



Nama/ Nim : **Arimbi Putri Hapsari /2341760016**  
Mata Kuliah : Pemrograman Web Lanjut (PWL)  
Program Studi : D4 – Teknik Informatika / D4 – Sistem Informasi Bisnis  
Semester : 4 (empat) / 6 (enam)  
Pertemuan ke- : 11 (sebelas)

## 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()`

protected function `firstName()`: Attribute

```
{ //...
```



}

Jika membuat attribute/field image yang ada di table m\_user kita akan memberikan nilai full path dari direktori dimana file gambar tersebut disimpan. Contohnya pada UserModel ditambahkan

```
protected function image(): Attribute
{
    return Attribute::make(
        get: fn ($image) => url('/storage/posts/' . $image),
    );
}
```

Dengan begitu dapat melakukan import Eloquent Attribute dengan

```
use Illuminate\Database\Eloquent\Casts\Attribute;
```

Lalu method baru dengan nama image() melakukan return path nama file image itu berada

```
get: fn ($image) => url('/storage/posts/' . $image),
```

Hasil akhir memanggil attribute image

```
domain.com/storage/posts/nama_file_image.png
```

## 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 D:\laragon\www\PWL_25\week11\jobsheet\PWL_POS> php artisan make:migration add_image_to_m_user_table

INFO Migration [D:\laragon\www\PWL_25\week11\jobsheet\PWL_POS\database\Migrations\2025_05_03_050412_add_image_to_m_user_table.php] created successfully.
```

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



```
<?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');
        });
    }
};
```

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

```
eet\PWL_POS> php artisan migrate
```

```
INFO Running migrations.
```

```
2025_05_03_050412_add_image_to_m_user_table 81ms DONE
```



Showing rows 0 - 7 (8 total, Query took 0.0007 seconds.)

SELECT \* FROM `m\_user`

☐ Profiling [ Edit inline ] [ Edit ] [ Explain SQL ] [ Create PHP code ] [ Refresh ]

☐ Show all | Number of rows: 25 | Filter rows: Search this table | Sort by key: None

Extra options

	user_id	level_id	username	nama	password	user_profile_picture	created_at	updated_at
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	1	1	admin	Administrator	\$2y\$12\$3CL2Z3HpHbFyKfKulmf qAJKE6f9oEoGmszrmogLy...	profiles/profile_1_1744096681.jpg	NULL	2025-04-08 07:18:01

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

```
<?php
namespace App\Models;
use Illuminate\Database\Eloquent\Model; use
Tyron\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),
        );
    }
}
```

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;
```



KEMENTERIAN PENDIDIKAN, KEBUDAYAAN, RISET, DAN TEKNOLOGI  
**POLITEKNIK NEGERI MALANG**  
**JURUSAN TEKNOLOGI INFORMASI**  
Jl. Soekarno Hatta No. 9, Jatimulyo, Lowokwaru, Malang 65141  
Telp. (0341) 404424 – 404425, Fax (0341) 404420  
<http://www.polinema.ac.id>

---



```
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);
    }
}
```



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

```
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

127.0.0.1:8000/api/register1

POST 127.0.0.1:8000/api/register1

Params Authorization Headers (8) Body Scripts Tests Settings Cookies

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

Key	Value	Description
username	penggunaempat	
nama	pengguna4	
password	12345	
password_confirmation	12345	
level_id	2	
image	kucing.jpg	

Body Cookies Headers (10) Test Results 201 Created 549 ms 594 B

```
{
  "success": true,
  "user": {
    "username": "penggunaempat",
    "nama": "pengguna4",
    "level_id": "2",
    "image": "http://127.0.0.1:8000/storage/posts/fF3ebKNFDhGYe6UVWNBbKTHAapY3baVpE9HKm9S5.jpg",
    "updated_at": "2025-05-03T05:19:22.000000Z",
    "created_at": "2025-05-03T05:19:22.000000Z",
    "user_id": 32
  }
}
```

Data user tersimpan di tabel m\_user berupa path tanpa di hash

32	2	penggunaempat	pengguna4	\$2y\$12\$P4sQYRQI05m0sH0kPv7n4uXzEgzhDVASb9961zt1n.qU...	NULL	2025-05-03 05:19:22	2025-05-03 05:19:22	fF3ebKNFDhGYe6UVWNBbKTHAapY3
----	---	---------------	-----------	-----------------------------------------------------------	------	---------------------	---------------------	------------------------------

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





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

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

127.0.0.1:8000/api/register1

POST 127.0.0.1:8000/api/register1

Params Authorization Headers (8) Body Scripts Tests Settings Cookies

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

Key	Value	Description
username	staff03	
nama	STF_Bela	
password	12345	
password_confirmation	12345	
level_id	3	
image	kucing.jpg	

Body Cookies Headers (10) Test Results 201 Created • 601 ms • 587 B •

{ } JSON Preview Visualize

