



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 : Aditya Yuhanda Putra

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

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

```
PS C:\laragon\www\PWL2025\PWL_Week11\Js11\PWL_POS> php artisan migrate
INFO Running migrations.
2025_04_28_085845_add_image_to_m_user_table ..... 250ms 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),
        );
    }
}
```

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

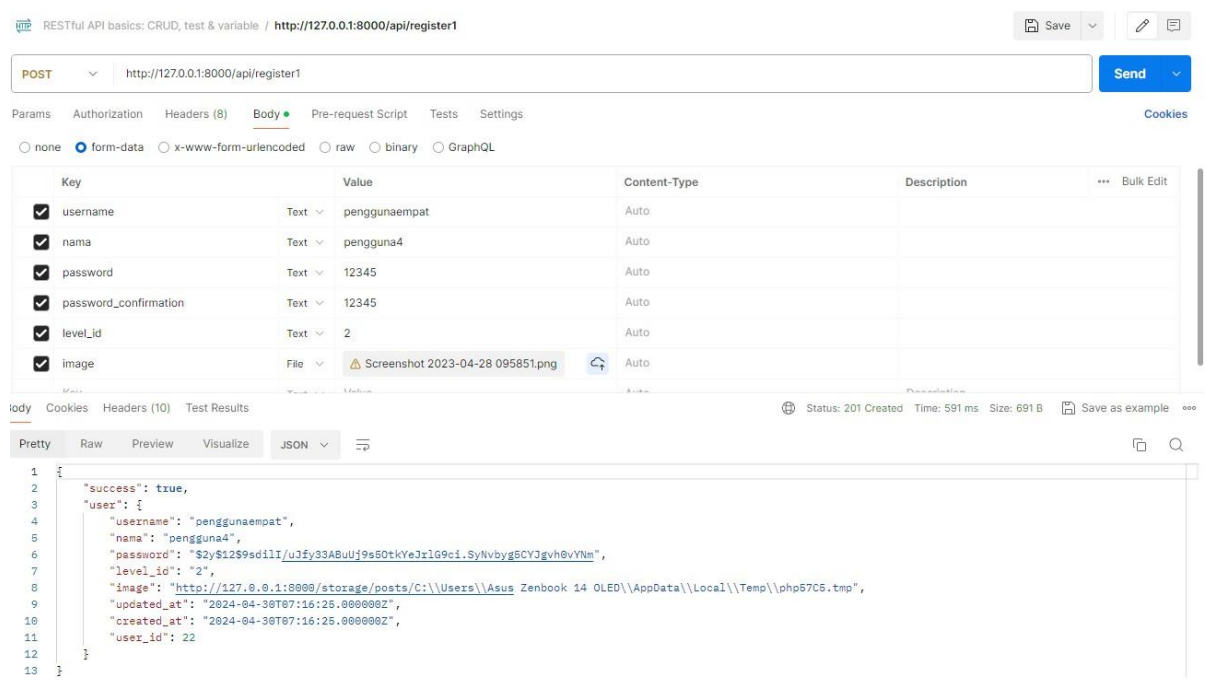
```
'image' => 'required|image|mimes:jpeg,png,jpg,gif,svg|max:2048',
```

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

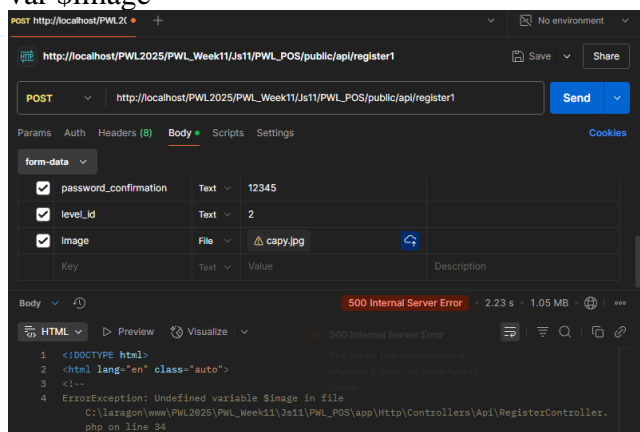


9. Pada Controllers/Api/RegisterController bagian create user ganti dengan

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

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

- Jawab : ketika dicoba, error, karena tidak ada request yang diterima dan langsung bernama var \$image





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.

