



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

```
C:\laragon\www\PWL_2025\Week11\Jobsheet>php artisan make:migration add_image_to_m_user_table
```

```
INFO Migration [C:\laragon\www\PWL_2025\Week11\Jobsheet\database\Migrations\2025_05_06_202627_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`
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),
        );
    }
}
```

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);
}
}
```



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

RESTful API basics: CRUD, test & variable / <http://127.0.0.1:8000/api/register1>

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

Params Authorization Headers (8) Body Pre-request Script Tests Settings Cookies

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

| Key | Value | Content-Type | Description |
|---|---------------------------------------|--------------|-------------|
| <input checked="" type="checkbox"/> username | Text penggunaempat | Auto | |
| <input checked="" type="checkbox"/> nama | Text pengguna4 | Auto | |
| <input checked="" type="checkbox"/> password | Text 12345 | Auto | |
| <input checked="" type="checkbox"/> password_confirmation | Text 12345 | Auto | |
| <input checked="" type="checkbox"/> level_id | Text 2 | Auto | |
| <input checked="" type="checkbox"/> image | File Screenshot 2023-04-28 095851.png | Auto | |

body Cookies Headers (10) Test Results Status: 201 Created Time: 591 ms Size: 691 B Save as example

Pretty Raw Preview Visualize JSON

```
1 {
2   "success": true,
3   "user": {
4     "username": "penggunaempat",
5     "nama": "pengguna4",
6     "password": "$2y$12$9sd1lI/uJfy33ABuUj9s50tkYeJr1G9ci.Sy/vbyg5CYjgvh0vYlm",
7     "level_id": "2",
8     "image": "http://127.0.0.1:8000/storage/posts/c:/Users/Asus Zenbook 14 OLED/AppData/Local/Temp/php57C5.tmp",
9     "updated_at": "2024-04-30T07:16:25.000000Z",
10    "created_at": "2024-04-30T07:16:25.000000Z",
11    "user_id": 22
12  }
13 }
```

Hasil:

<http://127.0.0.1:8000/api/register1> Save Share

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

Params Authorization Headers (8) Body Scripts Settings Cookies

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

| Key | Value | Content-Type | Description |
|---|--------------------|--------------|-------------|
| <input checked="" type="checkbox"/> username | Text penggunaempat | | |
| <input checked="" type="checkbox"/> nama | Text pengguna4 | | |
| <input checked="" type="checkbox"/> password | Text 12345 | | |
| <input checked="" type="checkbox"/> password_confirmation | Text 12345 | | |
| <input checked="" type="checkbox"/> level_id | Text 2 | | |
| <input checked="" type="checkbox"/> image | File download.jpeg | | |

Body Cookies Headers (10) Test Results 201 Created 2.05 s 594 B

JSON Preview Visualize

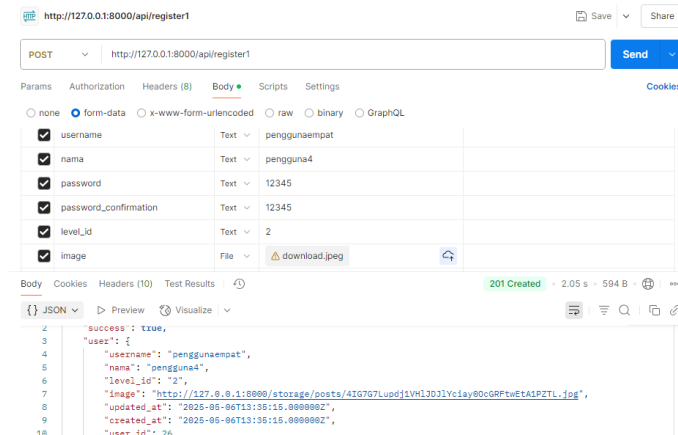
```
2 {
3   "success": true,
4   "user": {
5     "username": "penggunaempat",
6     "nama": "pengguna4",
7     "level_id": "2"
8   }
9 }
```



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

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

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



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.

- M_barang

- Make migration for image

```
C:\laragon\www\PWL_2025\Week11\Jobsheet>php artisan make:migration add_image_to_m_barang_table
```

```
INFO Migration [C:\laragon\www\PWL_2025\Week11\Jobsheet\database\Migrations\2025_05_06_204307_add_image_to_m_barang_table.php] created successfully.
```

- Modify migration

```
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     public function up(): void
10    {
11        Schema::table('m_barang', function (Blueprint $table) {
12            $table->string('image')->nullable(); // boleh null
13        });
14    }
15
16    public function down(): void
17    {
18        Schema::table('m_barang', function (Blueprint $table) {
19            $table->dropColumn('image');
20        });
21    }
22 };
```




■ Migrate

INFO Running migrations.

2025_05_06_204307_add_image_to_m_barang_table 61ms **DONE**

■ Edit controller

