6   * (c) Sean Tymon <tymon148@gmail.com>
7   *
8   * For the full copyright and license information, please view the LICENSE
9   * file that was distributed with this source code.
10  */
11
12  return [
13
14      /*
15       |-----
16       | JWT Authentication Secret
17       |-----
18       |
19       | Don't forget to set this in your .env file, as it will be used to sign
20       | your tokens. A helper command is provided for this:
21       | `php artisan jwt:secret`
22       |
23       | Note: This will be used for Symmetric algorithms only (HMAC),
24       | since RSA and ECDSA use a private/public key combo (See below).
25       |
26       */
27
28      'secret' => env('JWT_SECRET'),
```

- Setelah itu jalankan perintah untuk membuat secret key JWT.

Jawab:

```
GalaAldebara@LAPTOP-OD4VKIRU MINGW64 /d/Polinema/Semester 4/Pemrograman Web Lanjut/pwl_pos (main)
$ php artisan jwt:secret
jwt-auth secret [sN2xhsIVl2g2HLNq0np44cqc1JesadRLhmhSiF81uCc96vsABObqrEWRcobVMvF] set successfully.

GalaAldebara@LAPTOP-OD4VKIRU MINGW64 /d/Polinema/Semester 4/Pemrograman Web Lanjut/pwl_pos (main)
$ |
```

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

Jawab:

```
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

Jawab:

```
UserModel.php M
app > Models > UserModel.php > ...
...
1  <?php
2
3  namespace App\Models;
4
5  use Tymon\JWTAuth\Contracts\JWTSubject;
6  use Illuminate\Foundation\Auth\User as Authenticatable;
7  ...
8  class UserModel extends Authenticatable implements JWTSubject
9  {
10     public function getJWTIdentifier()
11     {
12         return $this->getKey();
13     }
14
15     public function getJWTCustomClaims()
16     {
17         return [];
18     }
19
20     protected $table = 'm_user';
21     protected $primaryKey = 'user_id';
```

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

Jawab:

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS GITLENS
PS D:\Polinema\Semester 4\Pemrograman Web Lanjut\pwl_pos> php artisan make:controller Api/RegisterContro
ller

INFO Controller [D:\Polinema\Semester 4\Pemrograman Web Lanjut\pwl_pos\app\Http\Controllers\Api\Regi
sterController.php] created successfully.

PS D:\Polinema\Semester 4\Pemrograman Web Lanjut\pwl_pos>
```

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

Jawab:

```
Explorer (Ctrl+Shift+E) - 1 unsaved file RegisterController.php U X
app > Http > Controllers > Api > RegisterController.php > ...