1. Buat migration untuk gambar/file pada tabel m_barang

```
PS C:\laragon\www\PWL2025\PWL_Week11\Js11\PWL_POS> php artisan make:migration add_image_to_m_barang_table  
INFO Migration [C:\laragon\www\PWL2025\PWL_Week11\Js11\PWL_POS\database\migrations\2025_04_28_115635_add_image_to_m_barang_table.php] created successfully.
```

2. Isi migration nya

```
database > migrations > 2025_04_28_115635_add_image_to_m_barang_table.php > class > up > Closure  
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     Windsurf: Refactor | Explain | Generate Function Comment | X  
10     public function up(): void  
11     {  
12         Schema::table('m_barang', callback: function (Blueprint $table): void {  
13             $table->string(column: 'image')->nullable();  
14         });  
15  
16     Windsurf: Refactor | Explain | Generate Function Comment | X  
17     public function down(): void  
18     {  
19         Schema::table('m_barang', callback: function (Blueprint $table): void {  
20             $table->dropColumn(columns: 'image');  
21         });  
22     }  
23 };
```

3. Update isi kode BarangController.php yang ada pada /api dengan menambahkan image

```
app > Http > Controllers > Api > BarangController.php > PHP > BarangController  
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 6 references | 0 implementations | Windsurf: Refactor | Explain  
11 class BarangController extends Controller  
12 {  
13     1 reference | 0 overrides | Windsurf: Refactor | Explain | Generate Function Comment | X  
14     public function index(): JsonResponse|mixed  
15     {  
16         $barang = BarangModel::all();  
17         return response()->json(data: [  
18             'success' => true,  
19             'data' => $barang,  
20         ]);  
21     }  
22 }
```



```
20
21 1 reference | 0 overrides | Windsurf: Refactor | Explain | Generate Function Comment | X
22 public function store(Request $request): JsonResponse|mixed
23 {
24     // Validasi
25     $validator = Validator::make(data: $request->all(), rules: [
26         'kategori_id' => 'required|exists:m_kategori,kategori_id',
27         'barang_kode' => 'required|unique:m_barang,barang_kode',
28         'barang_nama' => 'required',
29         'harga_beli' => 'required|numeric',
30         'harga_jual' => 'required|numeric',
31         'image' => 'nullable|image|mimes:jpeg,png,jpg,gif,svg|max:2048'
32     ]);
33
34     if ($validator->fails()) {
35         return response()->json(data: $validator->errors(), status: 422);
36     }
37
38     // Upload image kalau ada
39     $imageName = null;
40     if ($request->hasFile(key: 'image')) {
41         $imageName = $request->image->hashName();
42         $request->image->storeAs('public/barang', $imageName);
43     }
44
45     $barang = BarangModel::create(attributes: [
46         'kategori_id' => $request->kategori_id,
47         'barang_kode' => $request->barang_kode,
48         'barang_nama' => $request->barang_nama,
49         'harga_beli' => $request->harga_beli,
50         'harga_jual' => $request->harga_jual,
51         'image' => $imageName,
52     ]);
53
54     return response()->json(data: [
55         'success' => true,
56         'data' => $barang,
57     ], status: 201);
58 }
59
60 1 reference | 0 overrides | Windsurf: Refactor | Explain | Generate Function Comment | X
61 public function show($id): JsonResponse|mixed
62 {
63     $barang = BarangModel::find(id: $id);
64
65     if (!$barang) {
66         return response()->json(data: [
67             'success' => false,
68             'message' => 'Barang tidak ditemukan',
69         ], status: 404);
70     }
71
72     return response()->json(data: [
73         'success' => true,
74         'data' => $barang,
75     ]);
76 }
77
78 1 reference | 0 overrides | Windsurf: Refactor | Explain | Generate Function Comment | X
79 public function update(Request $request, $id): JsonResponse|mixed
80 {
81     $barang = BarangModel::find(id: $id);
82
83     if (!$barang) {
84         return response()->json(data: [
85             'success' => false,
86             'message' => 'Barang tidak ditemukan',
87         ], status: 404);
88     }
89
90     // Validasi
91     $validator = Validator::make(data: $request->all(), rules: [
92         '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(data: $validator->errors(), status: 422);
99     }
100
101     // Upload image kalau ada
102     if ($request->hasFile(key: 'image')) {
103         $imageName = $request->image->hashName();
104         $request->image->storeAs('public/barang', $imageName);
105         $barang->image = $imageName;
106     }
107
108     $barang->update(attributes: $request->except(keys: 'image')); // Kecuali image, karena sudah dihandle manual
109     $barang->save();
110
111     return response()->json(data: [
112         'success' => true,
113         'data' => $barang,
114     ]);
115 }
116
117 1 reference | 0 overrides | Windsurf Refactor | Explain | Generate Function Comment | X
118 public function destroy($id): JsonResponse|mixed
119 {
120     $barang = BarangModel::find(id: $id);
121
122     if (!$barang) {
123         return response()->json(data: [
124             'success' => false,
125             'message' => 'Barang tidak ditemukan',
126         ], status: 404);
127     }
128
129     $barang->delete();
130
131     return response()->json(data: [
132         'success' => true,
133         'message' => 'Barang berhasil dihapus',
134     ]);
135 }
```

