



Mata Kuliah : Pemrograman Web Lanjut (PWL)  
Program Studi : D4 – Teknik Informatika / D4 – Sistem Informasi Bisnis  
Semester : 4 (empat) / 6 (enam)  
Pertemuan ke- : 11 (sebelas)  
Nama : Muhammad Fairuz Daffa Athallah  
Kelas : SIB 2B / 15  
NIM : 2341760079

## **JOBSHEET 11**

### **RESTFUL API 2**

Sebelumnya kita sudah membahas mengenai *RESTFUL API dan JSON Web Token (JWT)* pada Laravel. Dimana kita telah membuat RESTful API Register, RESTful API Login, RESTful API Logout, dan implementasi CRUD dalam RESTful API pada halaman web. Pada pertemuan kali ini, kita akan mempelajari penerapan RESTFUL API lanjutan 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. ELOQUENT ACCESSOR**

Laravel memiliki fitur yang bernama mutator, accessor dan casting, fitur-fitur ini digunakan untuk melakukan manipulasi data di dalam attribute database dengan sangat mudah. Contohnya pada saat insert data dengan enkripsi ke dalam database dan melakukan deskripsi saat menampilkan dari database secara otomatis.

Accessor dapat mengubah nilai saat attribute atau field eloquent diakses. Untuk mendefinisikan Accessor dapat membuat method di dalam model untuk menentukan attribute yang akan diakses. Nama method yang dibuat harus sama dengan nama attribute yang akan di format: contoh :

Attribute/field pada tabel `first_name` maka methodnya `firstName()`





```
<?php

use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

return new class extends Migration
{
    /**
     * Run the migrations.
     */
    public function up(): void
    {
        Schema::table('m_user', function (Blueprint $table) {
            $table->string('image');
        });
    }

    /**
     * Reverse the migrations.
     */
    public function down(): void
    {
        Schema::table('m_user', function (Blueprint $table) {
            $table->dropColumn('image');
        });
    }
};
```

Migration



```
UserModel.php M 2025_04_24_112419_add_image_to_m_user_table.php U X
Minggu_11 > PWL_POS > database > migrations > 2025_04_24_112419_add_image_to_m_
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 }
29 };
```

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

```
PS E:\Semester3\Pemrograman_Web\laragon\www\PWL_2025_Daffa\Minggu_11\PWL_POS> php artisan migrate

Warning: PHP Startup: Unable to load dynamic library 'php_sqlsrv.dll' (tried: E:\Semester3\Pemrograman_Web\laragon\bin\php\php-8.3.17-Win32-vs16-x64\mester3\Pemrograman_Web\laragon\bin\php\php-8.3.17-Win32-vs16-x64\ext\php_php_sqlsrv.dll.dll (The specified module could not be found)) in Unknown on
Warning: PHP Startup: Unable to load dynamic library 'php_pdo_sqlsrv.dll' (tried: E:\Semester3\Pemrograman_Web\laragon\bin\php\php-8.3.17-Win32-vs16-x64\ext\php_php_pdo_sqlsrv.dll.dll (The specified module could not be found)) i

[INFO] Running migrations.

2025_04_24_112419_add_image_to_m_user_table ..... 418ms DONE
```

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

```
<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Model;
use Tymon\JWTAuth\Contracts\JWTSubject;
use Illuminate\Database\Eloquent\Casts\Attribute;
use Illuminate\Foundation\Auth\User as Authenticatable;
```



```
class UserModel extends Authenticatable implements JWTSubject
{

    public function getJWTIdentifier(){
        return $this->getKey();
    }

    public function getJWTCustomClaims(){
        return [];
    }

    protected $table = 'm_user';
    protected $primaryKey = 'user_id';

    protected $fillable = [
        'username',
        'nama',
        'password',
        'level_id',
        'image'//tambahan
    ];

    public function level()
    {
        return $this->belongsTo(LevelModel::class, 'level_id',
'level_id');
    }
    protected function image(): Attribute
    {
        return Attribute::make(
            get: fn ($image) => url('/storage/posts/' . $image),
        );
    }
}
```