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

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

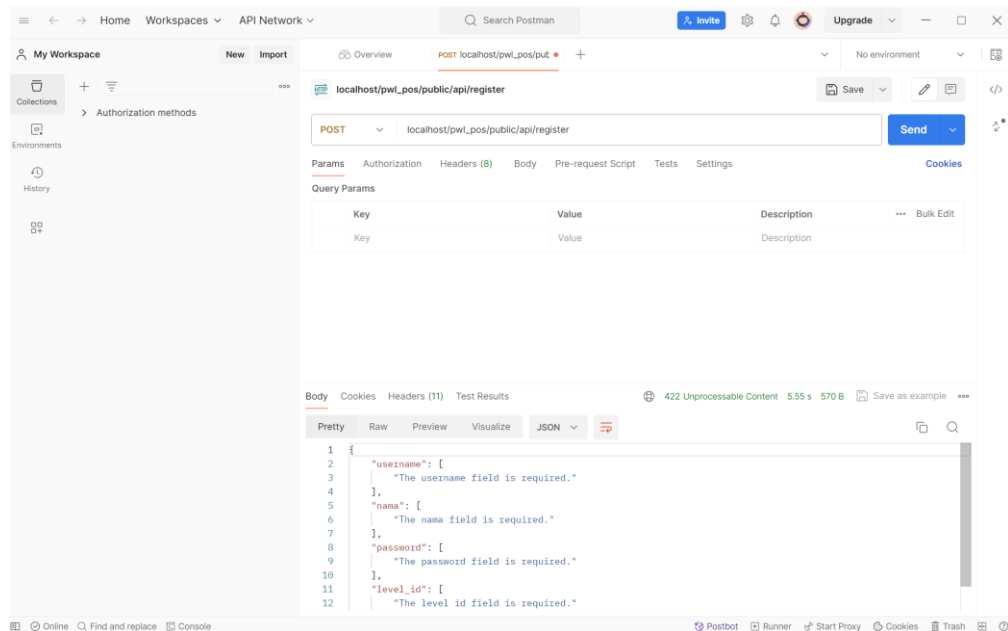
Jawab:

```
routes > api.php
You, 1 second ago | 1 author (You)

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

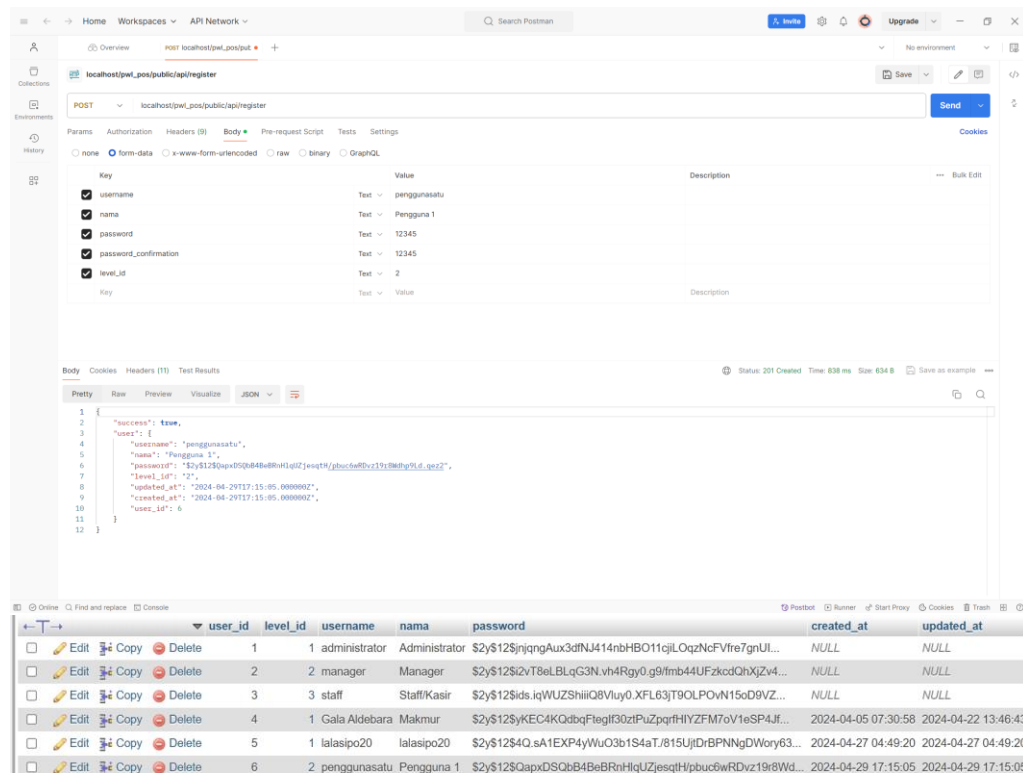
11. Jika sudah, kita akan melakukan uji coba REST API melalui aplikasi postman. Buka aplikasi Postman, isi URL localhost/PWL\_POS/public/api/register serta method POST. Klik Send.

Jawab:



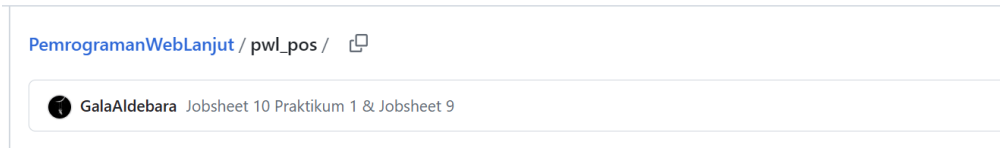
12. Sekarang kita coba masukkan data. Klik tab Body dan pilih form-data. Isikan key sesuai dengan kolom data, serta isikan data registrasi menggunakan nilai yang anda inginkan.

Jawab:



13. Lakukan commit perubahan file pada Github.

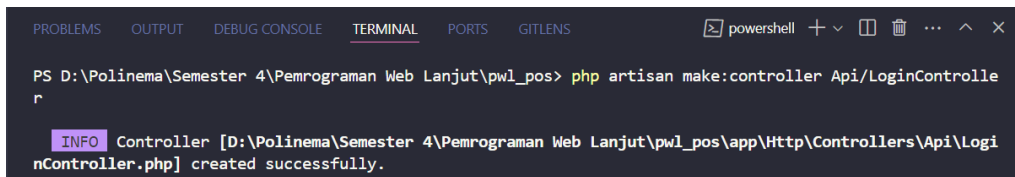
Jawab:



## PRAKTIKUM 2 – Membuat Restful API Login

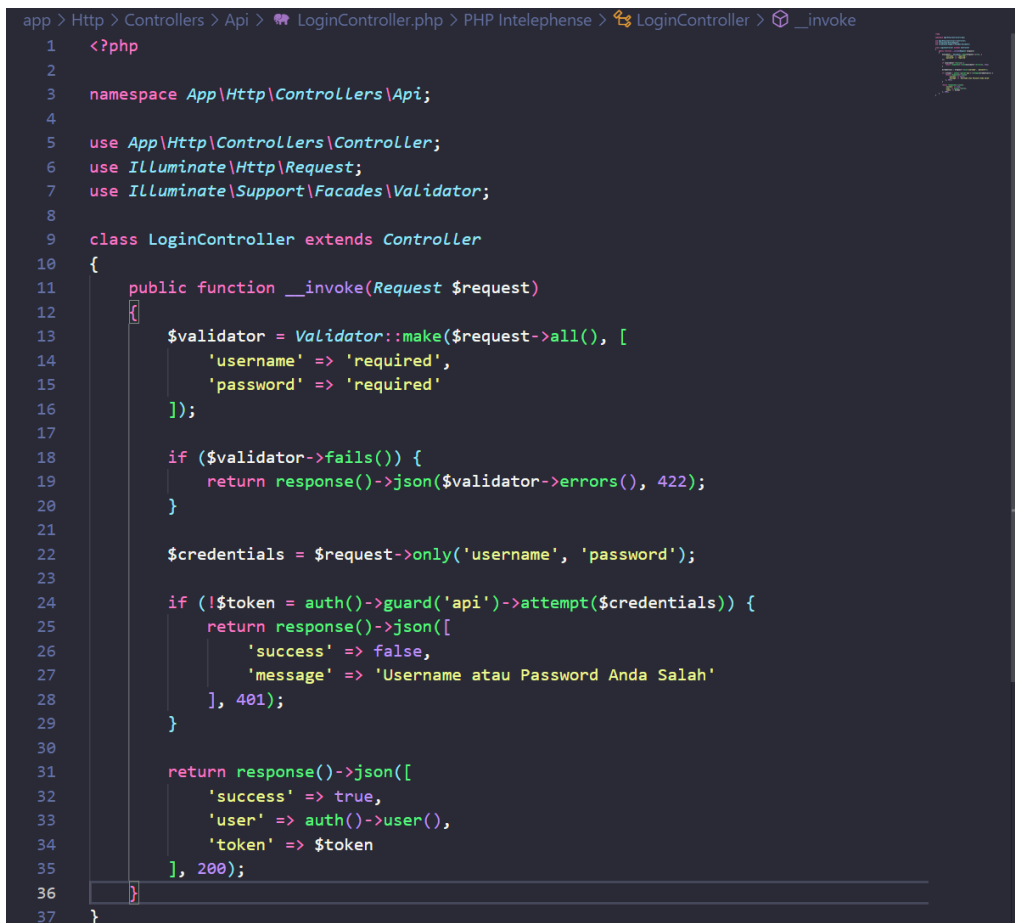
1. Kita buat file controller dengan nama LoginController

Jawab:



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

Jawab:



- Jawab:

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.

- 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.

The screenshot displays the Swagger UI for a REST API. The selected endpoint is a POST request to `localhost/pwt_pos/public/api/login`. The request body is a JSON object with the following fields:

- `username`: penggunasatu
- `password`: 12345
- `updated_at`: 2024-04-29T17:15:05.000000Z

