LAPORAN JOBSHEET 10
PEMROGRAMAN WEB LANJUT
RESTFUL API

Dosen pengajar: Dimas Wahyu Wibowo, ST., MT

Disusun Oleh:

Nama : M. Fatih Al Ghifary

NIM : 2341720194

Kelas : TI-2A

No : 16

PRODI TEKNIK INFORMATIKA D-IV

JURUSAN TEKNOLOGI INFORMASI

POLITEKNIK NEGERI MALANG

2025

Praktikum 1 – Membuat RESTful API Register

1. Download aplikasi Postman di https://www.postman.com/downloads.
2. Lakukan instalasi JWT dengan mengetikkan perintah berikut:
   composer require tymon/jwt-auth:2.1.1

```
D:\KULIAH\SEMESTER-4\file\laragon\www\PROWEBLNJT\Week 10\PWL_POS>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)
```

3. publish konfigurasi file dengan perintah berikut:
   php artisan vendor:publish --provider="Tymon\JWTAuth\Providers\LaravelServiceProvider"

```
D:\KULIAH\SEMESTER-4\file\laragon\www\PROWEBLNJT\Week 10\PWL_POS>php artisan vendor:publish --provider="Tymon\JWTAuth\Pr
oviders\LaravelServiceProvider"

  INFO  Publishing assets.

  Copying file [D:\KULIAH\SEMESTER-4\file\laragon\www\PROWEBLNJT\Week 10\PWL_POS\vendor\tymon\jwt-auth\config\config.php
] to [D:\KULIAH\SEMESTER-4\file\laragon\www\PROWEBLNJT\Week 10\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.

Week 10 > PWL_POS > config > jwt.php

```php
<?php

/*
 * This file is part of jwt-auth.
 *
 * (c) Sean Tymon <tymon148@gmail.com>
 *
 * For the full copyright and license information, please view the LICENSE
 * file that was distributed with this source code.
 */
```

5. Jalankan perintah berikut untuk membuat secret key JWT.
   php artisan jwt:secret

```
D:\KULIAH\SEMESTER-4\file\laragon\www\PROWEBLNJT\Week 10\PWL_POS>php artisan jwt:secret
jwt-auth secret [WUKs2PvsWZ4B73SDYxxvuqra2aYAmiJ9RhOi2mzC0cwWnEpCzQy6ppu2Y6w3dcFN] set successfully.
```

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

```
JWT_SECRET=WUKs2PvsWZ4B73SDYxxvuqra2aYAmiJ9RhOi2mzC0cwWnEpCzQy6ppu2Y6w3dcFN
```

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

```
    'guards' => [
        'web' => [
            'driver' => 'session',
            'provider' => 'users',
        ],
        'api' => [
            'driver' => 'jwt',
            'provider' => 'users',
        ],
    ],
```

7. Tambahkan kode di model UserModel

```
class UserModel extends Authenticatable implements JWTSubject
{
    use HasFactory;
    0 references | 0 overrides
    public function getJWTIdentifier(): mixed
    {
        return $this->getKey();
    }
    0 references | 0 overrides
    public function getJWTCustomClaims(): array
    {
        return [];
    }

    0 references
    protected $table = 'm_user';
    0 references
    protected $primaryKey = 'user_id';
```

8. Membuat controller untuk register dengan menjalankan peintah *php artisan make:controller Api/RegisterController*

```
D:\KULIAH\SEMESTER-4\file\laragon\www\PROWEBLNJT\Week 10\PWL_POS>php artisan make:controller Api/RegisterController

  INFO  Controller [D:\KULIAH\SEMESTER-4\file\laragon\www\PROWEBLNJT\Week 10\PWL_POS\app\Http\Controllers\Api\RegisterC
ontroller.php] created successfully.
```

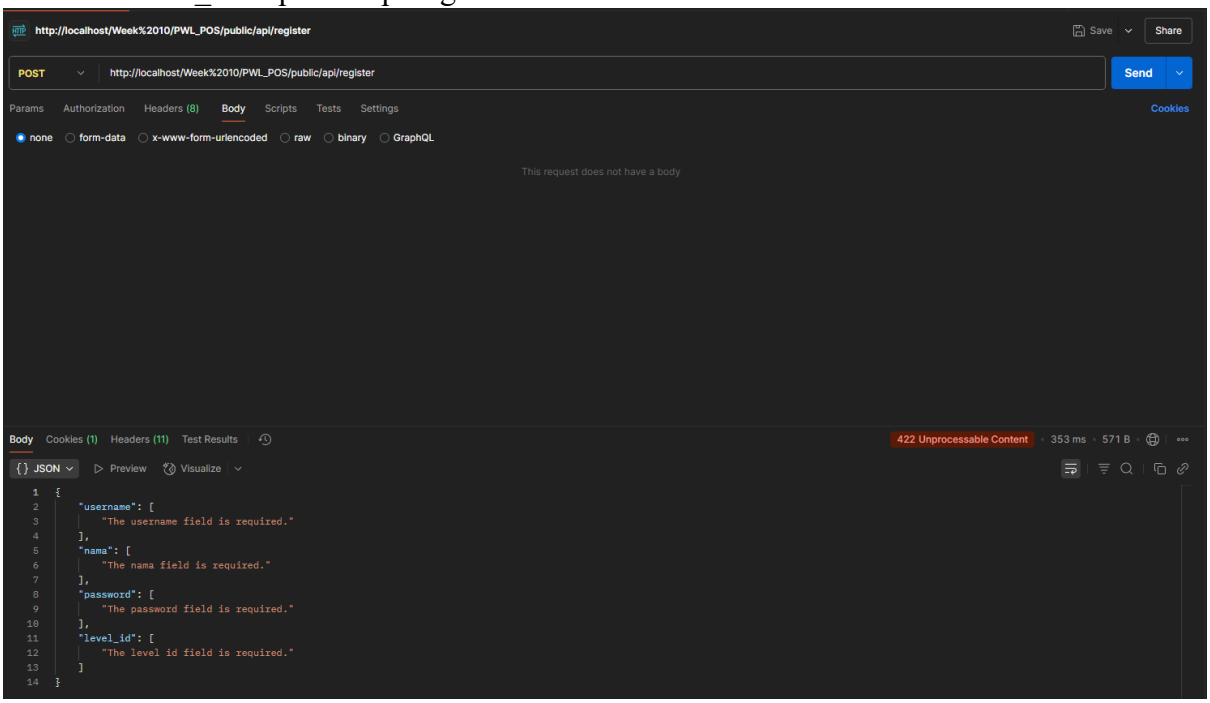
9. Ubah kode RegisterController