UserModel



```

UserModel.php M x
Minggu_11 > PNL_POS > app > Models > UserModel.php > UserModel
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\Foundation\Auth\User as Authenticatable;
10 use Tymon\JWTAuth\Contracts\JWTSubject;
11
12
13 class UserModel extends Authenticatable implements JWTSubject
14 {
15
16     public function getJWTIdentifier()
17     {
18         return $this->getKey();
19     }
20
21     public function getJWTCustomClaims()
22     {
23         return [];
24     }
25
26     use HasFactory;
27
28     protected $table = 'm_user';
29     protected $primaryKey = 'user_id'; // sendefinisi primary key dari tabel
30
31     protected $fillable=['level_id','username','nama','password','created_at','updated_at','profile_picture','image'];
32
33     protected $hidden = ['password']; //jangan di tampilkan saat select
34     protected $casts = ['password' => 'hashed']; //casting password agar otomatis di hash
35
36     public function image(){
37         return Attribute::make(
38             get: fn ($image) => url('/storage/post/' . $image),
39         );
40     }
41
42     public function level(): BelongsTo
43     {
44         return $this->belongsTo(LevelModel::class, 'level_id', 'level_id');
45     }
46
47     public function getRoleName(): string{
48         return $this->level->level_nama;
49     }
50
51     public function hasRole($role): bool{
52         return $this->level->level_kode == $role;
53     }
54
55     public function getRole() {
56         return $this->level->level_kode;
57     }
58 }
```

6. Lakukan modifikasi pada Controllers/Api/RegisterController

```

<?php

namespace App\Http\Controllers\Api;

use App\Models\UserModel;
use App\Http\Controllers\Controller;
use Illuminate\Http\Request;
```



```
use Illuminate\Support\Facades\Validator;

class RegisterController extends Controller
{
    public function __invoke(Request $request)
    {
        //set validation
        $validator = Validator::make($request->all(), [
            'username' => 'required',
            'nama' => 'required',
            'password' => 'required|min:5|confirmed',
            'level_id' => 'required',
            'image' => 'required'

        ]);

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

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

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

        //return JSON process insert failed
        return response()->json([
            'success' => false,
        ], 409);
    }
}
```





## RegisterController

```
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         $validator = Validator::make($request->all(), [
15             'username' => 'required',
16             'nama' => 'required',
17             'password' => 'required|min:5|confirmed',
18             'level_id' => 'required',
19             'image' => 'required',
20         ]);
21
22         if ($validator->fails()) {
23             return response()->json($validator->errors(), 422);
24         }
25
26         $user = UserModel::create([
27             'username' => $request->username,
28             'nama' => $request->nama,
29             'password' => bcrypt($request->password),
30             'level_id' => $request->level_id,
31             'image' => $request->image,
32         ]);
33
34         if ($user) {
35             return response()->json([
36                 'success' => true,
37                 'user' => $user,
38             ], 201);
39         }
40
41         return response()->json([
42             'success' => false,
43         ], 409);
44     }
45 }
```

7. Anda dapat menambahkan detail untuk spesifikasi image pada validator

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

```
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             'image' => 'required|image|mimes:jpeg,png,jpg,gif,svg|max:2048',
20         ]);
```





