

PEMROGRAMAN WEB LANJUT

“RESTFUL API 2”



Kelas : TI-2H

Disusun Oleh :

Fanesabhirawaning Sulistyio

PROGRAM STUDI D-IV TEKNIK INFORMATIKA

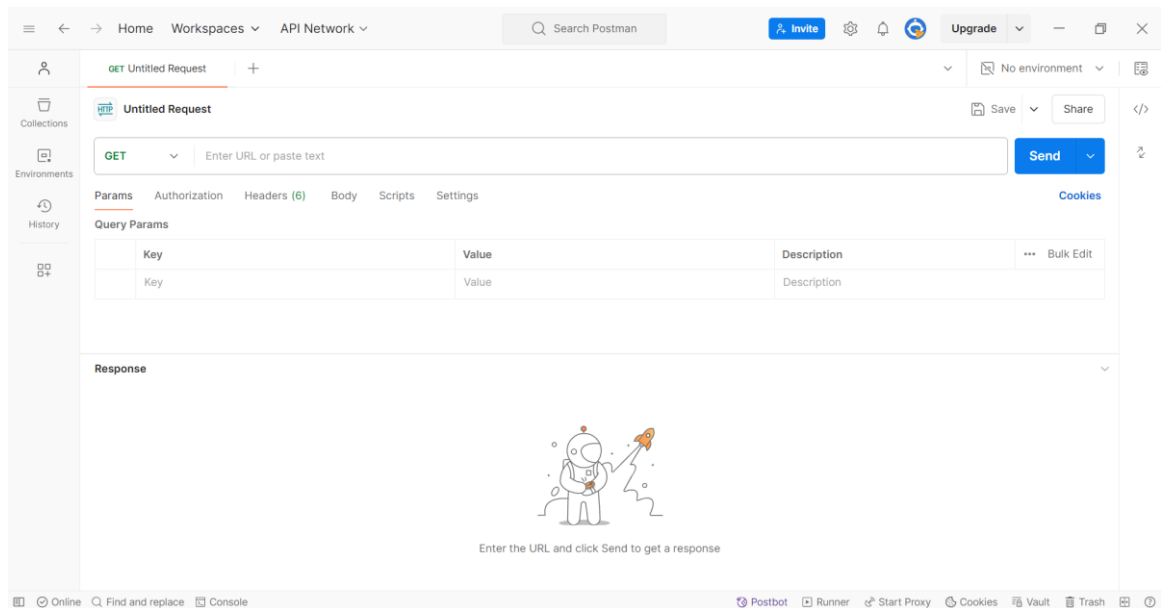
JURUSAN TEKNOLOGI INFORMASI

POLITEKNIK NEGERI MALANG

**Jl. Soekarno Hatta No.9, Jatimulyo, Kec. Lowokwaru, Kota Malang, Jawa Timur
65141**

Praktikum 1 – Implementasi Eloquent Accessor

1. Sebelum memulai pastikan REST API, terlebih dahulu pastikan sudah ter instal aplikasi Postman.



2. Kita akan memodifikasi Table m_user dengan menambahkan column : image, buka terminal lalu ketikkan
php artisan make:migration add_image_to_m_user_table

```
PS C:\laragon\www\PWL_POS> php artisan make:migration add_image_to_m_user_table
[INFO] Migration [C:\laragon\www\PWL_POS\database\Migrations\2024_05_14_064805_add_image_to_m_user_table.php] created successfully.
```

3. Buka file migrasi tersebut lalu modifikasi seperti ini lalu simpan:

```
database > migrations > 2024_05_14_064805_add_image_to_m_user_table.php > ...
1  <?php
2
3  use Illuminate\Database\Migrations\Migration;
4  use Illuminate\Database\Schema\Blueprint;
5  use Illuminate\Support\Facades\Schema;
6
7  return new class extends Migration
8  {
9      /**
10       * Run the migrations.
11       */
12     public function up(): void
13     {
14         Schema::table('m_user', function (Blueprint $table) {
15             $table->string('image');
16         });
17     }
18
19     /**
20      * Reverse the migrations.
21      */
22     public function down(): void
23     {
24         Schema::table('m_user', function (Blueprint $table) {
25             $table->dropColumn('image');
26         });
27     }
28 };
```

