

JOBSHEET 11

LAPORAN TUGAS

**Diajukan untuk Memenuhi Tugas
Pemograman Web Lanjut**

Jurusan:

TEKNOLOGI INFORMASI

Program Studi:

TEKNIK INFORMATIKA

Disusun oleh

1. Afrizal Dwi Septian (2241720122) / 01 / 2H



**POLITEKNIK NEGERI MALANG
TEKNIK INFORMATIKA**

2024



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.
```

```
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_05_15_055039_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');
        });
    }
};
```



```
database > migrations > 2024_05_15_055039_add_image_to_m_user_table.php > ...
1  <?php
2  use Illuminate\Database\Migrations\Migration;
3  use Illuminate\Database\Schema\Blueprint;
4  use Illuminate\Support\Facades\Schema;
5
6  return new class extends Migration
7  {
8      /**
9       * Run the migrations.
10      */
11      public function up(): void
12      {
13          Schema::table('m_user', callback: function (Blueprint $table) {
14              $table->string('image');
15          });
16      }
17
18      /**
19       * Reverse the migrations.
20      */
21      public function down(): void
22      {
23          Schema::table('m_user', callback: function (Blueprint $table) {
24              $table->dropColumn('image');
25          });
26      }
27  };
28
```

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

```
C:\laragon\www\PWL_POS>php artisan migrate
INFO Running migrations.
2024_05_15_055039_add_image_to_m_user_table ..... 124ms 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),  
        );  
    }  
}
```



```
app > Models > UserModel.php > ...
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 App\Models\LevelModel;
9  use Illuminate\Foundation\Auth\User as userAuthentication;
10 use Tymon\JWTAuth\Contracts\JWTSubject;
11 use Illuminate\Database\Eloquent\Casts\Attribute;
12
13 class UserModel extends userAuthentication implements JWTSubject
14 {
15     public function getJWTIdentifier()
16     {
17         return $this->getKey();
18     }
19     public function getJWTCustomClaims(){
20         return [];
21     }
22
23     protected $table = 'm_user';
24     protected $primaryKey = 'user_id';
25     protected $fillable = [
26         'username',
27         'nama',
28         'password',
29         'level_id',
30         'image'//tambahan
31     ];
32     public function level()
33     {
34         return $this->belongsTo(related: LevelModel::class, foreignKey: 'level_id', ownerKey: 'level_id');
35     }
36     protected function image(): Attribute
37     {
38         return Attribute::make(
39             get: fn ($image) => url(path: '/storage/posts/' . $image),
40             );
41     }
42 }
```

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);          }

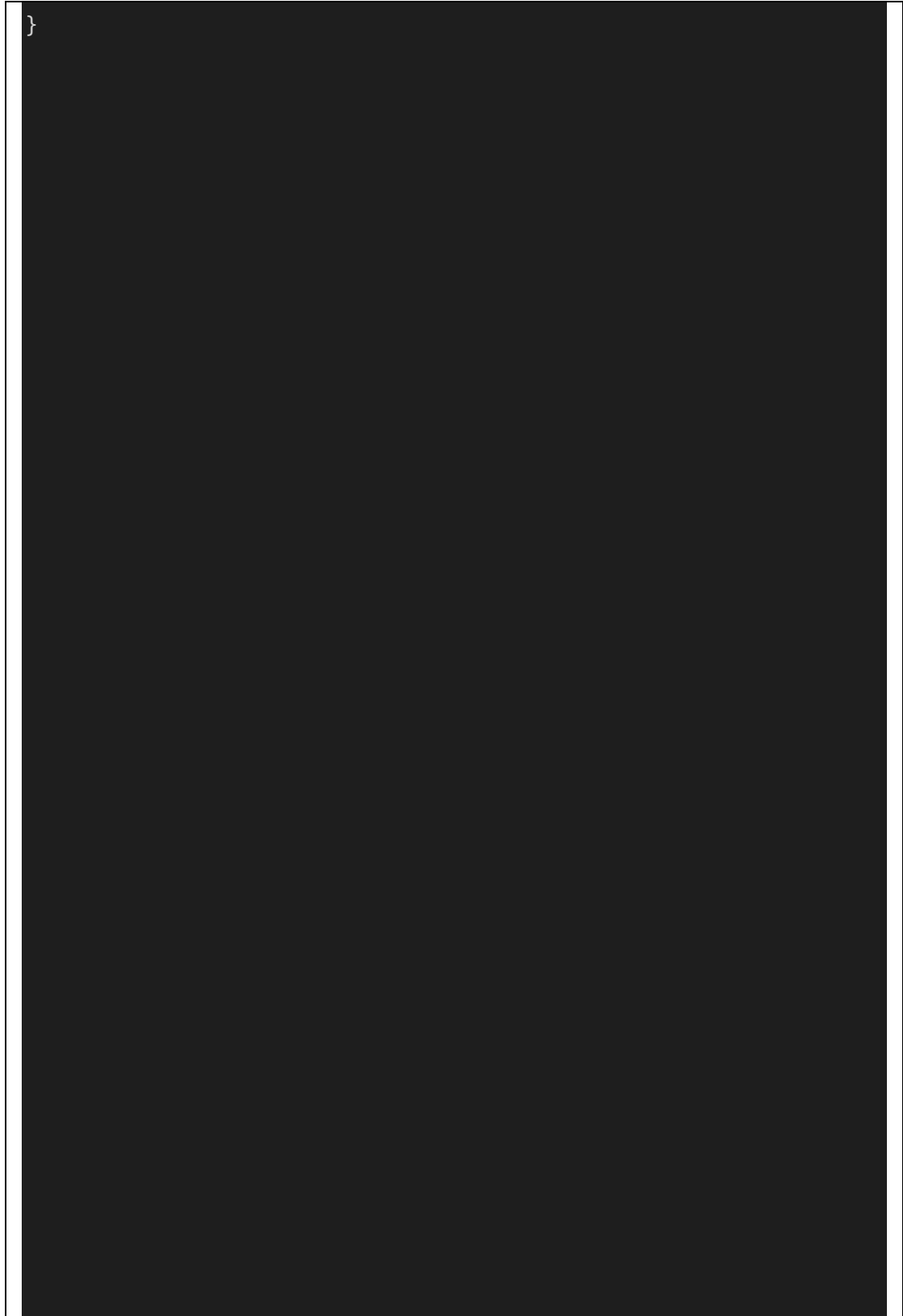
        //create user
        $user = UserModel::create([
            'username' => $request->username,
            'nama' => $request->nama,
            'password' => bcrypt($request->password),
            'level_id' => $request->level_id,
            'image' => $request->image
        ]);

        //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);
    }
}
```