```php
<?php

namespace App\Http\Controllers\Api;

use App\Http\Controllers\Controller;
use Illuminate\Http\Request;
use App\Models\UserModel;
use Illuminate\Support\Facades\Validator;

2 references | 0 implementations
class RegisterController extends Controller
{
    0 references | 0 overrides
    public function __invoke(Request $request): JsonResponse|mixed
    {
        //set validation
        $validator = Validator::make($request->all(), [
            'username' => 'required',
            'nama' => 'required',
            'password' => 'required|min:5|confirmed',
            'level_id' => 'required'
        ]);

        //if validations fails
        if ($validator->fails()) {
            return response()->json($validator->errors(), 422);
        }

        //create user
        $user = UserModel::create([
            'username' => $request->username,
            'nama' => $request->nama,
            'password' => bcrypt(value: $request->password),
            'level_id' => $request->level_id,
        ]);

        //return response JSON user is created
        if ($user) {
            return response()->json([
                'success' => true,
                'user' => $user,
            ], 201);
        }
```

10. Ubah kode routes/api.php

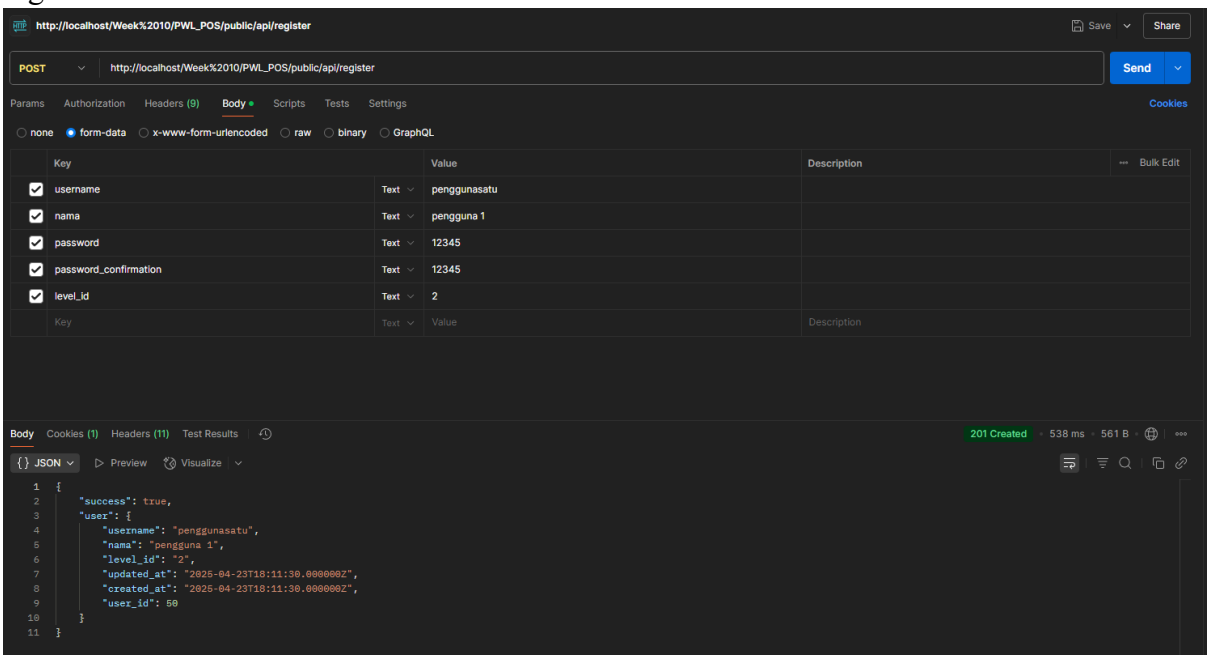
```php
Route::post('/register', RegisterController::class)->name('register');
```

11. 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.

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.



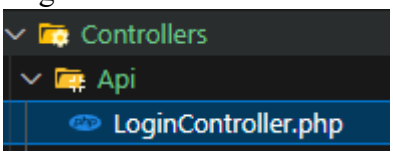
13. Lakukan commit perubahan file pada Github. https://github.com/fateehhh/PROWEBLNJT/commit/0479dc0ca13f6820ef8515c59c1123c60c48b334

Praktikum 2 – Membuat RESTful API Login

1. Membuat LoginController menggunakan perintah
   php artisan make:controller Api/LoginController