4. Lakukan jalankan update migrasi dengan cara:
php artisan migrate

```
PS C:\laragon\www\PWL_POS> php artisan migrate
INFO Running migrations.
2024_05_14_064805_add_image_to_m_user_table ..... 135ms DONE
```

5. Lalu lakukan modifikasi models pada App/Models/UserModel.php

```
app > Models > UserModel.php > ...
1  <?php
2
3  namespace App\Models;
4
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\HasOne;
9  use Illuminate\Foundation\Auth\User as Authenticatable;
10 use Tymon\JWTAuth\Contracts\JWTSubject;
11 use App\Models\LevelModel; // Add this import statement
12 use Illuminate\Database\Eloquent\Casts\Attribute;
13
14 class UserModel extends Authenticatable implements JWTSubject
15 {
16     use HasFactory;
17     public function getJWTIdentifier()
18     {
19         return $this->getKey();
20     }
21     public function getJWTCustomClaims()
22     {
23         return [];
24     }
25
26     protected $table = 'm_user';
27     public $timestamps = true;
28     protected $primaryKey = 'user_id';
29
30     protected $fillable = ['user_id', 'level_id', 'username', 'nama', 'password', 'image'];
31
32     public function level(): BelongsTo
33     {
34         return $this->belongsTo(LevelModel::class, 'level_id', 'level_id');
35     }
36     protected function image(): Attribute
37     {
38         return Attribute::make(
39             get: fn ($image) => url('/storage/posts/' . $image),
40         );
41     }
42 }
```

6. Lakukan modifikasi pada Controllers/Api/RegisterController

```
10 class RegisterController extends Controller
11 {
12     public function __invoke(Request $request)
13     {
14         //set validator
15         $validator = Validator::make($request->all(), [
16             'username' => 'required',
17             'nama' => 'required',
18             'password' => 'required|min:5|confirmed',
19             'level_id' => 'required',
20             'image' => 'required|image|mimes:jpeg,png,jpg,gif,svg|max:2048'
21         ]);
22
23         //if validator fails
24         if ($validator->fails()) {
25             return response()->json($validator->errors(), 422);
26         }
27
28         //create user
29         $image = $request->image;
30         $user = UserModel::create([
31             'username' => $request->username,
32             'nama' => $request->nama,
33             'password' => bcrypt($request->password),
34             'level_id' => $request->level_id,
35             'image' => $image->hashName(),
36         ]);
37     }
38 }
```