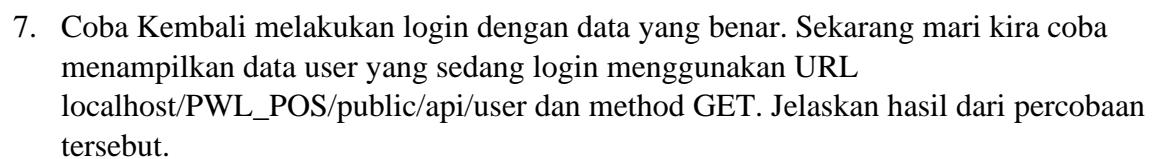
The response is a 200 OK status with a JSON body containing:

- `success`: true
- `users`: an array of user objects, each containing `user_id`, `level_id`, `username`, `nama`, `password`, `created_at`, and `updated_at`.
- `token`: a long JWT token.

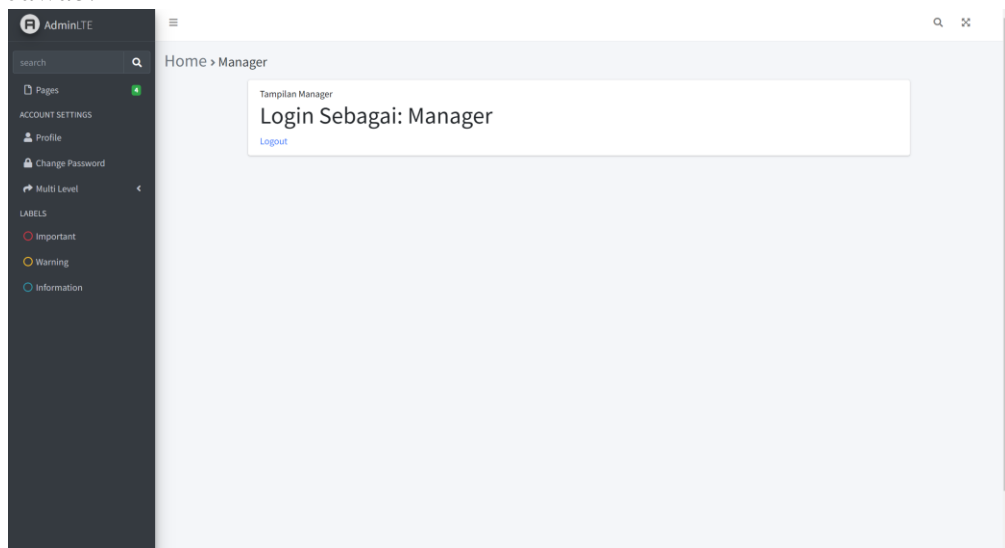


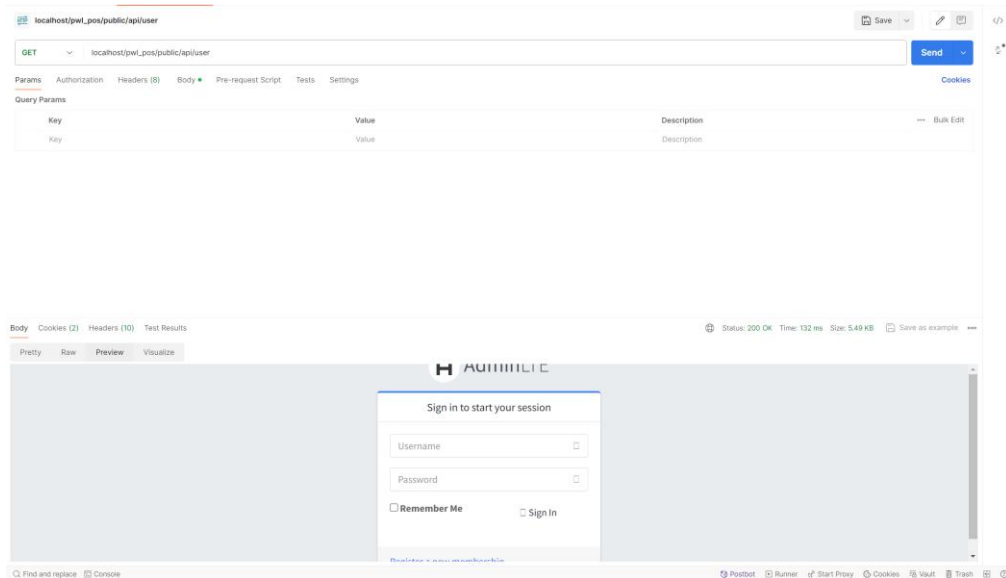
[illegible]

- Jawab;



Jawab:





- Lakukan commit perubahan file apda Github.  
Jawab:

GalaAldebara	Jobsheet 10 - Praktikum 2	a3cd43c · now	20 Commits
Post	Hasil Praktikum 1 Jobsheet 4	2 months ago	
pertemuan2	Menambahkan File Jobsheet Pertemuan3	2 months ago	
pwl_pos	Jobsheet 10 - Praktikum 2	now	
Pertemuan1PWL_TI2C_15_M.Iqbal.M.pdf	Menambahkan File Pertemuan 1	3 months ago	
Pertemuan2PWL_TI2C_15_M.Iqbal.pdf	Menambahkan File Jobsheet Pertemuan3	2 months ago	
Pertemuan3WPL_TI2C_14_M.Iqbal.pdf	Menambahkan File Jobsheet Pertemuan3	2 months ago	
Pertemuan4_TI2C_15_M.Iqbal.Makmur.pdf	Menambahkan File Jobsheet 4	2 months ago	
Pertemuan5_TI2C_15_M.Iqbal.Makmur.pdf	"Hasil Jobsheet 5"	2 months ago	
Pertemuan6_TI2C_15_M.Iqbal.Makmur.pdf	Hasil Jobsheet 6	2 months ago	
Pertemuan7_TI2C_15_M.Iqbal.Makmur.pdf	Pertemuan 7	2 weeks ago	
Pertemuan9_TI2C_15_M.Iqbal.Makmur.pdf	Jobsheet 10 Praktikum 1 & Jobsheet 9	last week	

### ***PRAKTIKUM 3 – Membuat Restful API Logout***

- Tambahkan kode berikut pada file .env  
Jawab:

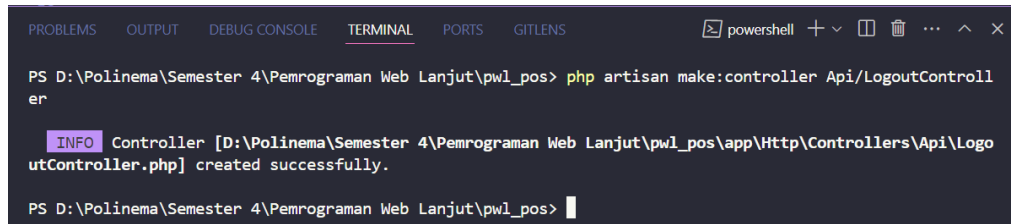
```