```
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         $barang = BarangModel::all();
15         return response()->json([
16             'success' => true,
17             'data' => $barang,
18         ]);
19     }
20
21     public function store(Request $request)
22     {
23         // Validasi
24         $validator = Validator::make($request->all(), [
25             'kategori_id' => 'required|exists:m_kategori,kategori_id',
26             'barang_kode' => 'required|unique:m_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         // Upload image kalau ada
38         $imageName = null;
39         if ($request->hasFile('image')) {
40             $imageName = $request->image->hashName();
41             $request->image->storeAs('public/barang', $imageName);
42         }
43
44         $barang = BarangModel::create([
45             'kategori_id' => $request->kategori_id,
46             'barang_kode' => $request->barang_kode,
47             'barang_nama' => $request->barang_nama,
48             'harga_beli' => $request->harga_beli,
49             'harga_jual' => $request->harga_jual,
50             'image' => $imageName,
51         ]);
52
53         return response()->json([
54             'success' => true,
55             'data' => $barang,
56         ], 201);
57     }
58
59     public function show($id)
60     {
61         $barang = BarangModel::find($id);
62
63         if (!$barang) {
64             return response()->json([
65                 'success' => false,
66                 'message' => 'Barang tidak ditemukan',
67             ], 404);
68         }
69     }
```

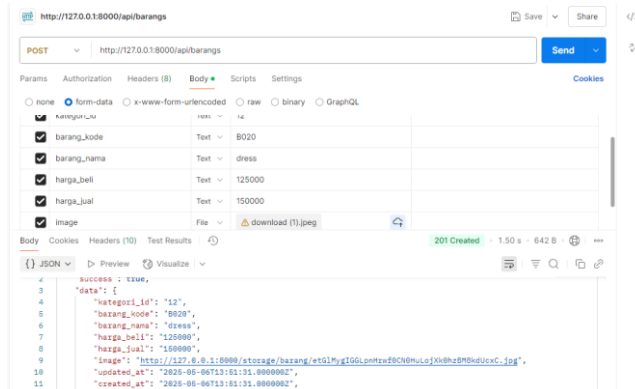
```
70     return response()->json([
71         'success' => true,
72         'data' => $barang,
73     ]);
74 }
75
76 public function update(Request $request, $id)
77 {
78     $barang = BarangModel::find($id);
79
80     if (!$barang) {
81         return response()->json([
82             'success' => false,
83             'message' => 'Barang tidak ditemukan',
84         ], 404);
85     }
86
87     // Validasi
88     $validator = Validator::make($request->all(), [
89         'kategori_id' => 'sometimes|required|exists:m_kategori,kategori_id',
90         'barang_kode' => 'sometimes|required|unique:m_barang,barang_kode', $id -> 'barang_id',
91         'barang_nama' => 'sometimes|required',
92         'harga_beli' => 'sometimes|required|numeric',
93         'harga_jual' => 'sometimes|required|numeric',
94         'image' => 'nullable|image|mimes:jpeg,png,jpg,gif,svg|max:2048',
95     ]);
96
97     if ($validator->fails()) {
98         return response()->json($validator->errors(), 422);
99     }
100
101     // Upload image kalau ada
102     if ($request->hasFile('image')) {
103         $imageName = $request->image->hashName();
104         $request->image->storeAs('public/barang', $imageName);
105         $barang->image = $imageName;
106     }
107
108     $barang->update($request->except('image')); // Kalau image, karena sudah dihandle manual
109     $barang->save();
110
111     return response()->json([
112         'success' => true,
113         'data' => $barang,
114     ]);
115 }
116
117 public function destroy($id)
118 {
119     $barang = BarangModel::find($id);
120
121     if (!$barang) {
122         return response()->json([
123             'success' => false,
124             'message' => 'Barang tidak ditemukan',
125         ], 404);
126     }
127
128     $barang->delete();
129
130     return response()->json([
131         'success' => true,
132         'message' => 'Barang berhasil dihapus',
133     ]);
134 }
135 }
```

■ Edit barangmodel

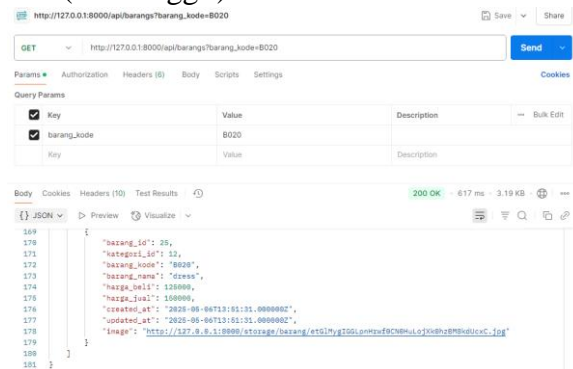
```
1 <?php
2
3 namespace App\Models;
4
5 use Illuminate\Database\Eloquent\Factories\HasFactory;
6 use Illuminate\Database\Eloquent\Model;
7 use Illuminate\Database\Eloquent\Relations\BelongsTo;
8 use Illuminate\Database\Eloquent\Relations\BelongsToMany;
9 use Illuminate\Contracts\Auth\Authenticatable;
10
11 class BarangModel extends Model
12 {
13     public function getMofidInstance()
14     {
15         return $this->getMofidInstance();
16     }
17
18     public function getMofidInstance()
19     {
20         return [];
21     }
22
23     use HasFactory;
24
25     protected $table = 'm_barang';
26     protected $primaryKey = 'barang_id';
27     protected $fillable = [
28         'kategori_id',
29         'barang_kode',
30         'barang_nama',
31         'harga_beli',
32         'harga_jual',
33         'image', // Column
34     ];
35
36     public function kategori(): BelongsToMany
37     {
38         return $this->belongsToMany(KategoriModel::class, 'kategori_id', 'barang_id');
39     }
40
41     public function stok()
42     {
43         return $this->hasMany(StokModel::class, 'barang_id', 'barang_id');
44     }
45
46     protected function image(): Attribute
47     {
48         return Attribute::make(
49             get: fn ($image) => $image ? url('storage/barang/' . $image) : null,
50         );
51     }
52 }
```




■ Post (create)



■ Get (memanggil)



● Transaksi

■ Make controller

INFO Controller [C:\laragon\www\PWL_2025\Week11\Jobsheet\app\Http\Controllers\Api\PenjualanController.php] created successfully.

■ Modify controller