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>





```
app > Http > Controllers > Api > RegisterController.php > RegisterController
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         $validator = Validator::make($request->all(), $rules: [
15             'username' => 'required',
16             'nama' => 'required',
17             'password' => 'required|min:5|confirmed',
18             'level_id' => 'required',
19             'image' => 'required|image|mimes:jpeg,png,jpg,gif,svg|max:2048'
20         ]);
21
22         if ($validator->fails()) {
23             return response()->json($validator->errors(), $status: 422);
24         }
25
26         $user = UserModel::create([
27             'username' => $request->username,
28             'nama' => $request->nama,
29             'password' => bcrypt($request->password),
30             'level_id' => $request->level_id,
31             'image' => $request->image
32         ]);
33
34         if($user){ return response()->json($data: [
35             'success' => true,
36             'user' => $user,
37         ], $status: 201);
38     }
39
40     return response()->json($data: [
41         'status' => false,
42     ], $status: 409);
43 }
```

7. Anda dapat menambahkan detail untuk spesifikasi image pada validator

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

```
public function __invoke(Request $request)
{
    $validator = Validator::make($request->all(), $rules: [
        'username' => 'required',
        'nama' => 'required',
        'password' => 'required|min:5|confirmed',
        'level_id' => 'required',
        'image' => 'required|image|mimes:jpeg,png,jpg,gif,svg|max:2048'
    ]);
```