61     JWT_SECRET=sN2xhsIV12g2HLNq0np44cqc1Je
62     JWT_SHOW_BLACKLIST_EXCEPTION=true

```

2. Buat Controller baru dengan nama LogoutController.

Jawab:



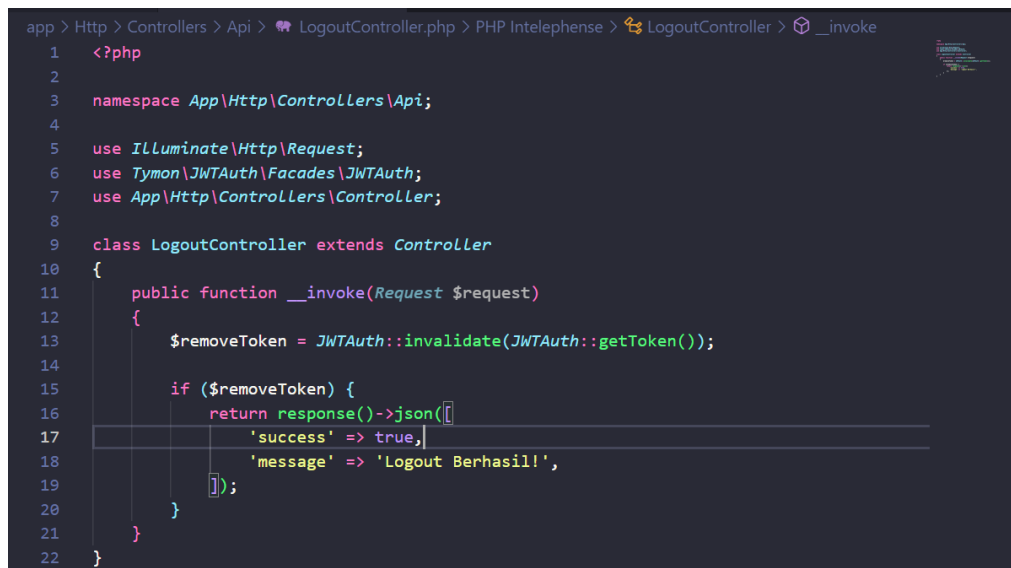
```
PS D:\Polinema\Semester 4\Pemrograman Web Lanjut\pwl_pos> php artisan make:controller Api/LogoutController