7. Anda dapat menambahkan detail untuk spesifikasi image pada validator

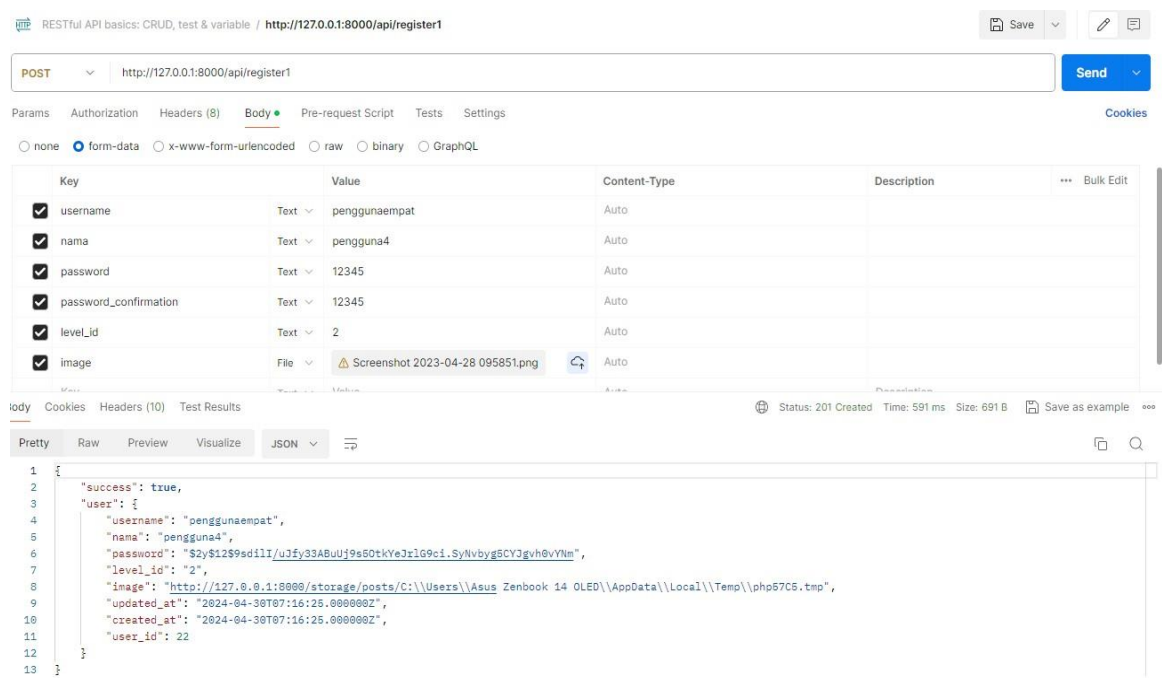
```
'image' => 'required|image|mimes:jpeg,png,jpg,gif,svg|max:2048'
```

8. Ubah atau tambahkan register1 pada routes/api.php

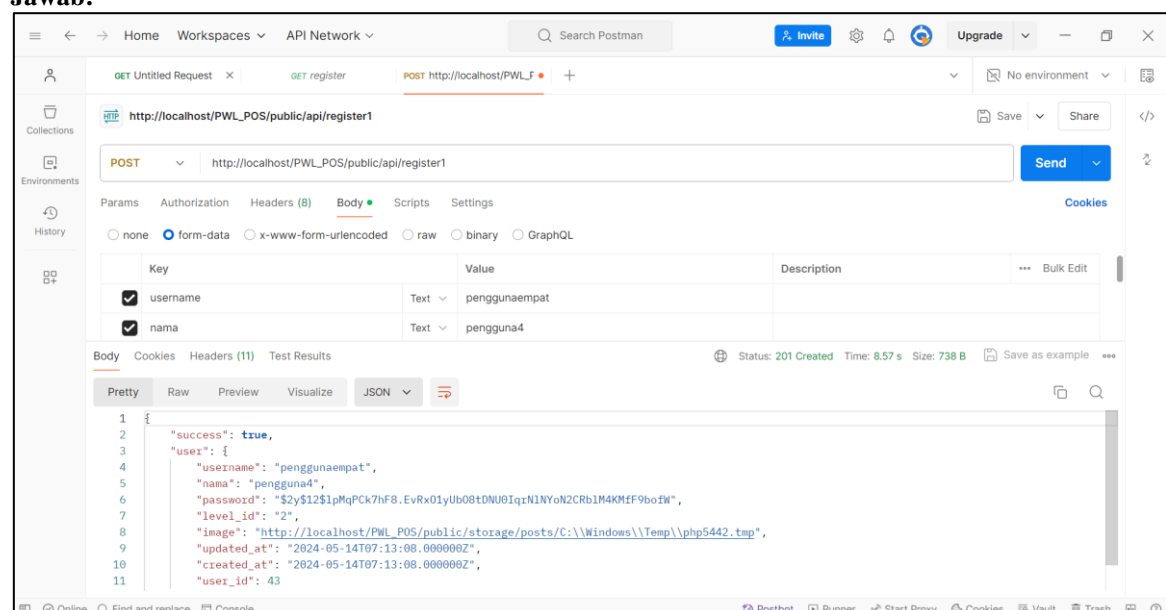
```
63 | Route::post('/register1', App\Http\Controllers\Api\RegisterController::class)->name('register1');
```