```
D:\KULIAH\SEMESTER-4\file\laragon\www\PROWEBLNJT\Week 10\PWL_POS>php artisan make:controller Api/LoginController

  INFO  Controller [D:\KULIAH\SEMESTER-4\file\laragon\www\PROWEBLNJT\Week 10\PWL_POS\app\Http\Controllers\Api\LoginCont
roller.php] created successfully.
```

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



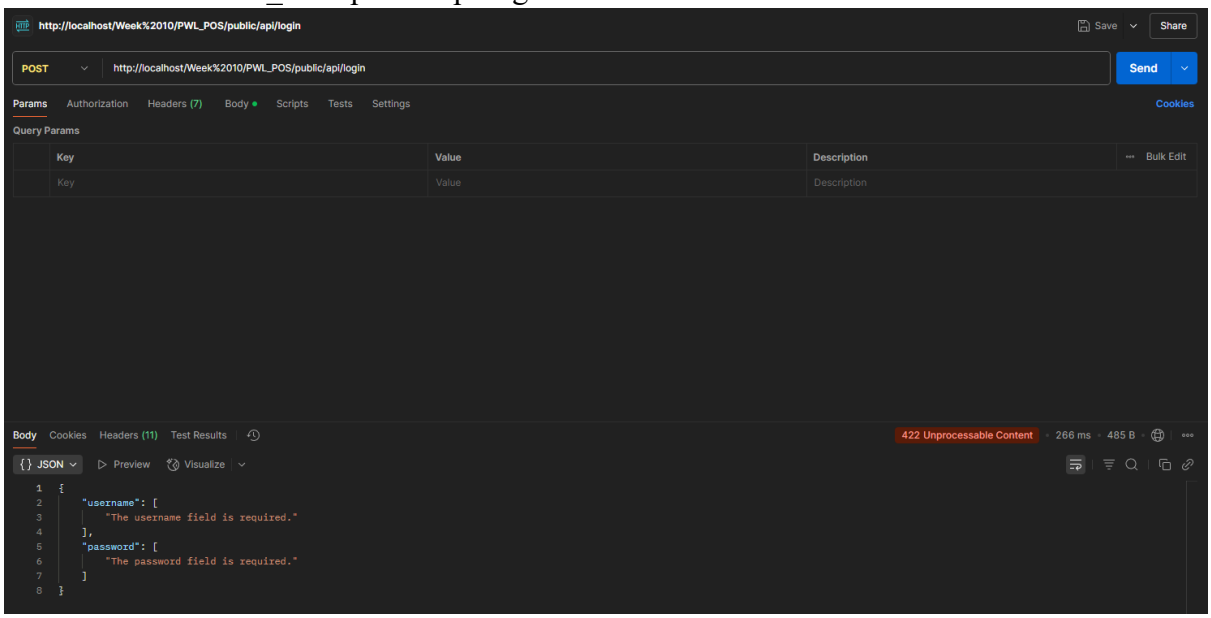
2. Modifikasi Login Controller

```php
<?php

namespace App\Http\Controllers\Api;

use App\Http\Controllers\Controller;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Validator;

0 references | 0 implementations
class LoginController extends Controller
{
    0 references | 0 overrides
    public function __invoke(Request $request): JsonResponse|mixed
    {
        $validator = Validator::make($request->all(), [
            'username' => 'required',
            'password' => 'required'
        ]);

        if ($validator->fails()) {
            return response()->json($validator->errors(), 422);
        }

        $credentials = $request->only('username', 'password');

        if (!$token = auth()->guard('api')->attempt(credentials: $credentials)) {
            return response()->json([
                'success' => false,
                'message' => 'Username atau Password Anda salah'
            ], 401);
        }