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

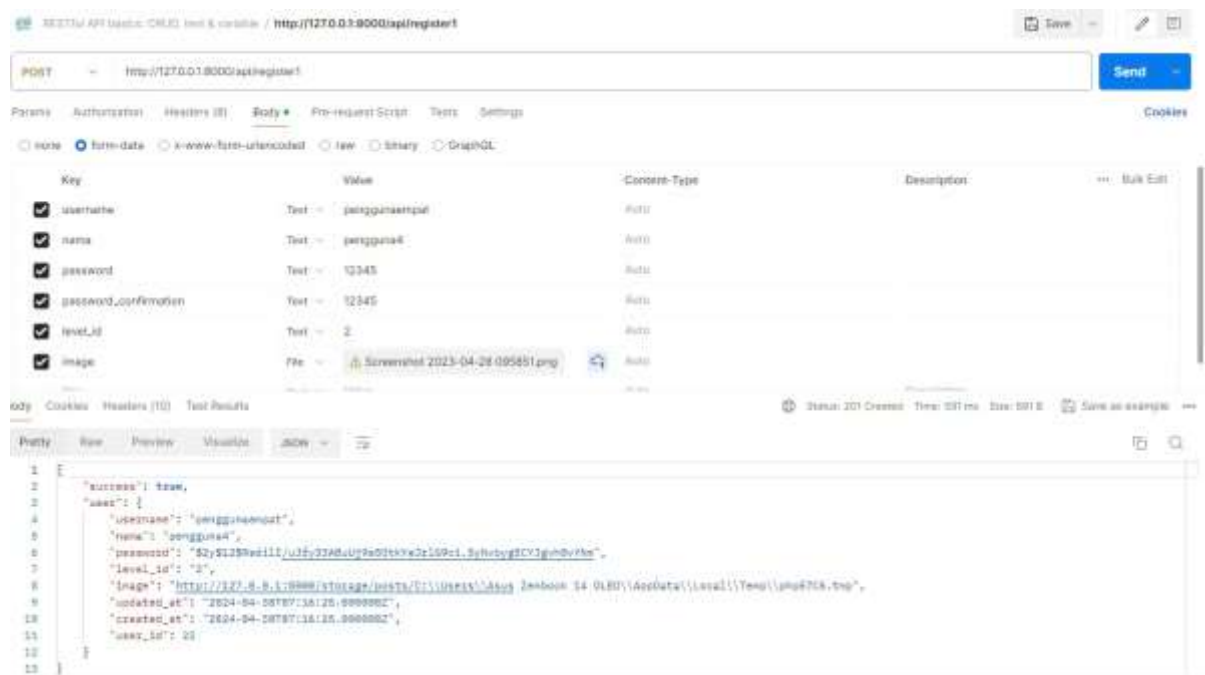
```
Route::post('/register1', App\Http\Controllers\Api\RegisterController::class)->name('register1');
```

Api.php

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

9. 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



Postman



POST http://127.0.0.1:8000/a. + No environment

HTTP http://127.0.0.1:8000/api/register1 Save

POST http://127.0.0.1:8000/api/register1 Send

Params Authorization Headers (9) 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"/> username	Text okee			
<input checked="" type="checkbox"/> nama	Text okee			
<input checked="" type="checkbox"/> password	Text 123456			
<input checked="" type="checkbox"/> level_id	Text 1			
<input checked="" type="checkbox"/> image	File WhatsApp Image 2025-04-...			

Body Cookies Headers (9) Test Results 201 Created 2.28 s 549 B Save as example

Pretty Raw Preview Visualize JSON

```
1 {
2   "success": true,
3   "user": {
4     "username": "okee",
5     "nama": "okee",
6     "level_id": "1",
7     "image": "http://127.0.0.1:8000/storage/post/C:\\Users\\ADMIN\\AppData\\Local\\Temp\\phpAAFC.tmp",
8     "updated_at": "2025-04-24T12:05:10.000000Z",
9     "created_at": "2025-04-24T12:05:10.000000Z",
10    "user_id": 49
11  }
```

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

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

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



POST http://127.0.0.1:8000/a, + No environment

HTTP http://127.0.0.1:8000/api/register1 Save

POST http://127.0.0.1:8000/api/register1 Send

Params Authorization Headers (9) 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"/> username	Text satudua			
<input checked="" type="checkbox"/> nama	Text satudua			
<input checked="" type="checkbox"/> password	Text 123456			
<input checked="" type="checkbox"/> level_id	Text 1			
<input checked="" type="checkbox"/> image	File WhatsApp Image 2025-04-...			

Body Cookies Headers (9) Test Results 201 Created 1325 ms 548 B Save as example

Pretty Raw Preview Visualize JSON

```
1 {
2   "success": true,
3   "user": {
4     "username": "satudua",
5     "nama": "satudua",
6     "level_id": "1",
7     "image": "http://127.0.0.1:8000/storage/post/PZn0t6LRs4fxuvhMmYkbjAV4CmVWCShC12piTKF.png",
8     "updated_at": "2025-04-24T12:14:10.000000Z",
9     "created_at": "2025-04-24T12:14:10.000000Z",
10    "user_id": 50
  }
```