8. Simpan dan akses pada aplikasi Postman, atur pada Body isi manual Key dan Valuenya pada Key image tambahkan value File dan upload gambar

<http://127.0.0.1:8000/api/register1> dengan method POST dan klik send



Jawab:

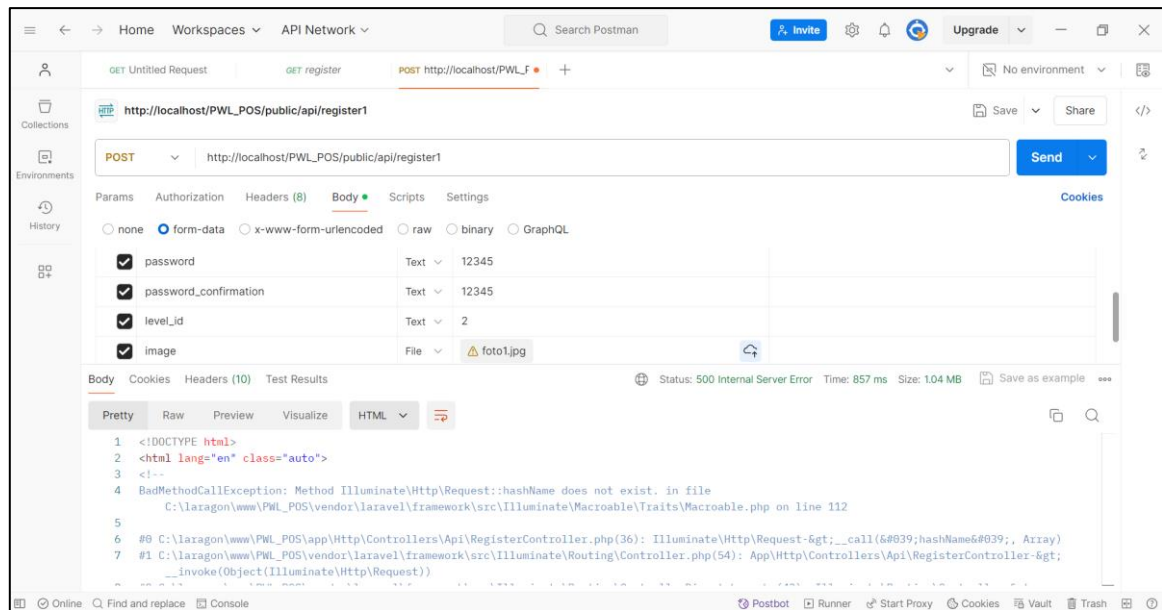


9. Pada Controllers/Api/RegisterController bagian create user ganti dengan

```
'image' => $image->hashName(),|
```

10. Uji coba dan screenshot hasilnya apa perbedaan dari yang sebelumnya

Jawab:



Jawab:

File yang diinputkan akan gagal diproses dan menyebabkan terjadinya eroro dan data yang diproses tidak disimpan. Dikarenakan penggantian

'image' => \$request->image **menjadi** 'image' => \$request->hashName(),

TUGAS

Implementasikan API untuk upload file/gambar pada tabel lainnya yaitu tabel m_barang dan gunakan pada transaksi. Uji coba dengan method GET untuk memanggil data yang sudah di inputkan.

Jawab:

- Kita akan memodifikasi Table m_barang dengan menambahkan column : image, buka terminal lalu ketikkan
php artisan make:migration add_image_to_m_barang_table

```
PS C:\laragon\www\PWL_POS> php artisan make:migration add_image_to_m_barang_table
INFO Migration [C:\laragon\www\PWL_POS\database\Migrations\2024_05_14_072104_add_image_to_m_barang_table.php] created successfully.
```