[INFO] Controller [D:\Polinema\Semester 4\Pemrograman Web Lanjut\pwl_pos\app\Http\Controllers\Api\LogoutController.php] created successfully.

PS D:\Polinema\Semester 4\Pemrograman Web Lanjut\pwl_pos>
```

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

Jawab:

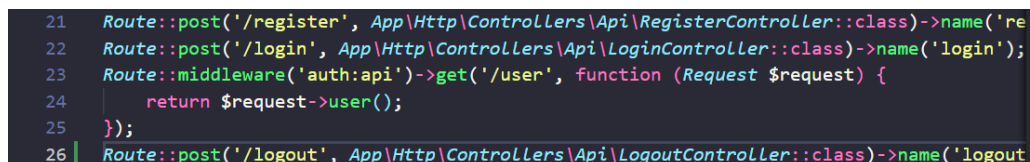


```
app > Http > Controllers > Api > LogoutController.php > PHP Intelephense > LogoutController > __invoke

1  <?php
2
3  namespace App\Http\Controllers\Api;
4
5  use Illuminate\Http\Request;
6  use Tymon\JWTAuth\Facades\JWTAuth;
7  use App\Http\Controllers\Controller;
8
9  class LogoutController extends Controller
10 {
11     public function __invoke(Request $request)
12     {
13         $removeToken = JWTAuth::invalidate(JWTAuth::getToken());
14
15         if ($removeToken) {
16             return response()->json([
17                 'success' => true,
18                 'message' => 'Logout Berhasil!',
19             ]);
20         }
21     }
22 }
```

4. Lalu kita tambahkan routes pada api.php

Jawab:

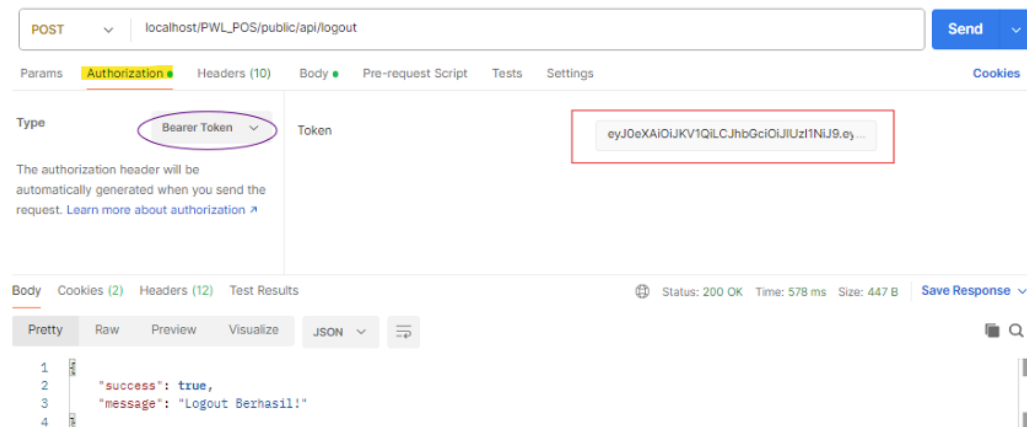


```
21 Route::post('/register', App\Http\Controllers\Api\RegisterController::class)->name('register');
22 Route::post('/login', App\Http\Controllers\Api>LoginController::class)->name('login');
23 Route::middleware('auth:api')->get('/user', function (Request $request) {
24     return $request->user();
25 });
26 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.

- Isi token pada tab Authorization, pilih type yaitu Bearer Token. Isikan token yang didapat saat login. Jika sudah klik Send.

Jawab:



- Lakukan commit perubahan file pada github.