```
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\PenjualanModel;
8 use Illuminate\Support\Facades\Storage;
9 use Illuminate\Support\Facades\Validator;
10
11 class PenjualanController extends Controller
12 {
13     public function index()
14     {
15         $data = PenjualanModel::select('user', 'details_barang')->get();
16         return response()->json($data);
17     }
18
19     public function show($transaksi)
20     {
21         $penjualan = PenjualanModel::select('user', 'details_barang')->findOrFail($transaksi);
22         return response()->json($penjualan);
23     }
24
25     public function store(Request $request)
26     {
27         $v = Validator::make($request->all(), [
28             'user_id' => 'required|exists:users,user_id',
29             'kategori' => 'required|string|max:255',
30             'barang_kode' => 'required|string|unique:penjualan,barang_kode',
31             'barang_nama' => 'required|string|max:255',
32             'harga_beli' => 'required|numeric',
33             'harga_jual' => 'required|numeric',
34             'image' => 'nullable|image|mimes:jpg,jpeg,png|max:2048',
35         ]);
36
37         if ($v->fails()) {
38             return response()->json(['success'=>false, 'errors'=>$v->errors()], 422);
39         }
40
41         $strx = PenjualanModel::create($v->validated());
42
43         if ($request->hasFile('image')) {
44             $fn = time().'-'.request->image->extension();
45             $request->image->storeAs('public/transaksi', $fn);
46             $strx->update(['image'=>$fn]);
47         }
48
49         return response()->json([
50             'success'=>true,
51             'penjualan'=>$strx,
52             'image_url'=>$strx->image->url() ? asset('storage/transaksi/'.$strx->image) : null,
53         ], 201);
54     }
55 }
```

```
56 public function update(Request $request, $transaksi)
57 {
58     $strx = PenjualanModel::findOrFail($transaksi);
59
60     $v = Validator::make($request->all(), [
61         'kategori' => 'sometimes|required|string|max:255',
62         'barang_kode' => 'sometimes|required|string|unique:penjualan,barang_kode',
63         'barang_nama' => 'sometimes|required|string|max:255',
64         'harga_beli' => 'sometimes|required|numeric',
65         'harga_jual' => 'sometimes|required|numeric',
66         'image' => 'nullable|image|mimes:jpg,jpeg,png|max:2048',
67     ]);
68
69     if ($v->fails()) {
70         return response()->json(['success'=>false, 'errors'=>$v->errors()], 422);
71     }
72
73     $strx->update($v->validated());
74
75     if ($request->hasFile('image')) {
76         if ($strx->image) {
77             Storage::delete('public/transaksi/'.$strx->image);
78         }
79         $fn = time().'-'.request->image->extension();
80         $request->image->storeAs('public/transaksi', $fn);
81         $strx->update(['image'=>$fn]);
82     }
83
84     return response()->json($strx);
85 }
86
87 public function destroy($transaksi)
88 {
89     $strx = PenjualanModel::findOrFail($transaksi);
90     //cek file exists
91     if ($strx->image && Storage::exists('public/transaksi/'.$strx->image)) {
92         Storage::delete('public/transaksi/'.$strx->image);
93     }
94     $strx->delete();
95     return response()->json(['success'=>true, 'message'=>'transaksi dihapus']);
96 }
```



- Make migration for image