- Buka file migrasi tersebut lalu modifikasi seperti ini lalu simpan:

```
2024_05_14_072104_add_image_to_m_barang_table.php U X
database > migrations > 2024_05_14_072104_add_image_to_m_barang_table.php > class > down
1  <?php
2
3  use Illuminate\Database\Migrations\Migration;
4  use Illuminate\Database\Schema\Blueprint;
5  use Illuminate\Support\Facades\Schema;
6
7  return new class extends Migration
8  {
9      /**
10       * Run the migrations.
11       */
12     public function up(): void
13     {
14         Schema::table('m_barang', function (Blueprint $table) {
15             $table->string('image');
16         });
17     }
18
19     /**
20      * Reverse the migrations.
21      */
22     public function down(): void
23     {
24         Schema::table('m_barang', function (Blueprint $table) {
25             $table->dropColumn('image');
26         });
27     }
28 };
```

- Lakukan jalankan update migrasi dengan cara:

php artisan migrate

```
PS C:\laragon\www\PWL_POS> php artisan migrate

INFO Running migrations.

2024_05_14_072104_add_image_to_m_barang_table ..... 109ms DONE
```

- Lalu lakukan modifikasi models pada App/Models/BarangModel.php

```
app > Models > BarangModel.php > image

1  <?php
2
3  namespace App\Models;
4
5  use Illuminate\Database\Eloquent\Casts\Attribute;
6  use Illuminate\Database\Eloquent\Factories\HasFactory;
7  use Illuminate\Database\Eloquent\Model;
8  use Illuminate\Database\Eloquent\Relations\BelongsTo;
9  use Illuminate\Database\Eloquent\Relations\HasOne;
10
11 class BarangModel extends Model
12 {
13     use HasFactory;
14     protected $table = 'm_barang';
15     protected $primaryKey = 'barang_id';
16     protected $fillable = [
17         'barang_kode',
18         'barang_nama',
19         'harga_beli',
20         'harga_jual',
21         'kategori_id',
22         'image',
23     ];
24
25     // Pastikan tipe data stok adalah integer
26     protected $casts = [
27         'stok' => 'integer',
28     ];
29
30     public function kategori(): BelongsTo
31     {
32         return $this->belongsTo(KategoriModel::class, 'kategori_id', 'kategori_id');
33     }
34
35     public function stok(): HasOne
36     {
37         return $this->hasOne(StokModel::class, 'barang_id', 'barang_id');
38     }
39     public function image(): Attribute
40     {
41         return Attribute::make(
42             get: fn ($image) => url('/storage/posts/' . $image),
43         );
44     }
45 }
```