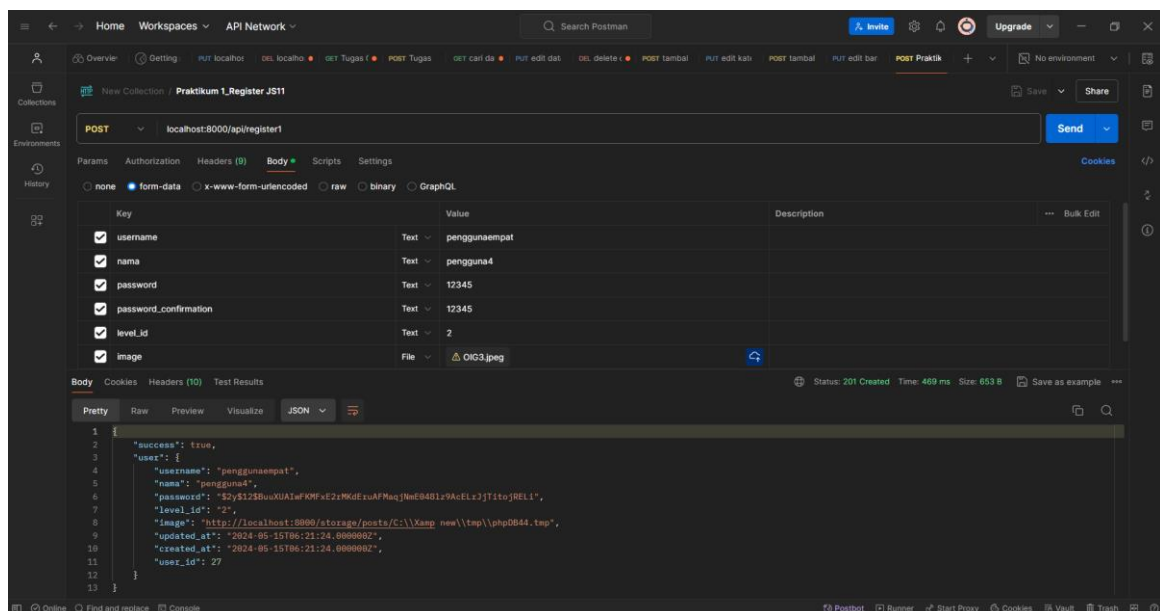
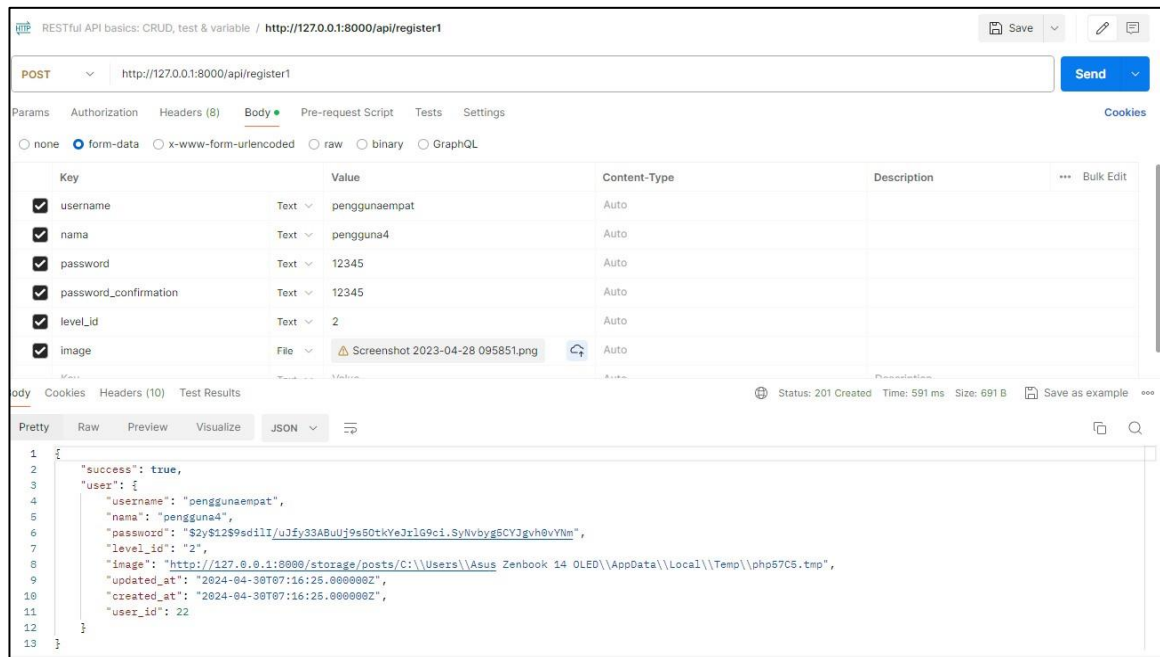
8. Ubah atau tambahkan register1 pada routes/api.php

```
Route::post('/register1', App\Http\Controllers\Api\RegisterController::class)->name('register1');
Route::post($uri: '/register1', $action: App\Http\Controllers\Api\RegisterController::class)->name($name: 'register1');
```