```
1 {
2   "success": true,
3   "user": {
4     "username": "staff03",
5     "nama": "STF_Bela",
6     "level_id": "3",
7     "image": "http://127.0.0.1:8000/storage/posts/N5Nw16JhPbLXa0BiPEkBIhpG5YdHnGP98pBCWLgX.jpg",
8     "updated_at": "2025-05-03T05:26:40.000000Z",
9     "created_at": "2025-05-03T05:26:40.000000Z",
10    "user_id": 33
11  }
12 }
```

<input type="checkbox"/>	Edit	Copy	Delete	33	3 staff03	STF_Bela	\$2y\$12\$KydW3y8G9CHkzg6RjAaOvuzEueCKaJLJr1qsVU9a...	NULL	2025-05-03 05:26:40	2025-05-03 05:26:40	N5Nw16JhPbLXa0BiPEkBIhpG5YdHnGP98pBCWLgX.jpg
With selected: Edit Copy Delete Export											

**Jawab:** Field image user akan disimpan menggunakan hash. hashName() adalah method untuk menghasilkan nama file yang unik dengan cara menghash nama file asli. Hash **digunakan untuk**



alasan keamanan dan mencegah konflik apabila ada user yang mengupload image dengan nama yang sama.

## 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.

### a. Tabel m\_barang

⇒ Buat Migrasi untuk menambahkan kolom image

```
PS D:\laragon\www\PWL_25\week11\jobsheet\PWL_POS> php artisan make:migration add_image_to_m_barang_table

INFO Migration [D:\laragon\www\PWL_25\week11\jobsheet\PWL_POS\database\Migrations\2025_05_03_053211_add_image_to_m_barang_table.php] created successfully.
```

⇒ Tambahkan function pada file migration yang telah dibuat

```
week11 > jobsheet > PWL_POS > database > migrations > 2025_05_03_053211_add_image_to_m_barang_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_barang', function (Blueprint $table) {
15             $table->string('image')->nullable();
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 };
```

⇒ Jalankan migrasi

```
php artisan migrate

INFO Running migrations.

2025_05_03_053211_add_image_to_m_barang_table 58ms DONE
```

		barang_id	kategori_id	barang_kode	barang_nama	harga_beli	harga_jual	created_at	updated_at	image
<input type="checkbox"/>	 Edit  Copy  Delete	1	1	BRG001	Facial Wash A	20000	30000	NULL	NULL	NULL
<input type="checkbox"/>	 Edit  Copy  Delete	2	1	BRG002	Facial Wash B	25000	35000	NULL	NULL	NULL

⇒ Modifikasi pada file BarangModel



```
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\casts\Attribute;
9  use Tymon\JWTAuth\Contracts\JWTSubject;
10
11 class BarangModel extends Model
12 {
13     use HasFactory;
14
15     protected $table = "m_barang";
16     protected $primaryKey = "barang_id";
17     protected $fillable = [
18         'kategori_id',
19         'barang_kode',
20         'barang_nama',
21         'harga_beli',
22         'harga_jual',
23         'image'
24     ];
25
26     public function kategori(): BelongsTo
27     {
28         return $this->belongsTo(KategoriModel::class, 'kategori_id', 'kategori_id');
29     }
30
31     public function image(): Attribute
32     {
33         return Attribute::make(
34             get: fn ($image) => url('/storage/barang/' . $image),
35         );
36     }
37 }
```

⇒ Modifikasi function pada file Api/BarangController