```
C:\laragon\www\pwl_2025\Week11\Jobsheet>php artisan make:migration add_image_to_t_penjualan_table
```

```
INFO Migration [C:\laragon\www\pwl_2025\Week11\Jobsheet\database\Migrations\2025_05_06_210011_add_image_to_t_penjualan_table.php] created successfully.
```

- Modify migration

```
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     public function up(): void
10     {
11         Schema::table('t_penjualan', function (Blueprint $table) {
12             $table->string('image')->nullable()->after('penjualan_tanggal');
13         });
14     }
15
16     public function down(): void
17     {
18         Schema::table('t_penjualan', function (Blueprint $table) {
19             $table->dropColumn('image');
20         });
21     }
22 };
23
```

- Migrate

```
INFO Running migrations.
```

```
2025_05_06_210011_add_image_to_t_penjualan_table ..... 62ms DONE
```

- Edit penjualanmodel

```
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\HasMany;
9 use App\Models\UserModel;
10 use App\Models\DetailPenjualanModel;
11 use Illuminate\Database\Eloquent\CastAttribute;
12 use Tymon\JWTAuth\Contracts\JWTSubject;
13 use Illuminate\Foundation\Auth\User as Authenticatable;
14
15 class PenjualanModel extends Model
16 {
17
18     public function getJWTIdentifier(){
19         return $this->getKey();
20     }
21
22     public function getJWTCustomClaims(){
23         return [];
24     }
25
26     use HasFactory;
27     protected $table = 't_penjualan';
28     protected $primaryKey = 'penjualan_id';
29     /**
30      * The attributes that are mass assignable.
31      *
32      * @var array
33      */
34     protected $fillable = ['user_id', 'pembeli', 'penjualan_kode', 'penjualan_tanggal', 'image'];
35
36     //Relasi ke tabel user
37     public function user(): BelongsTo
38     {
39         return $this->belongsTo(UserModel::class, 'user_id', 'user_id');
40     }
41
42     public function details(): HasMany
43     {
44         return $this->hasMany(DetailPenjualanModel::class, 'penjualan_id', 'penjualan_id');
45     }
46
47     // Menambahkan method untuk mendapatkan nama
48     public function getUsernameAttribute()
49     {
50         return $this->user->name ?? null; // Ambil nama dari relasi user
51     }
52
53     protected function image(): Attribute
54     {
55         return Attribute::make(
56             get: fn ($image) => $image ? url('/storage/transaksi/' . $image) : null
57         );
58     }
59 }
```



- Edit routes in API

```
1 // Route Transaksi
2 Route::get('transaksi', [PenjualanController::class, 'index']);
3 Route::get('transaksi/{transaksi}', [PenjualanController::class, 'show']);
4 Route::post('transaksi', [PenjualanController::class, 'store']);
5 Route::post('transaksi/{transaksi}', [PenjualanController::class, 'update']);
6 Route::delete('transaksi/{transaksi}', [PenjualanController::class, 'destroy']);
```

- Post (create)

POST http://127.0.0.1:8000/api/transaksi

201 Created - 628 ms - 750 B

```
{
  "success": true,
  "data": {
    "penjualan": {
      "user_id": 2,
      "pembeli": "Customer33",
      "penjualan_kode": "TRX060",
      "penjualan_tanggal": "2025-04-02 11:12:00",
      "image": "http://127.0.0.1:8000/storage/transaksi/1746549458.jpg",
      "created_at": "2025-05-06T14:07:35.000000Z",
      "updated_at": "2025-05-06T14:07:35.000000Z"
    }
  }
}
```

- Get(memanggil)

GET http://127.0.0.1:8000/api/transaksi?penjualan_kode=TRX060

200 OK - 509 ms - 15.21 KB

```
{
  "penjualan_id": 20,
  "user_id": 2,
  "pembeli": "Customer33",
  "penjualan_kode": "TRX060",
  "penjualan_tanggal": "2025-04-02 11:12:00",
  "image": "http://127.0.0.1:8000/storage/transaksi/1746549458.jpg",
  "created_at": "2025-05-06T14:07:35.000000Z",
  "updated_at": "2025-05-06T14:07:35.000000Z",
  "user": {
    "user_id": 2,
    "level_id": 2,
    "profile_picture": null,
    "username": "manager",
    "nama": "Manager",
    "created_at": null,
    "updated_at": null,
    "image": "http://127.0.0.1:8000/storage/posts"
  }
}
```

*** Sekian, dan selamat belajar ***