Alamat url dari image akan di enkripsi

## 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.



## BarangModel

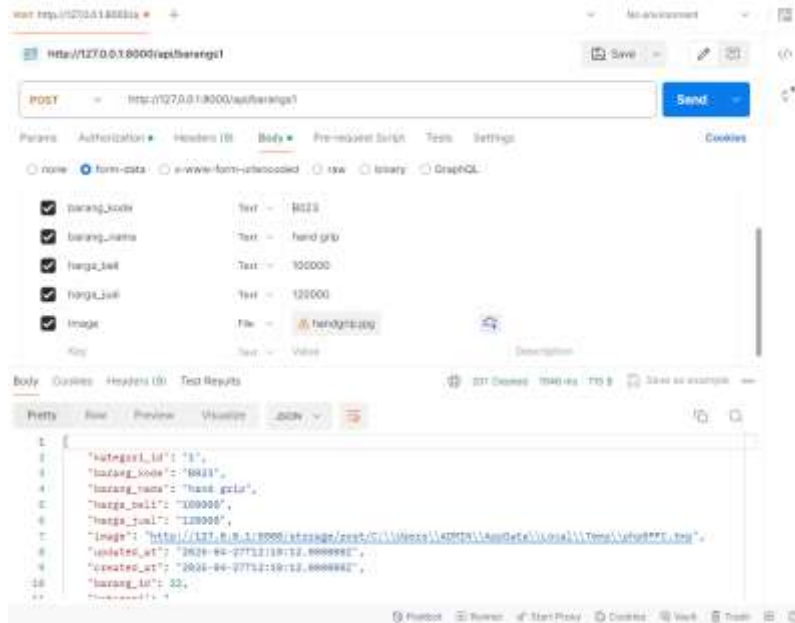
```
TransaksiController.php M 2025.04.27.022334 add image to m_barang table.php U BarangController.php M api.php M BarangModel.php M X
Minggu, 11 > PwL_POS > app > Models > BarangModel.php > BarangModel > image
3 namespace App\Models;
4
5 use Illuminate\Database\Eloquent\Factories\HasFactory;
6 use Illuminate\Database\Eloquent\Model;
7 use Illuminate\Database\Eloquent\Relations\BelongsTo;
8 use Illuminate\Database\Eloquent\casts\Attribute;
9
10 class BarangModel extends Model
11 {
12     use HasFactory;
13
14     protected $table = 'm_barang';
15     protected $primaryKey = 'barang_id';
16     protected $fillable = [
17         'kategori_id',
18         'barang_kode',
19         'barang_nama',
20         'harga_beli',
21         'harga_jual',
22         'image',
23     ];
24
25     public function image(): Attribute
26     {
27         return Attribute::make(
28             get: fn ($image) => url('/storage/post/'. $image),
29         );
30     }
31
32     public function kategori(): BelongsTo
33     {
34         return $this->belongsTo(KategoriModel::class, 'kategori_id', 'kategori_id');
35     }
36
37     public function stok()
38     {
39         return $this->hasOne(StokModel::class, 'barang_id', 'barang_id');
40     }
41 }
```