        return response()->json([
            'success' => true,
            'user' => auth()->guard('api')->user(),
            'token' => $token,
        ], 200);
    }
}
```

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

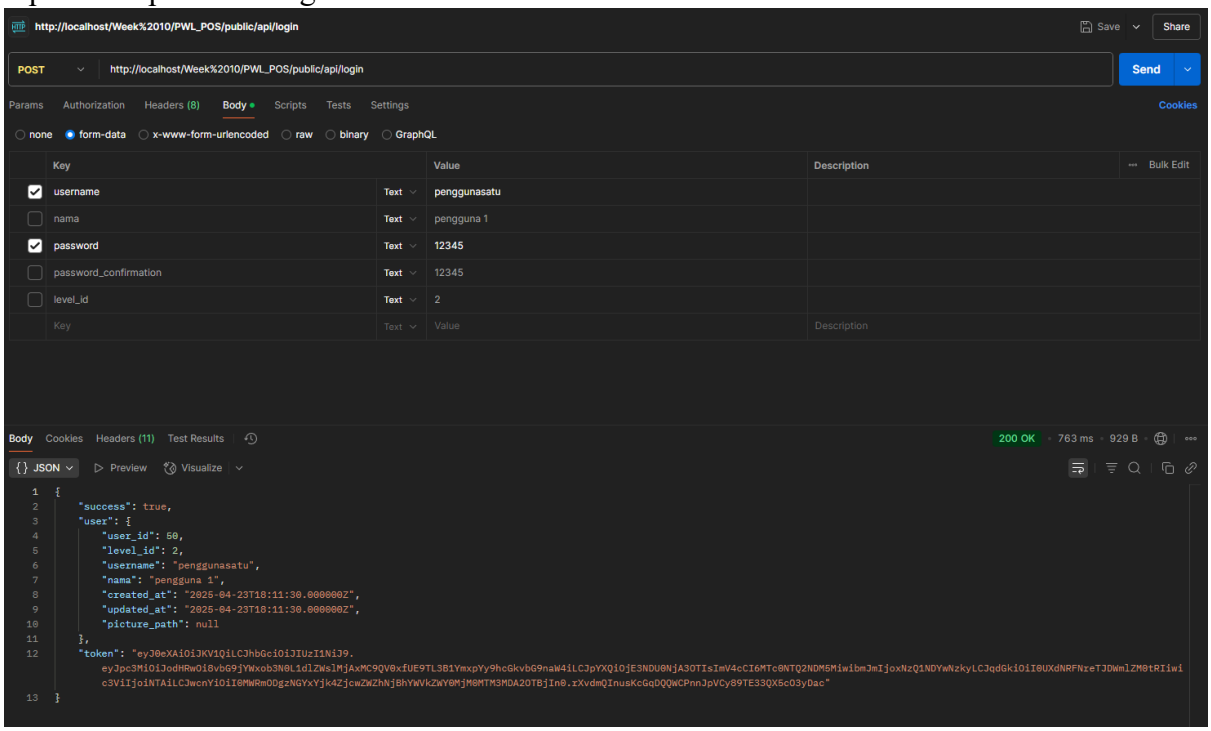
```php
Route::post('/login', App\Http\Controllers\Api\LoginController::class)->name('login');
Route::middleware('auth:api')->get('/user', function (Request $request): mixed {
    return $request->user();
});
```

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



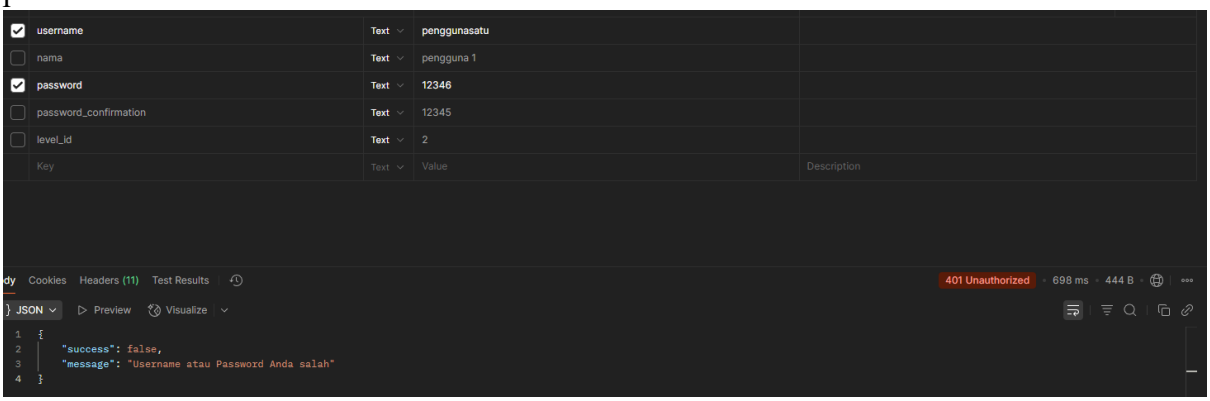
Jika berhasil akan muncul error validasi seperti gambar di atas.

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.



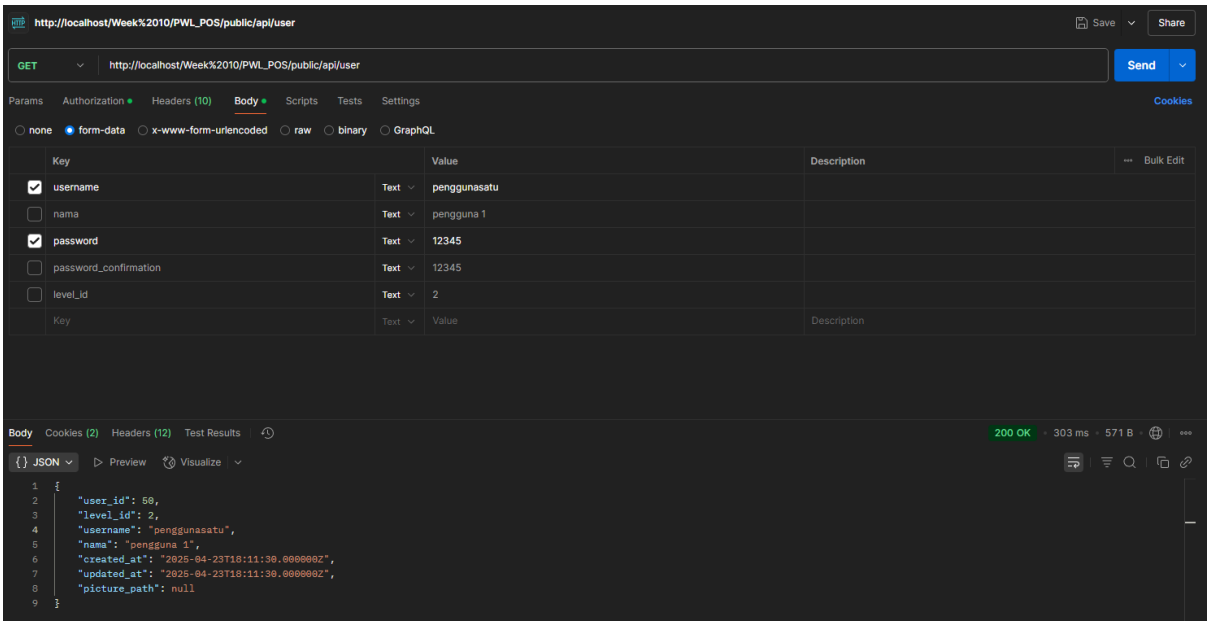
Token:
eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJpc3MiOiJodHRwOi8vbG9jYWxob3N0L1dlZWslMjAxMC9QV0xfUE9TL3B1YmxpYy9hcGkvbG9naW4iLCJpYXQiOjE3NDU0NjA3OTIsImV4cCI6MTc0NTQ2NDM5MiwibmJmIjoxNzQ1NDYwNzkyLCJqdGkiOiI0UXdNRFNreTJDWmlZM0tRIiwic3ViIjoiNTAiLCJwcnYiOiI0MWRmODgzNGYxYjk4ZjcwZWZhNjBhYWVkZWY0MjM0MTM3MDA2OTBjIn0.rXvdmQInusKcGqDQQWCPnnJpVCy89TE33QX5cO3yDac

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



disini passwordnya salah, sehingga menampilkan command error

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.



Buka Authorization dulu, ubah Auth typenya jadi Bearer Token, kemudian masukkan token sesuai dengan yang sudah kita copykan

8. Lakukan commit perubahan file pada Github.
https://github.com/fateehhh/PROWEBLNJT/commit/2220f43502ae68a926e6966d2ff40644bb97944e

Praktikum 3 – Membuat RESTful API Logout

1. Tambahkan kode berikut pada file .env
   JWT_SHOW_BLACKLIST_EXCEPTION=true