```
1 <?php
2
3 namespace App\Http\Controllers\Api;
4
5 use App\Http\Controllers\Controller;
6 use App\Models\Barang;
7 use Illuminate\Http\Request;
8 use Illuminate\Support\Facades\Validator;
9
10 class BarangController extends Controller
11 {
12     public function index()
13     {
14         $barang = Barang::all();
15
16         return response()->json([
17             'success' => true,
18             'data' => $barang,
19         ]);
20     }
21
22     public function store(Request $request)
23     {
24         $validator = Validator::make($request->all(), [
25             'kategori_id' => 'required|exists:kategori,kategori_id',
26             'barang_kode' => 'required|unique:barang,barang_kode',
27             'barang_nama' => 'required',
28             'harga_beli' => 'required|numeric',
29             'harga_jual' => 'required|numeric',
30             'image' => 'nullable|image|mimes:jpeg,png,jpg,gif,svg|max:2048',
31         ]);
32
33         if ($validator->fails()) {
34             return response()->json($validator->errors(), 422);
35         }
36
37         $imageName = null;
38         if ($request->hasFile('image')) {
39             $imageName = $request->image->hashName();
40             $request->image->storeAs('public/barang', $imageName);
41         }
42
43         $barang = Barang::create([
44             'kategori_id' => $request->kategori_id,
45             'barang_kode' => $request->barang_kode,
46             'barang_nama' => $request->barang_nama,
47             'harga_beli' => $request->harga_beli,
48             'harga_jual' => $request->harga_jual,
49             'image' => $imageName,
50         ]);
51
52         return response()->json([
53             'success' => true,
54             'data' => $barang,
55         ], 201);
56     }
57
58     public function show($id)
59     {
60         $barang = Barang::find($id);
61
62         if (!$barang) {
63             return response()->json([
64                 'success' => false,
65                 'message' => 'Barang tidak ditemukan',
66             ], 404);
67         }
68
69         return response()->json([
70             'success' => true,
71             'data' => $barang,
72         ]);
73     }
74
75     public function update(Request $request, $id)
76     {
77         $barang = Barang::find($id);
78
79         if (!$barang) {
80             return response()->json([
81                 'success' => false,
82                 'message' => 'Barang tidak ditemukan',
83             ], 404);
84         }
85
86         $validator = Validator::make($request->all(), [
87             'kategori_id' => 'sometimes|required|exists:kategori,kategori_id',
88             'barang_kode' => 'sometimes|required|unique:barang,barang_kode', $id . ' . ', 'barang_id',
89             'barang_nama' => 'sometimes|required',
90             'harga_beli' => 'sometimes|required|numeric',
91             'harga_jual' => 'sometimes|required|numeric',
92             'image' => 'nullable|image|mimes:jpeg,png,jpg,gif,svg|max:2048',
93         ]);
94
95         if ($validator->fails()) {
96             return response()->json($validator->errors(), 422);
97         }
98
99         if ($request->hasFile('image')) {
100             $imageName = $request->image->hashName();
101             $request->image->storeAs('public/barang', $imageName);
102             $barang->image = $imageName;
103         }
104
105         $barang->update($request->except('image'));
106         $barang->save();
107
108         return response()->json([
109             'success' => true,
110             'data' => $barang,
111         ]);
112     }
113
114     public function destroy($id)
115     {
116         $barang = Barang::find($id);
117
118         if (!$barang) {
119             return response()->json([
120                 'success' => false,
121                 'message' => 'Barang tidak ditemukan',
122             ], 404);
123         }
124
125         $barang->delete();
126
127         return response()->json([
128             'success' => true,
129             'message' => 'Barang berhasil dihapus',
130         ]);
131     }
132 }
133 }
```



⇒ Test pada Postman

- Create data barang menggunakan method POST

The screenshot shows a Postman interface for a POST request to `127.0.0.1:8000/api/barangs`. The request body is set to 'form-data' and contains the following fields:

Key	Value	Description
barang_nama	Gulaku Gula Pasir 1kg	
harga_beli	16500	
harga_jual	20000	
image	gulaku 1kg.jpeg	
barang_kode	SBK-001	
kategori_id	1	

The response is shown in JSON format:

```
{
  "success": true,
  "data": {
    "kategori_id": "1",
    "barang_kode": "SBK-001",
    "barang_nama": "Gulaku Gula Pasir 1kg",
    "harga_beli": "16500",
    "harga_jual": "20000",
    "image": "http://127.0.0.1:8000/storage/barang/DEEF4USrcktrpoX8I1LL0KzJZ8K6Kf1JehSVC7Fj.jpg",
    "updated_at": "2025-05-03T05:51:26.000000Z",
    "created_at": "2025-05-03T05:51:26.000000Z",
    "barang_id": 24
  }
}
```

Data berhasil tersimpan

	24	1 SBK-001	Gulaku Gula Pasir 1kg	16500	20000	2025-05-03 05:51:26	2025-05-03 05:51:26	DEEF4USrcktrpoX8I1LL0KzJZ8K6Kf1JehSVC7Fj.jpg
With selected:	Edit	Copy	Delete	Export				

- Read data barang menggunakan method GET

The screenshot shows a Postman interface for a GET request to `127.0.0.1:8000/api/barangs`. The response status is **200 OK** with a response time of 380 ms.



```
{
  "barang_id": 24,
  "kategori_id": 1,
  "barang_kode": "SBK-001",
  "barang_nama": "Gulaku Gula Pasir 1kg",
  "harga_beli": 16500,
  "harga_jual": 20000,
  "created_at": "2025-05-03T05:51:26.000000Z",
  "updated_at": "2025-05-03T05:51:26.000000Z",
  "image": "http://127.0.0.1:8000/storage/barang/DEEF4USrcktpoX8I1LL0KrJZ8K6Kf1JehSVC7Fj.jpg"
}
```

### b. Tabel t\_penjualan

⇒ Buat controller Api/PenjualanController

```
PS D:\laragon\www\PWL_25\week11\jobsheet\PWL_POS> php artisan make:controller Api/PenjualanController
```

```
INFO Controller [D:\laragon\www\PWL_25\week11\jobsheet\PWL_POS\app\Http\Controllers\Api\PenjualanController.php] created successfully.
```



```
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\PengjualanModel;
8 use Illuminate\Support\Facades\Storage;
9 use Illuminate\Support\Facades\Validator;
10
11 class PengjualanController extends Controller
12 {
13     // Menampilkan daftar semua transaksi pengjualan
14     public function index()
15     {
16         $data = PengjualanModel::with(['user', 'details.barang'])->get();
17         return response()->json($data);
18     }
19
20     // Menampilkan detail transaksi pengjualan berdasarkan ID
21     public function show($transaksi)
22     {
23         $pengjualan = PengjualanModel::with(['user', 'details.barang'])->findOrFail($transaksi);
24         return response()->json($pengjualan);
25     }
26
27     // Menyimpan transaksi pengjualan baru ke database
28     public function store(Request $request)
29     {
30         $v = Validator::make($request->all(), [
31             'user_id' => 'required|exists:m_user,user_id',
32             'pembeli' => 'required|string|max:100',
33             'pengjualan_kode' => 'required|string|unique:t_pengjualan,pengjualan_kode',
34             'pengjualan_tanggal' => 'required|date_format:Y-m-d H:i:s',
35             'image' => 'nullable|image|mimes:jpg,jpeg,png|max:2048',
36         ]);
37
38         if ($v->fails()) {
39             return response()->json(['success'=>false,'errors'=>$v->errors()], 422);
40         }
41
42         $transaksi = PengjualanModel::create($v->validated());
43
44         if ($request->hasFile('image')) {
45             $fn = time().'.'.$request->image->extension();
46             $request->image->storeAs('public/transaksi', $fn);
47             $transaksi->update(['image'=>$fn]);
48         }
49
50         return response()->json([
51             'success' => true,
52             'data' => [
53                 'pengjualan' => $transaksi,
54                 'image_url' => $transaksi->image ? asset('storage/transaksi/'.$transaksi->image) : null,
55             ],
56         ], 201);
57     }
58
59     // Memperbarui data transaksi pengjualan berdasarkan ID
60     public function update(Request $request, $transaksi)
61     {
62         $transaksi = PengjualanModel::findOrFail($transaksi);
63
64         $v = Validator::make($request->all(), [
65             'pembeli' => 'sometimes|required|string|max:100',
66             'pengjualan_kode' => 'sometimes|required|string|unique:t_pengjualan,pengjualan_kode, '.$transaksi->id,
67             'pengjualan_tanggal' => 'sometimes|required|date',
68             'image' => 'nullable|image|mimes:jpg,jpeg,png|max:2048',
69         ]);
70
71         if ($v->fails()) {
72             return response()->json(['success'=>false,'errors'=>$v->errors()], 422);
73         }
74
75         $transaksi->update($v->validated());
76
77         if ($request->hasFile('image')) {
78             if ($transaksi->image) {
79                 Storage::delete('public/transaksi/'.$transaksi->image);
80             }
81             $fn = time().'.'.$request->image->extension();
82             $request->image->storeAs('public/transaksi', $fn);
83             $transaksi->update(['image'=>$fn]);
84         }
85
86         return response()->json($transaksi);
87     }
88
89     // Menghapus transaksi pengjualan dari database berdasarkan ID
90     public function destroy($transaksi)
91     {
92         $transaksi = PengjualanModel::findOrFail($transaksi);
93
94         if ($transaksi->image && Storage::exists('public/transaksi/'.$transaksi->image)) {
95             Storage::delete('public/transaksi/'.$transaksi->image);
96         }
97         $transaksi->delete();
98         return response()->json(['success'=>true,'message'=>'Transaksi dihapus']);
99     }
100 }
```



⇒ Buat migrasi