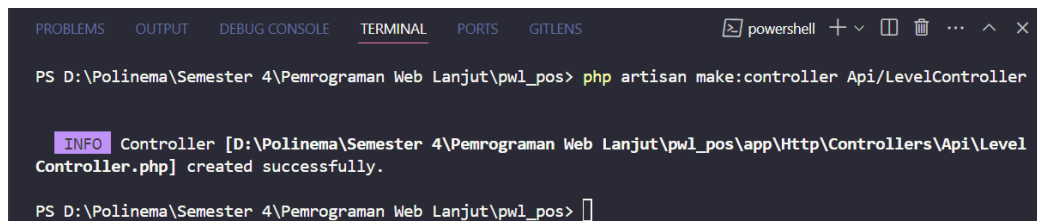
Jawab:

GalaAldebara	Jobsheet 10 - Praktikum 3	2eaa1bf · now	🕒 21 Commits
Post	Hasil Praktikum 1 Jobsheet 4		2 months ago
pertemuan2	Menambahkan File Jobsheet Pertemuan3		2 months ago
pwl_pos	Jobsheet 10 - Praktikum 3		now
Pertemuan1PWL_Ti2C_15_M.Iqbal.M.pdf	Menambahkan File Pertemuan 1		3 months ago
Pertemuan2PWL_Ti2C_15_M.Iqbal.pdf	Menambahkan File Jobsheet Pertemuan3		2 months ago
Pertemuan3WPL_Ti2C_14_M.Iqbal.pdf	Menambahkan File Jobsheet Pertemuan3		2 months ago
Pertemuan4_Ti2C_15_M.Iqbal.Makmur.pdf	Menambahkan File Jobsheet 4		2 months ago
Pertemuan5_Ti2C_15_M.Iqbal.Makmur.pdf	"Hasil Jobsheet 5"		2 months ago
Pertemuan6_Ti2C_15_M.Iqbal.Makmur.pdf	Hasil Jobsheet 6		2 months ago
Pertemuan7_Ti2C_15_M.Iqbal.Makmur.pdf	Pertemuan 7		2 weeks ago
Pertemuan9_Ti2C_15_M.Iqbal.Makmur.pdf	Jobsheet 10 Praktikum 1 & Jobsheet 9		last week

## PRAKTIKUM 4 – Implementasi CRUD dalam RESTful API

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

Jawab:



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

Jawab:

```
app > Http > Controllers > Api > LevelController.php > PHP Intelephense > 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     {
13         return LevelModel::all();
14     }
15
16     public function store(Request $request)
17     {
18         $level = LevelModel::create($request->all());
19         return response()->json($level, 201);
20     }
21
22     public function show(LevelModel $level)
23     {
24         return LevelModel::find($level);
25     }
26
27     public function update(Request $request, LevelModel $level)
28     {
29         $level->update($request->all());
30         return LevelModel::find($level);
31     }
32
33     public function destroy(LevelModel $user)
34     {
35         $user->delete();
36
37         return response()->json([
38             'success' => true,
39             'message' => 'Data Terhapus',
40         ]);
41     }
42 }
```

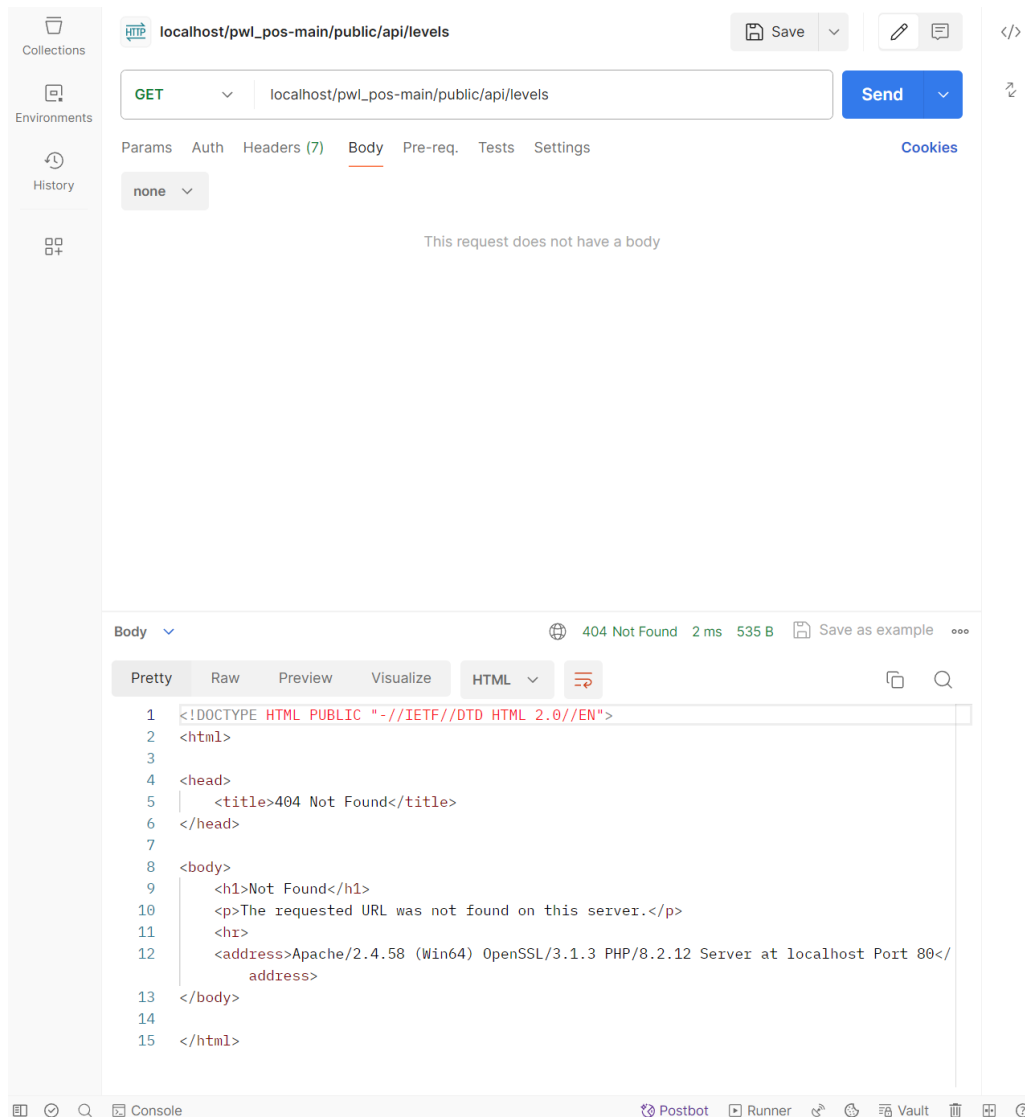
- Kemudian kita lengkapi routes pada api.php