```
JWT_SECRET=WUKs2PvsWZ4B73SDYxxvuqra2aYAmiJ9RhOi2mzC0cwWnEpCzQy6ppu2Y6w3dcFN
JWT_SHOW_BLACKLIST_EXCEPTION=true
```

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

```
D:\KULIAH\SEMESTER-4\file\laragon\www\PROWEBLNJT\Week 10\PWL_POS>php artisan make:controller Api/LogoutController

  INFO  Controller [D:\KULIAH\SEMESTER-4\file\laragon\www\PROWEBLNJT\Week 10\PWL_POS\app\Http\Controllers\Api\LogoutCon
troller.php] created successfully.
```

3. Modifikasi LogoutController

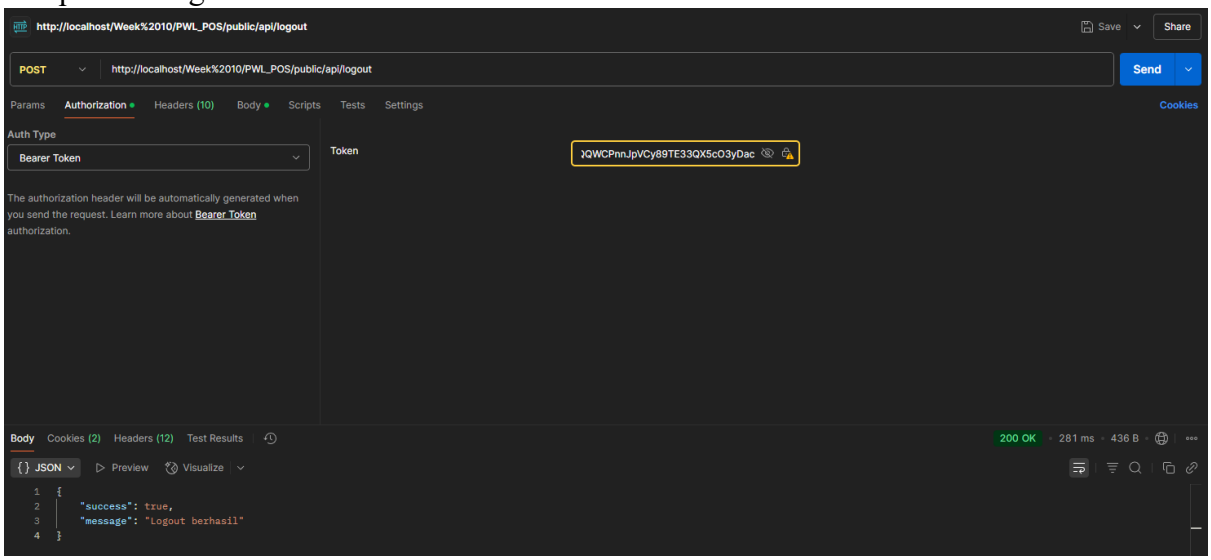
```php
<?php

namespace App\Http\Controllers\Api;

use App\Http\Controllers\Controller;
use Illuminate\Http\Request;
use Tymon\JWTAuth\Facades\JWTAuth;

