



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 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_04_30_040822_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');
        });
    }
};
```

```
<?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('foto')->nullable()->after('level_id');
        });
    }

    public function down(): void
    {
        Schema::table('m_user', function (Blueprint $table) {
            $table->dropColumn('foto');
        });
    }
};
```



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),
        );
    }
}
```



```
class UserModel extends Authenticatable implements JWTSubject
{
    use HasFactory;

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

    0 references
    protected $fillable = [
        'username',
        'password',
        'nama',
        'level_id',
        'foto',
        'created_at',
        'updated_at',
    ];

    0 references
    protected $hidden = ['password'];

    0 references
    protected $casts = [
        'password' => 'hashed',
    ];

    // Implementasi JWTSubject
    0 references | 0 overrides
    public function getJWTIdentifier(): mixed
    {
        return $this->getKey();
    }

    0 references | 0 overrides
    public function getJWTCustomClaims(): array
    {
        return [];
    }

    // Accessor untuk foto
    0 references | 0 overrides
    protected function foto(): Attribute
    {
        return Attribute::make(
            get: fn($foto): string|UrlGenerator|null => $foto ? url(path: 'storage/posts/' . $foto) : null
        );
    }
}
```

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);
    }
}
```



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

        // If validation fails
        if ($validator->fails()) {
            return response()->json(data: $validator->errors(), status: 422);
        }

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

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

        // Return JSON if insert failed
        return response()->json(data: [
            'success' => false,
        ], status: 409);
    }
}
```

7. Anda dapat menambahkan detail untuk spesifikasi image pada validator

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

```
0 references | 0 overrides
public function __invoke(Request $request): JsonResponse|mixed
{
    // Set validation
    $validator = Validator::make(data: $request->all(), rules: [
        'username' => 'required',
        'nama' => 'required',
        'password' => 'required|min:5|confirmed',
        'level_id' => 'required',
        'foto' => '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> Save

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	Bulk Edit
<input checked="" type="checkbox"/> username	Text <input type="text"/> penggunaempat	Auto		
<input checked="" type="checkbox"/> nama	Text <input type="text"/> pengguna4	Auto		
<input checked="" type="checkbox"/> password	Text <input type="text"/> 12345	Auto		
<input checked="" type="checkbox"/> password_confirmation	Text <input type="text"/> 12345	Auto		
<input checked="" type="checkbox"/> level_id	Text <input type="text"/> 2	Auto		
<input checked="" type="checkbox"/> image	File <input type="text"/> 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/u3fy33ABuJ9s50tkYe3rlG9ci.SyNvbyg5CY3gvh6vYNm",
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 }
```

POST <http://localhost:8000/api/register1>

Params Authorization Headers (15) **Body** Scripts Settings

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

<input checked="" type="checkbox"/> username	Text <input type="text"/> penggunaempat
<input checked="" type="checkbox"/> nama	Text <input type="text"/> pengguna4
<input checked="" type="checkbox"/> password	Text <input type="text"/> 12345
<input checked="" type="checkbox"/> password_confirmation	Text <input type="text"/> 12345
<input checked="" type="checkbox"/> level_id	Text <input type="text"/> 2
<input checked="" type="checkbox"/> foto	File <input type="text"/> 10083458_1.png
Key	Text <input type="text"/> Value

body Cookies Headers (10) Test Results

{ } JSON Preview Visualize

```
1 {
2   "success": true,
3   "user": {
4     "username": "penggunaempat",
5     "nama": "pengguna4",
6     "level_id": "2",
7     "foto": "http://localhost:8000/storage/posts/C:\\Users\\LENOVO\\AppData\\Local\\Temp\\php2C99.tmp",
8     "updated_at": "2025-04-27T20:45:24.000000Z",
9     "created_at": "2025-04-27T20:45:24.000000Z",
10    "user_id": 38
11  }
```

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

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



```
$user = UserModel::create(attributes: [  
    'username' => $request->username,  
    'nama' => $request->nama,  
    'password' => bcrypt(value: $request->password),  
    'level_id' => $request->level_id,  
    'foto' => $foto->hashName(), // Simpan nama file foto  
]);
```

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

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

<input checked="" type="checkbox"/>	username	Text	penggunatiga
<input checked="" type="checkbox"/>	nama	Text	pengguna3
<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"/>	foto	File	10083458_1.png
	Key	Text	Value

Body Cookies Headers (10) Test Results

JSON Preview Visualize

```
1 {  
2   "success": true,  
3   "user": {  
4     "username": "penggunatiga",  
5     "nama": "pengguna3",  
6     "level_id": "2",  
7     "foto": "http://localhost:8080/storage/posts/tqQRW3iNax48rh8ApAaw7P6TwFTNidd8Ya43u6I8.png",  
8     "updated_at": "2025-04-27T21:02:10.000000Z",  
9     "created_at": "2025-04-27T21:02:10.000000Z",  
10    "user_id": 40  
11  }  
12 }
```

