

LAPORAN JOBSHEET 11

RESTFUL API 2

MATA KULIAH PEMROGRAMAN WEB LANJUT

Dosen Pengampu : Dimas Wahyu Wibowo, S.T., M.T.



Disusun oleh :

Dahniar Davina SIB-2A / 2341760023

JURUSAN TEKNOLOGI INFORMASI

PROGRAM STUDI D-IV SISTEM INFORMASI BISNIS

POLITEKNIK NEGERI MALANG

2025

A. 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_2025\Minggu11\Jobsheet11> php artisan make:migration add_image_to_m_user_table
```

```
INFO Migration [C:\laragon\www\PWL_2025\Minggu11\Jobsheet11\database\migrations\2025_04_30_085209_add_image_to_m_user_table.php] created successfully.
```

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

```
database > migrations > 2025_04_30_085209_add_image_to_m_user_table.php > class > down
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 };
```

4. Lakukan jalankan update migrasi dengan cara :

```
php artisan migrate
```

```
PS C:\laragon\www\PWL_2025\Minggu11\Jobsheet11> php artisan migrate
```

```
INFO Running migrations.
```

```
2025_04_30_085209_add_image_to_m_user_table ..... 105ms DONE
```

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

```

app > Models > UserModel.php > UserModel > $fillable
1  <?php
2
3  namespace App\Models;
4
5  use Illuminate\Database\Eloquent\Model;
6  use Tymon\JWTAuth\Contracts\JWTSubject;
7  use Illuminate\Database\Eloquent\Casts\Attribute;
8  use Illuminate\Foundation\Auth\User as Authenticatable;
9
10 class UserModel extends Authenticatable implements JWTSubject
11 {
12     public function getJWTIdentifier(){
13         return $this->getKey();
14     }
15     public function getJWTCustomClaims(){
16         return [];
17     }
18
19     protected $table = 'm_user';
20     protected $primaryKey = 'user_id';
21
22     protected $fillable = [
23         'username',
24         'nama',
25         'password',
26         'level_id',
27         'image' // tambahan
28     ];

```

6. Lakukan modifikasi pada Controllers/Api/RegisterController

```

app > Http > Controllers > Api > RegisterController.php > ...
1  <?php
2
3  namespace App\Http\Controllers\Api;
4
5  use App\Models\UserModel;
6  use App\Http\Controllers\Controller;
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         // set validation
15         $validator = Validator::make($request->all(), [
16             'username' => 'required',
17             'nama' => 'required',
18             'password' => 'required|min:5|confirmed',
19             'level_id' => 'required',
20             'image' => 'required'
21         ]);
22
23         // if validation fails
24         if($validator->fails()){
25             return response()->json($validator->errors(), 422);
26         }
27

```

7. Anda dapat menambahkan detail untuk spesifikasi image pada validator

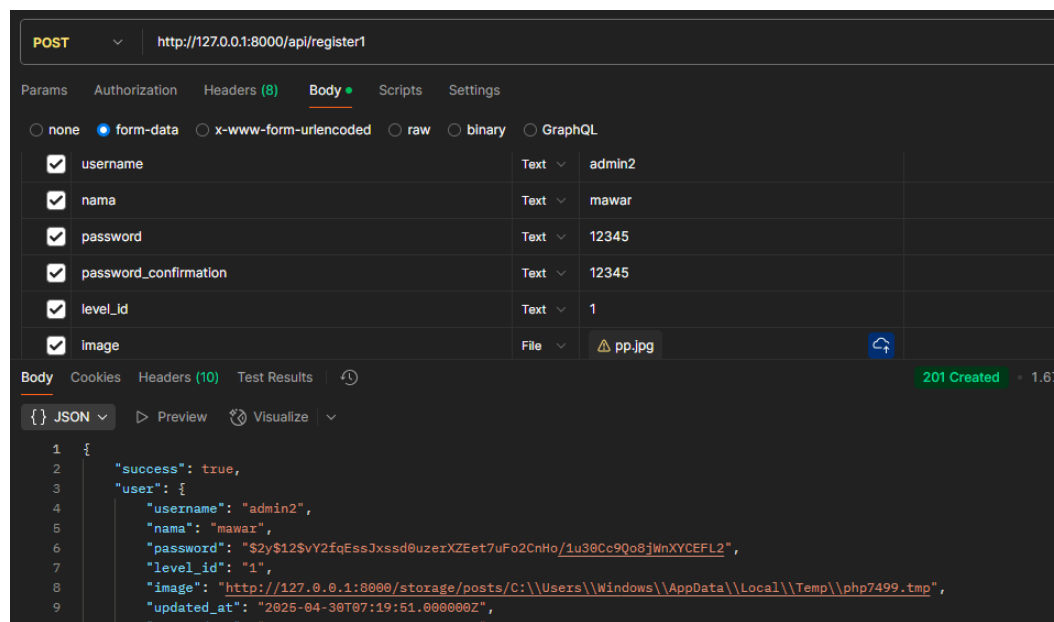
```
'image' => 'required|image|mimes:jpeg,png,jpg,gif,svg|max:2048'
```

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

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

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



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

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

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

- Jika menggunakan `'image' => $request->image` maka file langsung mengambil data yang di-upload tanpa diproses apa pun. Yang tersimpan ke database bukan nama file (contohnya gambar.jpg), tapi object file pathnya.
- Sedangkan Jika menggunakan `'image' => $request->image->hashName()` maka akan mengambil file → simpan dulu ke storage (storage/app/public/users) → baru simpan nama filenya ke database. File gambarnya akan masuk ke folder storage/app/public/users. Di database, hanya akan menyimpan nama file-nya aja, contoh ProfilePicture.jpg, jadi lebih ringan dan aman.

B. 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 diinputkan.

➤ M_barang

1. Buat migration pada table m_barang