0 references | 0 implementations
class LogoutController extends Controller
{
    0 references | 0 overrides
    public function __invoke(Request $request): JsonResponse|mixed
    {
        $removeToken = JWTAuth::invalidate(JWTAuth::getToken());

        if ($removeToken) {
            return response()->json([
                'success' => true,
                'message' => 'Logout berhasil'
            ]);
        }
    }
}
```

4. Kita tambahkan routes pada api.php

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

5. 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. https://github.com/fateehhh/PROWEBLNJT/commit/693ff0466d505d9fec69d5a50e88c59709a000c1

Praktikum 4 – Implementasi CRUD dalam RESTful API

1. Buat controller untuk mengolah API pada data level. php artisan make:controller Api/LevelController

```
D:\KULIAH\SEMESTER-4\file\laragon\www\PROWEBLNJT\Week 10\PWL_POS>php artisan make:controller Api/LevelController

  INFO  Controller [D:\KULIAH\SEMESTER-4\file\laragon\www\PROWEBLNJT\Week 10\PWL_POS\app\Http\Controllers\Api\LevelCont
roller.php] created successfully.
```

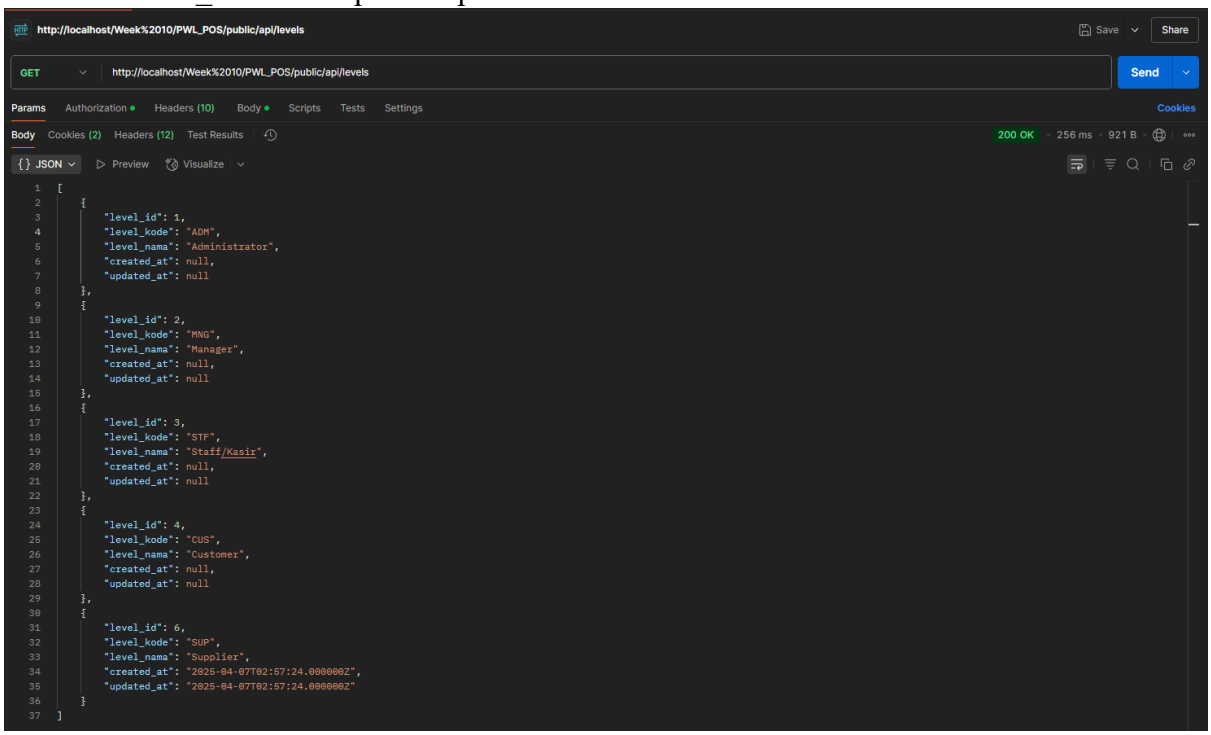
2. Modifikasi LevelController berisi fungsi CRUD

```php
class LevelController extends Controller
{
    1 reference | 0 overrides
    public function index(): Collection
    {
        return LevelModel::all();
    }

    1 reference | 0 overrides
    public function store(Request $request): JsonResponse|mixed
    {
        $level = LevelModel::create($request->all());
        return response()->json($level, 201);
    }