Jawab:

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

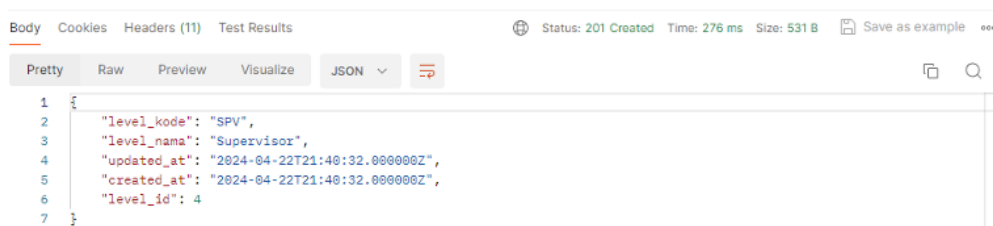
- Jika sudah. Lakukan uji coba API mulai dari fungsi untuk menampilkan data. Gunakan URL: localhost/PWL\_POS-main/public/api/levels dan method GET.

Jawab:



5. Kemudian, lakukan percobaan penambahan data dengan URL : localhost/PWL\_POS-main/public/api/levels dan method POST seperti di bawah ini.

Jawab:



6. Berikutnya lakukan percobaan menampilkan detail data. Jelaskan dan berikan screenshoot hasil percobaan Anda.
7. Jika sudah, kita coba untuk melakukan edit data menggunakan localhost/PWL\_POS-main/public/api/levels/{id} dan method PUT. Isika data yang ingin diubah pada tab Param.

Jawab:

```
body Cookies Headers (11) Test Results
Status: 200 OK Time: 266 ms Size: 528 B Save as example
Pretty Raw Preview Visualize JSON
1 {
2   {
3     "level_id": 4,
4     "level_kode": "SPR",
5     "level_nama": "Supervisor",
6     "created_at": "2024-04-22T21:48:32.000000Z",
7     "updated_at": "2024-04-22T21:48:19.000000Z"
8   }
9 }
```

8. Terakhir lakukan percobaan hapus data.
9. Lakukan commit perubahan file pada GitHub

Jawab:

main	1 Branch	0 Tags	Go to file	Add file	Code
GalaAldebara	Jobsheet 10 - Praktikum 4	256a5d7 - 2 minutes ago	22 Commits		
Post	Hasil Praktikum 1 Jobsheet 4	2 months ago			
pertemuan2	Menambahkan File Jobsheet Pertemuan3	2 months ago			
pwl_pos	Jobsheet 10 - Praktikum 4	2 minutes ago			
Pertemuan1PWL_TI2C_15_M.Iqbal.M.pdf	Menambahkan File Pertemuan 1	3 months ago			
Pertemuan2PWL_TI2C_15_M.Iqbal.pdf	Menambahkan File Jobsheet Pertemuan3	2 months ago			
Pertemuan3WPL_TI2C_14_M.Iqbal.pdf	Menambahkan File Jobsheet Pertemuan3	2 months ago			
Pertemuan4_TI2C_15_M.Iqbal.Makmur.pdf	Menambahkan File Jobsheet 4	2 months ago			
Pertemuan5_TI2C_15_M.Iqbal.Makmur.pdf	"Hasil Jobsheet 5"	2 months ago			
Pertemuan6_TI2C_15_M.Iqbal.Makmur.pdf	Hasil Jobsheet 6	2 months ago			
Pertemuan7_TI2C_15_M.Iqbal.Makmur.pdf	Pertemuan 7	2 weeks ago			
Pertemuan9_TI2C_15_M.Iqbal.Makmur.pdf	Jobsheet 10 Praktikum 1 & Jobsheet 9	last week			

## TUGAS

Implementasikan CRUD API pada tabel lainnya yaitu tabel m\_user, m\_kategori, dan m\_barang

### LevelModel

```
app > Models > LevelModel.php > ...
You, 8 minutes ago | 1 author (You)
1 <?php
2
3 namespace App\Models;
4
5 use Illuminate\Database\Eloquent\Model;
6 use Illuminate\Database\Eloquent\Relations\BelongsTo;
7 use Illuminate\Database\Eloquent\Factories\HasFactory;
8
9 class LevelModel extends Model
10 {
11     use HasFactory;
12
13     protected $table = 'm_level';
14     protected $primaryKey = 'level_id';
15
16     protected $fillable = ['level_kode', 'level_nama'];
17
18     public function users(): BelongsTo
19     {
20         return $this->belongsTo(UserModel::class, 'user_id', 'user_id');
21     }
22 }
```

**INFO** Controller [D:\Polinema\Semester 4\Pemrograman Web Lanjut\pwl\_pos\app\Http\Controllers\Api\BarangController.php] created successfully.