- Lakukan modifikasi pada Controllers/Api/BarangController

```

app > Http > Controllers > Api > BarangController.php > BarangController > update
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  use Illuminate\Support\Facades\Validator;
9
10 class BarangController extends Controller
11 {
12     public function index()
13     {
14         return BarangModel::all();
15     }
16
17     public function store(Request $request)
18     {
19         //set validator
20         $validator = Validator::make($request->all(), [
21             'barang_kode' => 'required',
22             'barang_nama' => 'required',
23             'harga_beli' => 'required',
24             'harga_jual' => 'required',
25             'kategori_id' => 'required',
26             'image' => 'required|image|mimes:jpeg,png,jpg,gif,svg|max:2048'
27         ]);
28         //if validations fails
29         if ($validator->fails()) {
30             return response()->json($validator->errors(), 422);
31         }
32         $barang = BarangModel::create($request->all());
33         if ($barang) {
34             return response()->json([
35                 'success' => true,
36                 'barang' => $barang,
37             ], 201);
38         }
39         //return JSON process insert failed
40         return response()->json([
41             'success' => false,
42         ], 409);
43     }
44
45     public function show(BarangModel $barang)
46     {
47         return BarangModel::find($barang);
48     }
49
50     public function update(Request $request, BarangModel $barang)
51     {
52         $validator = Validator::make($request->all(), [
53             'barang_kode' => 'required',
54             'barang_nama' => 'required',
55             'harga_beli' => 'required',
56             'harga_jual' => 'required',
57             'kategori_id' => 'required',
58             'image' => 'required|image|mimes:jpeg,png,jpg,gif,svg|max:2048'
59         ]);
60         //if validations fails
61         if ($validator->fails()) {
62             return response()->json($validator->errors(), 422);
63         }
64         BarangModel::find($barang)->update($request->all(), [
65             $request->image->hashName()
66         ]);
67         $barang = BarangModel::with('kategori')->find($barang);
68         if ($barang) {
69             return response()->json([
70                 'success' => true,
71                 'barang' => $barang,
72             ], 201);
73         }

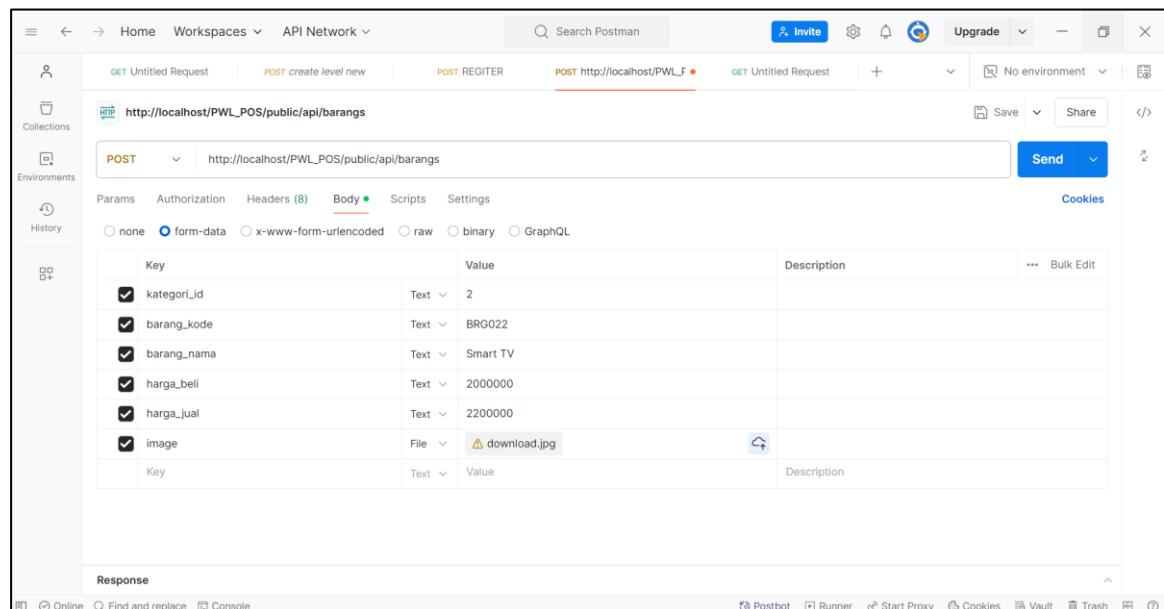
```

```

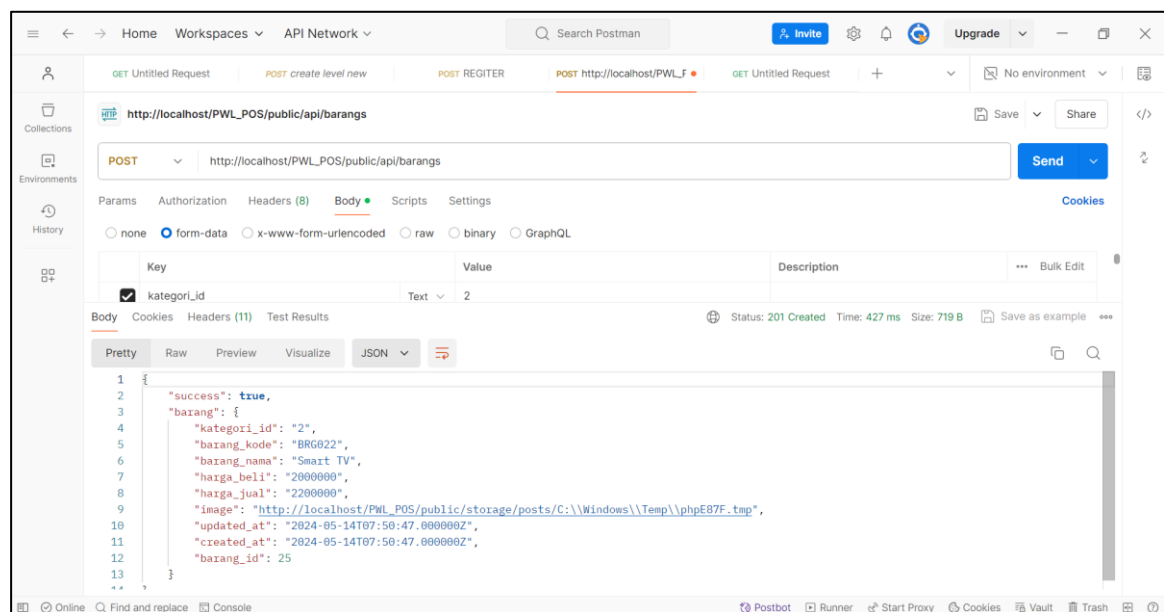
74 //return JSON process insert failed
75 return response()->json([
76     'success' => false,
77 ], 409);
78
79
80 public function destroy(BarangModel $barang)
81 {
82     $barang->delete();
83     return response()->json([
84         "success" => true,
85         "message" => "Data terhapus"
86     ]);
87 }
88 }