    1 reference | 0 overrides
    public function show(LevelModel $level): array|Collection|LevelMode
    {
        return LevelModel::find($level);
    }

    1 reference | 0 overrides
    public function update(Request $request, LevelModel $level): array
    {
        $level->update($request->all());
        return LevelModel::find($level);
    }

    1 reference | 0 overrides
    public function destroy(LevelModel $user): JsonResponse|mixed
    {
        $user->delete();
        return response()->json([
            'success' => true,
            'message' => 'Data Terhapus',
        ]);
    }
}
```

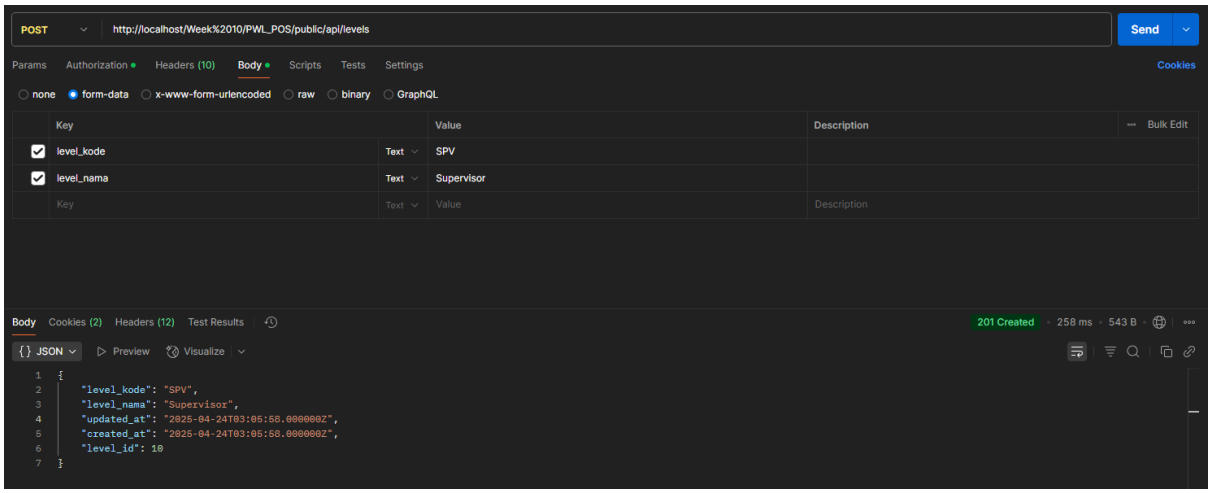
3. kita lengkapi routes pada api.php