## BarangController

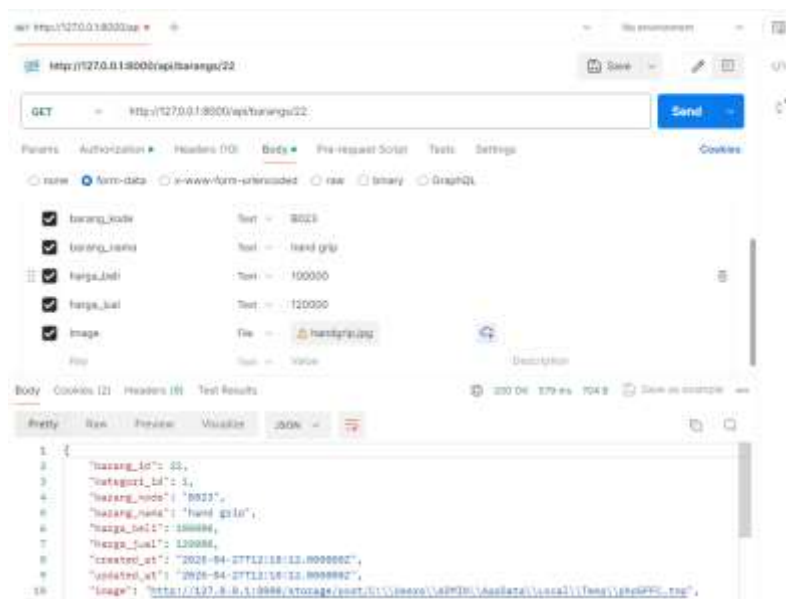
```
TransaksiController.php M 2025.04.27.022334 add image to m_barang table.php U BarangController.php M
Minggu, 11 > PwL_POS > app > Http > Controllers > Api > BarangController.php > BarangController > _invoke
1 <?php
2
3 namespace App\Http\Controllers\Api;
4
5 use App\Http\Controllers\Controller;
6 use App\Models\BarangModel;
7 use Illuminate\Http\Request;
8 use Illuminate\Support\Facades\Validator;
9
10 class BarangController 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',
18             'level_id' => 'required',
19             'image' => 'required|image|mimes:jpeg,png,jpg,gif,svg|max:2048',
20         ]);
21
22         if ($validator->fails()) {
23             return response()->json($validator->errors(), 422);
24         }
25
26         $user = BarangModel::create([
27             'username' => $request->username,
28             'nama' => $request->nama,
29             'password' => bcrypt($request->password),
30             'level_id' => $request->level_id,
31             'image' => $request->image,
32         ]);
33
34         if ($user) {
35             return response()->json([
36                 'success' => true,
37                 'user' => $user,
38             ], 201);
39         }
40
41         return response()->json([
42             'success' => false,
43         ], 400);
44     }
45
46     public function index()
47     {
48         return BarangModel::with('kategori')->get();
49     }
50
51     public function store(Request $request)
52     {
53         $barang = BarangModel::create($request->all());
54         return response()->json($barang->load('kategori'), 201);
55     }
56
57     public function show(BarangModel $barang)
58     {
59         return $barang->load('kategori');
60     }
61 }
```



## CreateData



## GetData





## Transaksi

### DetailPenjualanModel

```
Minggu, 11 > PWA_PC2 > app > Models > PenjualanDetailModel.php > PenjualanDetailModel > Image
1: <?php
2:
3: namespace App\Models;
4:
5: use Illuminate\Database\Eloquent\Concerns\HasAttributes;
6: use Illuminate\Database\Eloquent\Factories\HasFactory;
7: use Illuminate\Database\Eloquent\Model;
8: use Illuminate\Database\Eloquent\Relations\BelongsTo;
9:
10: class PenjualanDetailModel extends Model
11: {
12:     use HasFactory;
13:
14:     protected $table = 't_penjualan_detail';
15:     protected $primaryKey = 'detail_id';
16:     protected $fillable = [
17:         'penjualan_id',
18:         'barang_id',
19:         'harga',
20:         'jumlah',
21:         'image'
22:     ];
23:
24:     public function image():Attribute{
25:         return Attribute::make([
26:             get: fn ($image) => url('/storage/post/'. $image),
27:         ]);
28:     }
29:
30:     // Relasi ke header transaksi
31:     public function sales(): BelongsTo
32:     {
33:         return $this->belongsTo(PenjualanModel::class, 'penjualan_id', 'penjualan_id');
34:     }
35:
36:     // Relasi ke data barang
37:     public function barang(): BelongsTo
38:     {
39:         return $this->belongsTo(BarangModel::class, 'barang_id', 'barang_id');
40:     }
41: }
```