```
PS D:\laragon\www\PWL_25\week11\jobsheet\PWL_POS> php artisan make:migration add_image_to_t_penjualan_table
```

**INFO** Migration [D:\laragon\www\PWL\_25\week11\jobsheet\PWL\_POS\database\Migrations\2025\_05\_03\_060448\_add\_image\_to\_t\_penjualan\_table.php] created successfully.

⇒ Tambahkan function pada file migration yang sudah dibuat

```
week11 > jobsheet > PWL_POS > database > migrations > 2025_05_03_060448_add_image_to_t_penjualan_table.php > ...
1
2 <?php
3
4 use Illuminate\Database\Migrations\Migration;
5 use Illuminate\Database\Schema\Blueprint;
6 use Illuminate\Support\Facades\Schema;
7
8 return new class extends Migration
9 {
10     /**
11      * Run the migrations.
12      */
13     public function up(): void
14     {
15         Schema::table('t_penjualan', function (Blueprint $table) {
16             $table->string('image')->nullable()->after('penjualan_tanggal');
17         });
18     }
19
20     /**
21      * Reverse the migrations.
22      */
23     public function down(): void
24     {
25         Schema::table('t_penjualan', function (Blueprint $table) {
26             $table->dropColumn('image');
27         });
28     }
29 };
```

⇒ Jalankan migrasi

```
PS D:\lphp artisan migrate
```

**INFO** Running migrations.

2025\_05\_03\_060448\_add\_image\_to\_t\_penjualan\_table 466ms **DONE**

Showing rows 0 - 0 (1 total, Query took 0.0015 seconds.)

SELECT \* FROM `t\_penjualan`

Profiling [ Edit inline ] [ Edit ] [ Explain SQL ] [ Create PHP code ] [ Refresh ]

Show all | Number of rows: 25 | Filter rows: Search this table

Extra options

	penjualan_id	user_id	pembeli	penjualan_kode	penjualan_tanggal	image	created_at	updated_at
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	1	29	Sofyan W	PNJ-001	2025-05-03 14:30:00	NULL	NULL	NULL

Check all | With selected: Edit Copy Delete Export

⇒ Modifikasi pada file PenjualanModel





```
<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;
use Illuminate\Database\Eloquent\Relations\BelongsTo;
use Illuminate\Database\Eloquent\Relations\HasMany;
use App\Models\UserModel;
use App\Models\DetailPenjualanModel;
use Illuminate\Database\Eloquent\Casts\Attribute;
use Tymon\JWTAuth\Contracts\JWTSubject;
use Illuminate\Foundation\Auth\User as Authenticatable; //i

class PenjualanModel extends Model
{
    public function getJWTIdentifier(){
        return $this->getKey();
    }

    use HasFactory;
    protected $table = 't_penjualan';
    protected $primaryKey = 'penjualan_id';

    protected $fillable = ['user_id', 'pembeli', 'penjualan_kode', 'penjualan_tanggal', 'image'];

    51     public function image(): Attribute
    52     {
    53         return Attribute::make(
    54             get: fn ($image) => url('/storage/transaksi/' . $image),
    55         );
    56     }
    57 }
```

⇒ Tambahkan route pada file api.php