```php
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']);
```

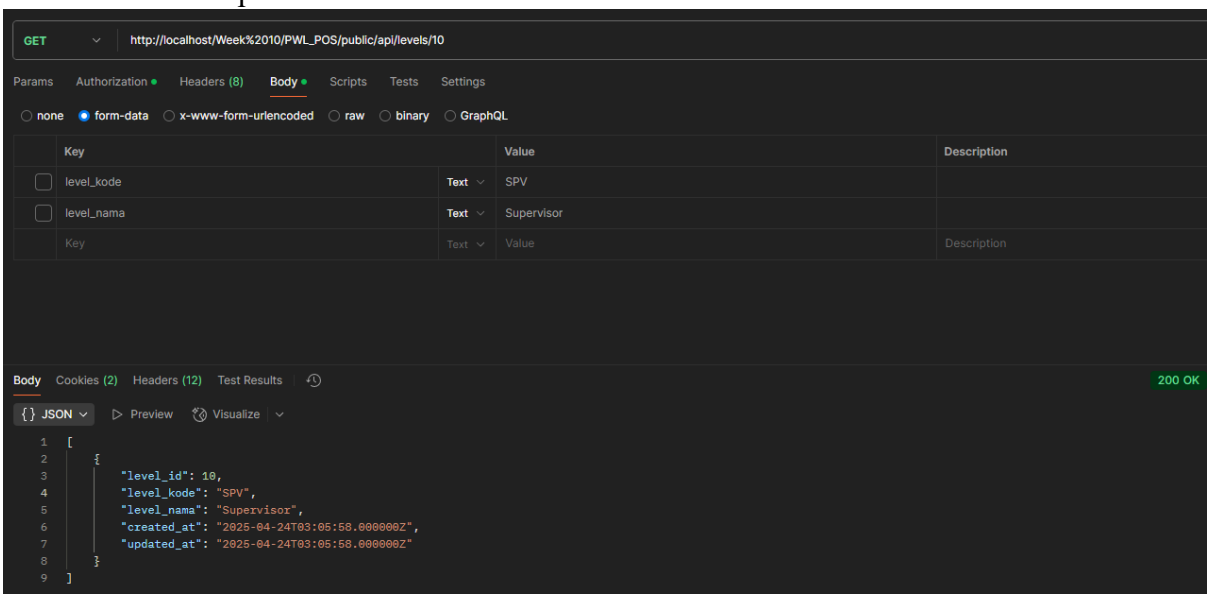
4. Uji coba API mulai dari fungsi untuk menampilkan data. Gunakan URL: localhost/PWL_POS-main/public/api/levels dan method GET



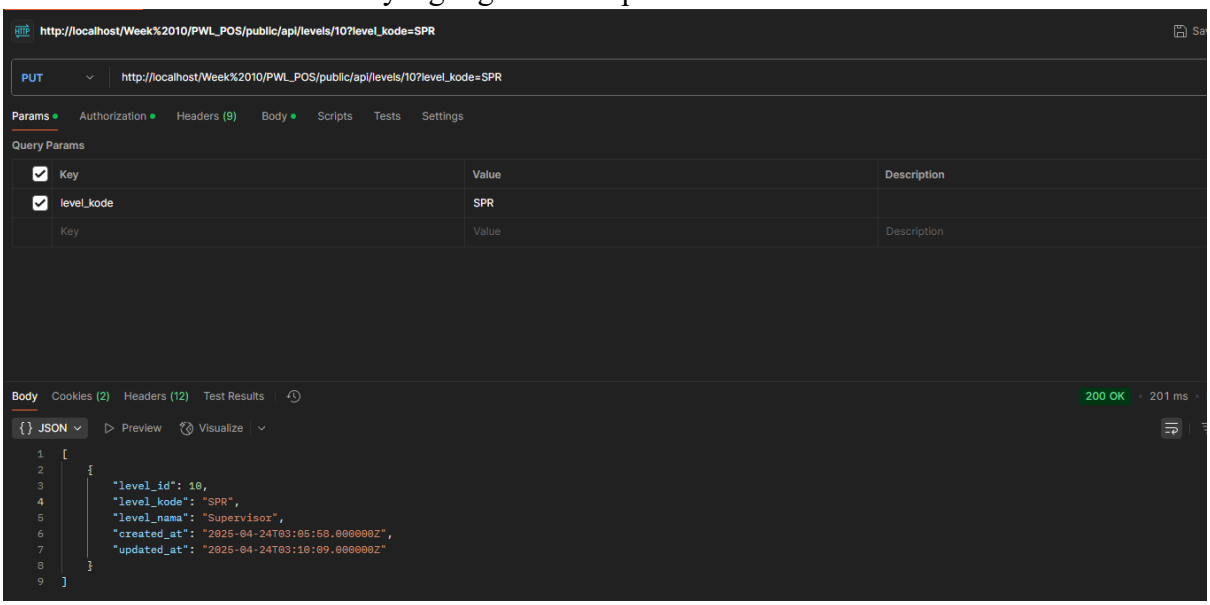
5. Percobaan penambahan data dengan URL :localhost/PWL_POSmain/public/api/levels dan method POST



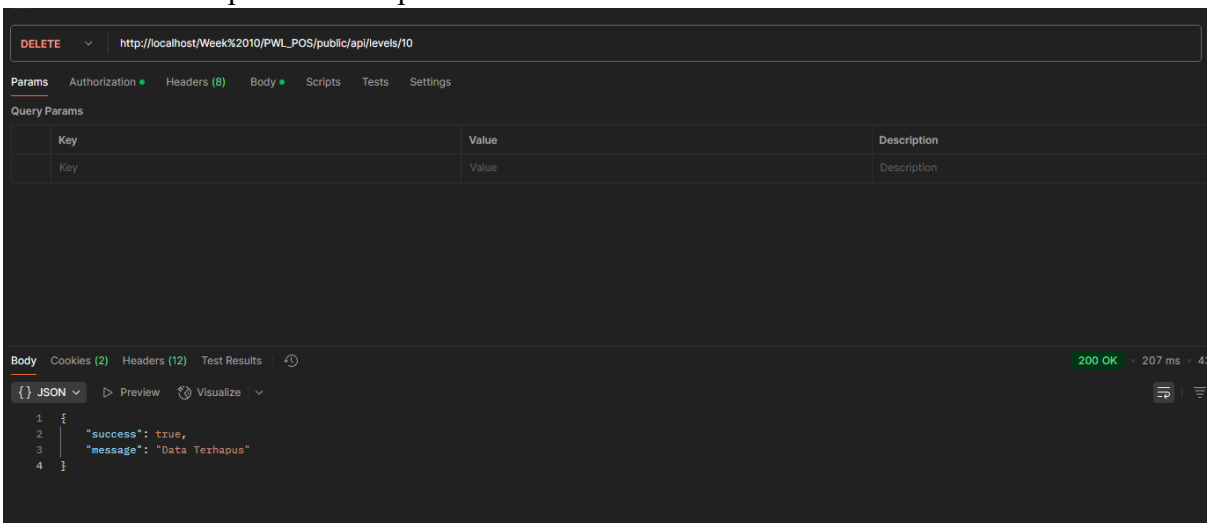
6. Percobaan menampilkan detail data.



7. 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.



9. Lakukan commit perubahan file pada Github
   https://github.com/fateehhh/PROWEBLNJT/commit/b8161618f800f51fa40068c80b4bcff569955cfb

TUGAS

1. Implementasikan CRUD API pada tabel lainnya yaitu tabel m_user, m_kategori, dan m_barang
   https://github.com/fateehhh/PROWEBLNJT/commit/ecc08bddb6388ab4b434e50790e6b426dfbecbc2