Sesudah pakai hashName(): nama file di server jadi acak dan unik

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

Post

POST ▼ <http://localhost:8000/api/barang1>

Params Authorization Headers (10) **Body** Scripts Settings

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

<input checked="" type="checkbox"/> kategori_id	Text ▼	1
<input checked="" type="checkbox"/> barang_kode	Text ▼	BRG100
<input checked="" type="checkbox"/> barang_nama	Text ▼	MEJA
<input checked="" type="checkbox"/> barang_foto	File ▼	10083458_1.png
<input checked="" type="checkbox"/> harga_beli	Text ▼	5000
<input checked="" type="checkbox"/> harga_jual	Text ▼	6000
Key	Value ▼	

Body Cookies (2) Headers (10) Test Results ↻

{} JSON ▼ ▶ Preview 🔗 Visualize ▼

```
1 {
2   "success": true,
3   "message": "Barang berhasil disimpan",
4   "data": {
5     "kategori_id": "1",
6     "barang_kode": "BRG100",
7     "barang_nama": "MEJA",
8     "barang_foto": "http://localhost:8000/storage/barang/whV73Qh1SzBw0n1mPTH04QA9o40aJjH686xWmwcF.png",
9     "harga_beli": "5000",
10    "harga_jual": "6000",
11    "updated_at": "2025-04-27T21:21:14.000000Z",
12    "created_at": "2025-04-27T21:21:14.000000Z",
13    "barang_id": 59
14  }
15 }
```

Get



GET <http://localhost:8000/api/barang/59>

Params Authorization Headers (8) Body Scripts Settings

Body Cookies (2) Headers (10) Test Results

JSON Preview Visualize

```
1 {
2   "success": true,
3   "data": {
4     "barang_id": 59,
5     "kategori_id": 1,
6     "barang_kode": "BRG100",
7     "barang_nama": "MEJA",
8     "barang_foto": "http://localhost:8000/storage/barang/whV73Qh1Sz8wOn1mPTH04QA9o40aJjH686rWmwcf.png",
9     "harga_beli": 5000,
10    "harga_jual": 6000,
11    "created_at": "2025-04-27T21:21:14.000000Z",
12    "updated_at": "2025-04-27T21:21:14.000000Z"
13  }
14 }
```

Transaksi

POST <http://localhost:8000/api/penjualan>

Params Authorization Headers (10) Body Scripts Settings

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

```
1 {
2   "user_id": 1,
3   "pembeli": "Pelanggan 10",
4   "penjualan_kode": "PJL010",
5   "penjualan_tanggal": "2025-04-28",
6   "detail": [
7     {
8       "barang_id": 59,
9       "harga": 6000,
10      "jumlah": 2
11    }
12  ]
13 }
```

Body Cookies (2) Headers (10) Test Results

JSON Preview Visualize

```
1 {
2   "success": true,
3   "message": "Transaksi berhasil disimpan",
4   "data": {
5     "user_id": 1,
6     "pembeli": "Pelanggan 10",
7     "penjualan_kode": "PJL010",
8     "penjualan_tanggal": "2025-04-28",
9     "updated_at": "2025-04-27T21:35:50.000000Z",
10    "created_at": "2025-04-27T21:35:50.000000Z",
11    "penjualan_id": 12
12  }
13 }
```

<input type="checkbox"/>	Edit Copy Delete	12	1 Pelanggan 10 PJL010	2025-04-28 00:00:00	2025-04-27 21:35:50	2025-04-27 21:35:50
--------------------------	------------------	----	-----------------------	---------------------	---------------------	---------------------

Get



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>

GET <http://localhost:8000/api/penjualan/12>

Params Authorization Headers (8) **Body** Scripts Settings

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

This request does not have a body

Body Cookies (2) Headers (10) Test Results Visualize

{ } JSON Preview Visualize

```
1 {
2   "success": true,
3   "data": {
4     "penjualan_id": 12,
5     "user_id": 1,
6     "pembeli": "Pelanggan 10",
7     "penjualan_kode": "PJL010",
8     "penjualan_tanggal": "2025-04-28 00:00:00",
9     "created_at": "2025-04-27T21:35:50.000000Z",
10    "updated_at": "2025-04-27T21:35:50.000000Z",
11    "details": [
12      {
13        "detail_id": 33,
14        "penjualan_id": 12,
15        "barang_id": 59,
16        "harga": 6000,
17        "jumlah": 2,
18        "created_at": "2025-04-27T21:35:50.000000Z",
19        "updated_at": "2025-04-27T21:35:50.000000Z",
20        "barang": {
21          "barang_id": 59,
22          "kategori_id": 1,
23          "barang_kode": "BRG100",
24          "barang_nama": "MEJA",
25          "barang_foto": "http://localhost:8000/storage/barang/whV73Qh1Sz8w0n1mPTH04QA9o40aJjH686zWmwcf.png",
26          "harga_beli": 5000,
27          "harga_jual": 6000,
28          "created_at": "2025-04-27T21:21:14.000000Z",
29          "updated_at": "2025-04-27T21:21:14.000000Z"
30        }
31      }
32    ]
33  }
34 }
```

*** Sekian, dan selamat belajar ***