```
INFO Migration [C:\laragon\www\PWL_2025\Minggu11\Jobsheet11\database\Migrations\2025_04_30_145930_add_image_to_m_barang_table.php] created successfully.
```

2. Ubah isi filenya

```
database > migrations > 2025_04_30_145930_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      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 }
```

3. Modifikasi isi file pada BarangModel.php

```
app > Models > BarangModel.php > BarangModel > getJWTIdentifier
1  <?php
2  namespace App\Models;
3
4  use Illuminate\Database\Eloquent\Factories\HasFactory;
5  use Illuminate\Database\Eloquent\Model;
6  use Illuminate\Database\Eloquent\Relations\BelongsTo;
7  use Illuminate\Database\Eloquent\Casts\Attribute;
8  use Tymon\JWTAuth\Contracts\JWTSubject;
9
10 class BarangModel extends Model
11 {
12     public function getJWTIdentifier(){
13         return $this->getKey();
14     }
15     public function getJWTCustomClaims(){ return []; }
16 }
17 use HasFactory;
```

4. Modifikasi isi file pada Api/BarangController.php

```
app > Http > Controllers > Api > BarangController.php > BarangController > update
1  <?php
2  namespace App\Http\Controllers\API;
3
4  use App\Http\Controllers\Controller;
5  use Illuminate\Http\Request;
6  use App\Models\BarangModel;
7  use Illuminate\Support\Facades\Validator;
8
9  class BarangController extends Controller
10 {
11     public function index()
12     {
13         $barang = BarangModel::all();
14         return response()->json([
15             'success' => true,
16             'data' => $barang,
17         ]);
18     }
19     public function store(Request $request)
20     {
21         // Validasi
22         $validator = Validator::make($request->all(), [
23             'kategori_id' => 'required|exists:m_kategori,kategori_id',
24             'barang_kode' => 'required|unique:m_barang,barang_kode',
25             'barang_nama' => 'required',
26             'harga_beli' => 'required|numeric',
27             'harga_jual' => 'required|numeric',
28             'image' => 'nullable|image|mimes:jpeg,png,jpg,gif,svg|max:2048',
29         ]);
30     }
31 }
```

5. Proses pada Postman

POST http://127.0.0.1:8000/api/barangs?kategori_id&barang_kc... Send

Params Auth Headers (8) Body Scripts Settings Cookies

form-data

<input checked="" type="checkbox"/>	kategori_id	Text	9
<input checked="" type="checkbox"/>	barang_kode	Text	BRG022
<input checked="" type="checkbox"/>	barang_nama	Text	Sunscreen Gel Azarine H...
<input checked="" type="checkbox"/>	harga_beli	Text	26000
<input checked="" type="checkbox"/>	harga_jual	Text	35000
<input checked="" type="checkbox"/>	image	File	azarine.jpg

Body 201 Created 1.14 s 683 B Save Response

JSON Preview Visualize