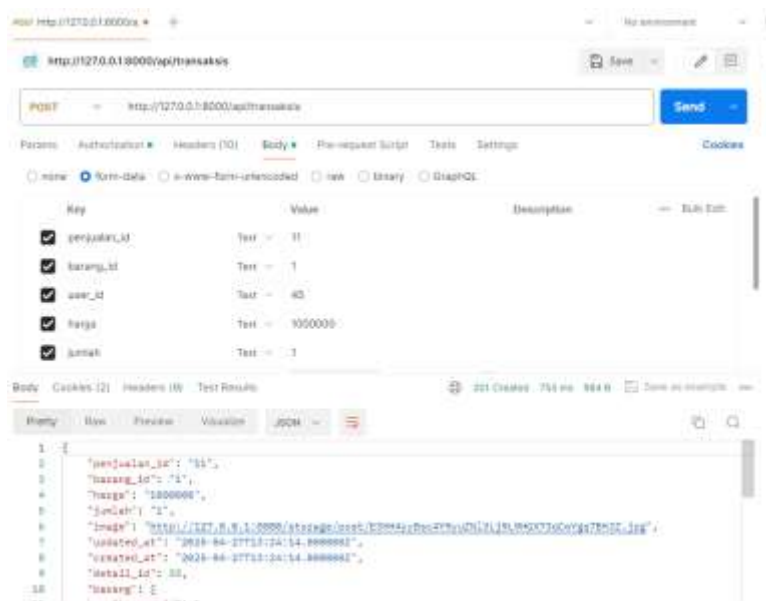
### TransaksiController





```
Minggu, 11 > PWL_POS > app > http > Controllers > Api > TransaksiController.php > TransaksiController > store
1  <?php
2
3  namespace App\Http\Controllers\Api;
4
5  use App\Http\Controllers\Controller;
6  use App\Models\PengjualanDetailModel;
7  use Illuminate\Http\Request;
8  use App\Models\PengjualanModel;
9  use Illuminate\Support\Facades\Storage;
10 use Illuminate\Support\Facades\Validator;
11
12 class TransaksiController extends Controller
13 {
14     public function index()
15     {
16         $data = PengjualanDetailModel::with(['sales', 'barang'])->get();
17         return response()->json($data);
18     }
19
20     public function show($transaksi)
21     {
22         $pengjualan = PengjualanDetailModel::with(['sales', 'barang'])->findOrFail($transaksi);
23         return response()->json($pengjualan);
24     }
25
26     /** POST /api/transaksi */
27     public function store(Request $request)
28     {
29         $validated = $request->validate([
30             'pengjualan_id' => 'required|exists:t_pengjualan,pengjualan_id',
31             'barang_id' => 'required|exists:m_barang,barang_id',
32             'harga' => 'required|numeric',
33             'jumlah' => 'required|numeric',
34             'image' => 'nullable|image|mimes:jpeg,png,jpg,gif,svg|max:2048',
35         ]);
36
37         if ($request->hasFile('image')) {
38             $validated['image'] = basename($request->file('image')->store('post', 'public'));
39         }
40
41         $pengjualanDetail = PengjualanDetailModel::create($validated);
42
43         return response()->json($pengjualanDetail->load('barang', 'sales'), 201);
44     }
45
46     /** GET /api/transaksi/{transaksi} */
47     // public function show($transaksi)
48     // {
49     //     $trx = PengjualanDetailModel::findOrFail($transaksi);
50     //     return response()->json($trx);
51     // }
52
53     /** PUT /api/transaksi/{transaksi} */
54     public function update(Request $request, $transaksi)
55     {
56         $trx = PengjualanDetailModel::findOrFail($transaksi);
57
58         $v = Validator::make($request->all(), [
```

## StoreData





## GetData

GET http://127.0.0.1:8000/api/transaksi/33

Params Authorization Headers (10) Body Pre-request Script Tests Settings Cookies

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

Key	Value	Description
<input checked="" type="checkbox"/> penjualan_id	Text 11	
<input checked="" type="checkbox"/> barang_id	Text 1	
<input checked="" type="checkbox"/> user_id	Text 45	
<input checked="" type="checkbox"/> harga	Text 1000000	
<input checked="" type="checkbox"/> jumlah	Text 1	

Body Cookies (2) Headers (9) Test Results 200 OK 886 ms 871 B Save as example

Pretty Raw Preview Visualize JSON

```
1 {
2   "detail_id": 33,
3   "penjualan_id": 11,
4   "barang_id": 1,
5   "harga": 1000000,
6   "jumlah": 1,
7   "created_at": "2025-04-27T13:14:14.889000Z",
8   "updated_at": "2025-04-27T13:14:14.889000Z",
9   "image": "http://127.0.0.1:8000/storage/post/63M4yyBsc4Y9yUzN13Lj9L9HGK7SoCoVgq7BNJZ_img",
10  "sales": [
11  ]
12 }
```

\*\*\* Sekian, dan selamat belajar \*\*\*