4. Kemudian update kode untuk BarangModel nya

```
app > Models > BarangModel.php > PHP Intelephense > BarangModel > Fillable
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 17 references | 0 implementations | Windsurf Refactor | Explain
12 class BarangModel extends Model
13 {
14     0 references | 0 overrides | Windsurf Refactor | Explain | Generate Function Comment | X
15     public function getJWTIdentifier(): mixed{
16         return $this->getKey();
17     }
18
19     0 references | 0 overrides | Windsurf Refactor | Explain | Generate Function Comment | X
20     public function getJWTCustomClaims(): array{ return [];
21     }
22
23     use HasFactory;
24
25     0 references:
26     protected $table = "m_barang";
27
28     0 references:
29     protected $primaryKey = "barang_id";
30
31     0 references:
32     protected $fillable = [
33         'kategori_id',
34         'barang_kode',
35         'barang_nama',
36         'harga_beli',
37         'harga_jual',
38         'image' // tambahan
39     ];
40
41     0 references | 0 overrides | Windsurf Refactor | Explain | Generate Function Comment | X
42     public function kategori(): BelongsTo
43     {
44         return $this->belongsTo(related: KategoriModel::class, foreignKey: 'kategori_id', ownerKey: 'kategori_id');
45     }
46
47     0 references | 0 overrides | Windsurf Refactor | Explain | Generate Function Comment | X
48     public function stok(): HasOne
49     {
50         return $this->hasOne(related: StokModel::class, foreignKey: 'barang_id', localKey: 'barang_id');
51     }
52
53     0 references | 0 overrides | Windsurf Refactor | Explain | Generate Function Comment | X
54     protected function image(): Attribute
55     {
56         return Attribute::make(
57             get: fn ($image): string|UrlGenerator|null => $image ? url(path: './storage/barang/' . $image) : null
58         );
59     }
60 }
```



5. Kita coba untuk di postman nya dengan gambar, berikut isi tabel sebelum dimasukkan data baru

No	Kode Barang	Nama Barang	Harga Beli	Harga Jual	Kategori	Aksi
11	BRG011	Susu Ultra	6.500	7.500	Minuman	Detail Edit Hapus
12	BRG012	Celana Cihuy	65.000	75.000	Pakaian	Detail Edit Hapus
13	BRG013	Jelly Drink	5.000	6.000	Minuman	Detail Edit Hapus

Showing 11 to 13 of 13 entries

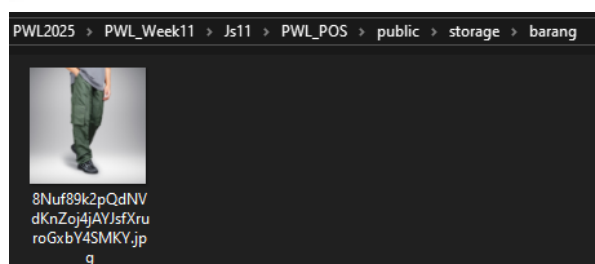
[Previous](#) [1](#) [2](#) [Next](#)

6. Masukkan data baru dengan gambar

```
{
  "success": true,
  "data": {
    "kategori_id": "5",
    "barang_kode": "BRG014",
    "barang_nama": "Celana Kargo",
    "harga_beli": "70000",
    "harga_jual": "100000",
    "image": "http://localhost/PWL2025/PWL_Week11/Js11/PWL_POS/public/storage/barang/8Nuf89k2pQdNVdKnZoj4jAYJsFXruoGxbY4SMKY.jpg",
    "updated_at": "2025-04-26T14:34:37.000000Z",
    "created_at": "2025-04-26T14:34:37.000000Z",
    "barang_id": 14
  }
}
```

7. Data baru masuk ke tabel dan foto disimpan di folder, serta namanya di database

14	BRG014	Celana Kargo	70.000	100.000	Pakaian	Detail Edit Hapus
----	--------	--------------	--------	---------	---------	---



*** Sekian, dan selamat belajar ***