PS D:\Polinema\Semester 4\Pemrograman Web Lanjut\pwl\_pos> php artisan make:controller Api/KategoriController

**INFO** Controller [D:\Polinema\Semester 4\Pemrograman Web Lanjut\pwl\_pos\app\Http\Controllers\Api\KategoriController.php] created successfully.

PS D:\Polinema\Semester 4\Pemrograman Web Lanjut\pwl\_pos> |

## KategoriController

```
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     {
13         return KategoriModel::all();
14     }
15
16     public function store(Request $request)
17     {
18         $kategori = KategoriModel::create($request->all());
19         return response()->json($kategori, 201);
20     }
21
22     public function show(KategoriModel $kategori)
23     {
24         return KategoriModel::find($kategori);
25     }
26
27     public function update(Request $request, KategoriModel $kategori)
28     {
29         $kategori->update($request->all());
30         return KategoriModel::find($kategori);
31     }
32
33     public function destroy(KategoriModel $kategori)
34     {
35         $kategori->delete();
36
37         return response()->json([
38             'success' => true,
39             'message' => 'Data terhapus'
40         ]);
41     }
42 }
43
```

## BarangController

```
app > Http > Controllers > Api > BarangController.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\BarangModel;
8
9 class BarangController extends Controller
10 {
11     public function index()
12     {
13         return BarangModel::all();
14     }
15
16     public function store(Request $request)
17     {
18         $barang = BarangModel::create($request->all());
19         return response()->json($barang, 201);
20     }
21
22     public function show(BarangModel $barang)
23     {
24         return BarangModel::find($barang);
25     }
26
27     public function update(Request $request, BarangModel $barang)
28     {
29         $barang->update($request->all());
30         return BarangModel::find($barang);
31     }
32
33     public function destroy(BarangModel $barang)
34     {
35         $barang->delete();
36
37         return response()->json([
38             'success' => true,
39             'message' => 'Data terhapus'
40         ]);
41     }
42 }
43
```



## UserController

```
app > Http > Controllers > Api > UserController.php > PHP Intelephense > UserController > 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\UserModel;
8
9  class UserController extends Controller
10 {
11     public function index()
12     {
13         return UserModel::all();
14     }
15
16     public function store(Request $request)
17     {
18         $user = UserModel::create($request->all());
19         return response()->json($user, 201);
20     }
21
22     public function show(UserModel $user)
23     {
24         return UserModel::find($user);
25     }
26
27     public function update(Request $request, UserModel $user)
28     {
29         $user->update($request->all());
30         return UserModel::find($user);
31     }
32
33     public function destroy(UserModel $user)
34     {
35         $user->delete();
36         return response()->json([
37             'success' => true,
38             'message' => 'Data terhapus'
39         ]);
40     }
41 }
42 }
```

## Route

```
26 Route::post('/register', App\Http\Controllers\Api\RegisterController::class->name('register'));
27 Route::post('/login', App\Http\Controllers\Api\LoginController::class->name('login'));
28 Route::middleware('auth:api')->get('/user', function (Request $request) {
29     return $request->user();
30 });
31 Route::post('/logout', App\Http\Controllers\Api\LogoutController::class->name('logout'));
32
33 Route::get('levels', [LevelController::class, 'index']);
34 Route::post('levels', [LevelController::class, 'store']);
35 Route::get('levels/{level}', [LevelController::class, 'show']);
36 Route::put('levels/{level}', [LevelController::class, 'update']);
37 Route::delete('levels/{level}', [LevelController::class, 'destroy']);
38
39 Route::get('users', [UserController::class, 'index']);
40 Route::post('users', [UserController::class, 'store']);
41 Route::get('users/{user}', [UserController::class, 'show']);
42 Route::put('users/{user}', [UserController::class, 'update']);
43 Route::delete('users/{user}', [UserController::class, 'destroy']);
44
45 Route::get('kategori', [KategoriController::class, 'index']);
46 Route::post('kategori', [KategoriController::class, 'store']);
47 Route::get('kategori/{kategori}', [KategoriController::class, 'show']);
48 Route::put('kategori/{kategori}', [KategoriController::class, 'update']);
49 Route::delete('kategori/{kategori}', [KategoriController::class, 'destroy']);
50
51 Route::get('barang', [BarangController::class, 'index']);
52 Route::post('barang', [BarangController::class, 'store']);
53 Route::get('barang/{barang}', [BarangController::class, 'show']);
54 Route::put('barang/{barang}', [BarangController::class, 'update']);
55 Route::delete('barang/{barang}', [BarangController::class, 'destroy']);
```