```
use App\Http\Controllers\Api\PenjualanController;

//Route API Transaksi Penjualan
Route::get('/penjualans', [PenjualanController::class, 'index']);
Route::post('/penjualans', [PenjualanController::class, 'store']);
Route::get('/penjualans/{penjualan}', [PenjualanController::class, 'show']);
Route::put('/penjualans/{penjualan}', [PenjualanController::class, 'update']);
Route::delete('/penjualans/{penjualan}', [PenjualanController::class, 'destroy']);
```

⇒ Test pada Postman



127.0.0.1:8000/api/penjualans

GET 127.0.0.1:8000/api/penjualans

Send

Params Authorization Headers (6) Body Scripts Tests Settings Cookies

Key	Value	Description
Key	Value	Description

Body Cookies Headers (10) Test Results

200 OK • 6.87 s • 787 B

```
{
  "penjualan_id": 1,
  "user_id": 29,
  "pembeli": "Sofyan W",
  "penjualan_kode": "PNJ-001",
  "penjualan_tanggal": "2025-05-03 14:30:00",
  "image": "http://127.0.0.1:8000/storage/transaksi",
  "created_at": null,
  "updated_at": null,
  "user": {
    "user_id": 29,
    "level_id": 2,
    "username": "penggunasatu",
    "nama": "Pengguna1"
  }
}
```

- Create data transaksi dengan method POST

POST 127.0.0.1:8000/api/penjualans

Send

Params Authorization Headers (8) Body Scripts Tests Settings Cookies

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

Key	Value	Description
<input checked="" type="checkbox"/> user_id	Text 33	
<input checked="" type="checkbox"/> pembeli	Text Andhika	
<input checked="" type="checkbox"/> penjualan_kode	Text PNJ-023	
<input checked="" type="checkbox"/> penjualan_tanggal	Text 2025-05-03 00:00:00	
<input checked="" type="checkbox"/> image	File Nota_Kosong_1_Ply_1_Le...	

Body Cookies Headers (10) Test Results

201 Created • 2.34 s • 748 B

```
{
  "success": true,
  "data": {
    "penjualan": {
      "user_id": "33",
      "pembeli": "Andhika",
      "penjualan_kode": "PNJ-023",
      "penjualan_tanggal": "2025-05-03 00:00:00",
      "image": "http://127.0.0.1:8000/storage/transaksi/1746253448.jpg",
      "updated_at": "2025-05-03T06:24:10.000000Z",
      "created_at": "2025-05-03T06:24:08.000000Z",
      "penjualan_id": 2
    },
    "image_url": "http://127.0.0.1:8000/storage/transaksi/http://127.0.0.1:8000/storage/transaksi/1746253448.jpg"
  }
}
```



- Read/menampilkan data transaksi dengan method GET

GET 127.0.0.1:8000/api/penjualans Send

Params Authorization Headers (8) Body Scripts Tests Settings Cookies

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

	Key		Value	Description	*** Bulk Edit
<input checked="" type="checkbox"/>	user_id	Text	33		
<input checked="" type="checkbox"/>	pembeli	Text	Andhika		

{ } JSON Preview Visualize

```
24      "penjualan_id": 2,  
25      "user_id": 33,  
26      "pembeli": "Andhika",  
27      "penjualan_kode": "PNJ-023",  
28      "penjualan_tanggal": "2025-05-03 00:00:00",  
29      "image": "http://127.0.0.1:8000/storage/transaksi/1746253448.jpg",  
30      "created_at": "2025-05-03T06:24:08.000000Z",  
31      "updated_at": "2025-05-03T06:24:10.000000Z",  
32      "user": {  
33          "user_id": 33,  
34          "level_id": 3,  
35          "username": "staff03",  
36          "nama": "STF_Bela",  
37          "user_profile_picture": null,  
38          "created_at": "2025-05-03T05:26:40.000000Z",  
39          "updated_at": "2025-05-03T05:26:40.000000Z",  
40          "image": "http://127.0.0.1:8000/storage/posts/N5Nw16JhPbLXa0BiPEkBIhpG5YdHnGP98pBCWLgX.jpg",  
41      },  
42      "details": []
```

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