```

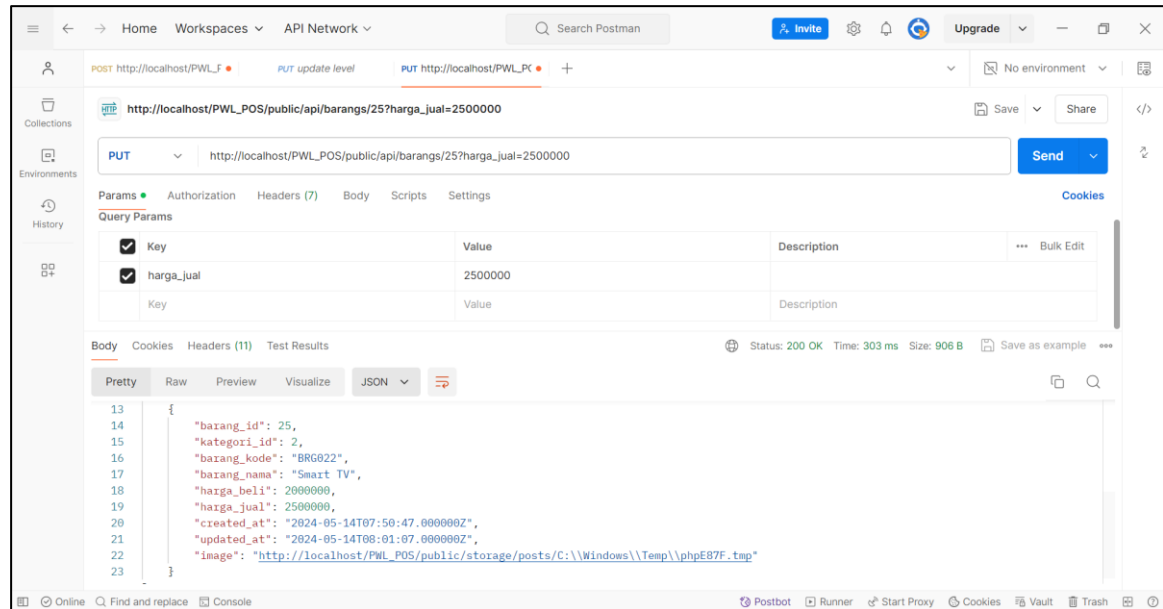
- Create Barang



- Hasil keterangan create barangs sukses atau tidak



- Update data barang



*** Sekian, dan Terima Kasih***