```
1 {
2   "success": true,
3   "data": {
4     "kategori_id": "9",
5     "barang_kode": "BRG022",
6     "barang_nama": "Sunscreen Gel Azarine Hydrasoothe SPF 45 PA
7     "harga_beli": "26000",
8     "harga_jual": "35000",
```

➤ M_transaksi

1. Buat Migration pada table t_penjualan

```
PS C:\laragon\www\PMI_2025\Minggu11\Jobsheet11> php artisan make:migration add_image_to_t_penjualan_table
[INFO] Migration [C:\laragon\www\PMI_2025\Minggu11\Jobsheet11\database\migrations\2025_03_214345_add_image_to_t_penjualan_table.php] created successfully.
```

2. Ubah isi filenya

```
<?php

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

return new class extends Migration
{
    public function up(): void
    {
        Schema::table('t_penjualan', function (Blueprint $table) {
            $table->string('image')->nullable()->after('penjualan_tanggal');
        });
    }

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

3. Buat API/PenjualanController

```
PS C:\laragon\www\PMI_2025\Minggu11\Jobsheet11> php artisan make:migration add_image_
ke:controller Api/PenjualanController
[INFO] Controller [C:\laragon\www\PMI_2025\Minggu11\Jobsheet11\app\Http\Controllers\Api\PenjualanController.php] created successfully.
```

4. Tambahkan isi filenya

```
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         $data = PenjualanModel::with(['user', 'details.barang'])->get();
16         return response()->json($data);
17     }
18
19     public function show($transaksi)
20     {
21         $penjualan = PenjualanModel::with(['user', 'details.barang'])->findOrFail($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:m_user,user_id',
29             'pembeli' => 'required|string|max:100',
30             'penjualan_kode' => 'required|string|unique:t_penjualan,penjualan_kode',
```

5. Modifikasi isi file pada PenjualanModel.php

```
app > Models > PenjualanModel.php > PenjualanModel > image
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\Casts\Attribute;
12 use Tymon\JWTAuth\Contracts\JWTSubject;
13 use Illuminate\Foundation\Auth\User as Authenticatable;
14
15 class PenjualanModel extends Model
16 {
17     public function getJWTIdentifier(){
18         return $this->getKey();
19     }
20
21     public function getJWTCustomClaims(){
22         return [];
23     }
24 }
```

6. Modifikasi tambahan pada route/api

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

Route::get('transaksi', [PenjualanController::class, 'index']);
Route::get('transaksi/{transaksi}', [PenjualanController::class, 'show']);
Route::post('transaksi', [PenjualanController::class, 'store']);
Route::post('transaksi/{transaksi}', [PenjualanController::class, 'update']);
Route::delete('transaksi/{transaksi}', [PenjualanController::class, 'destroy']);
```

7. Proses pada Postman

The screenshot shows the Postman interface for a POST request to `http://127.0.0.1:8000/api/transaksi?`. The request body is a form-data payload with the following fields:

Field	Type	Value
pembeli	Text	Mariana Sungkar
penjualan_kode	Text	TRX018
penjualan_tanggal	Text	2025-05-03 22:09:00
image	File	struk-belanja.jpeg
barang_id	Text	9

The response is a 201 Created status with a response time of 1.96 s and a body size of 771 B. The JSON response is as follows:

```
{
  "penjualan": {
    "barang_id": "9",
    "user_id": "2",
    "pembeli": "Mariana Sungkar",
    "penjualan_kode": "TRX018",
    "penjualan_tanggal": "2025-05-03 22:09:00",
    "image": "http://127.0.0.1:8000/storage/transaksi/1746285937.jpg",
    "updated_at": "2025-05-03T15:25:38.000000Z",
    "created_at": "2025-05-03T15:25:37.000000Z",
    "penjualan_id": 19
  }
}
```