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>

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



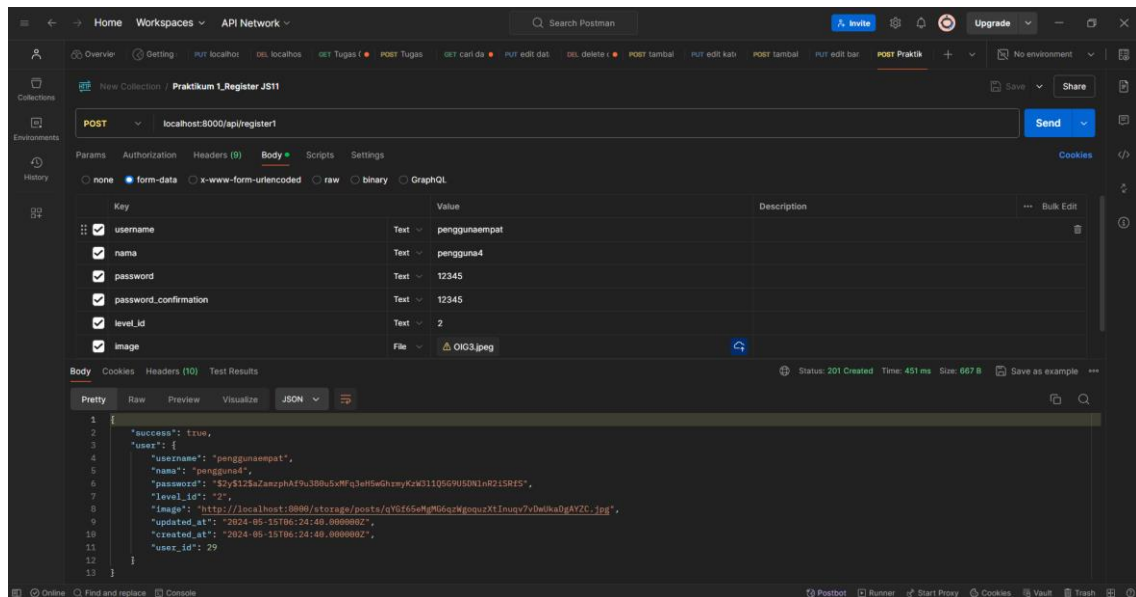
8. Pada Controllers/Api/RegisterController bagian create user ganti dengan

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



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

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



Jawab: Perbedaannya terletak pada nama gambarnya, pada saat pertama nama gambarnya disimpan dengan nama asli sedangkan setelah merubah kodenya seperti diatas nama dari gambar yang kita upload menjadi acak karena hashName.

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.

No	Screenshoot
1	<p>Kode</p> <pre>C:\laragon\www\PWL_POS>php artisan make:migration add_image_to_m_barang_table</pre> <pre>INFO Migration [C:\laragon\www\PWL_POS\database\Migrations\2024_05_15_064652_add_image_to_m_barang_table.php] created successfully.</pre> <pre>C:\laragon\www\PWL_POS>php artisan migrate</pre> <pre>INFO Running migrations.</pre> <pre>2024_05_15_064652_add_image_to_m_barang_table 87ms DONE</pre>



```
database > migrations > 2024_05_15_064652_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', callback: 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_barang', callback: function (Blueprint $table) {
25              $table->dropColumn('image');
26          });
27      }
28  };
```

```
app > Models > BarangModel.php > ...
```

```
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\KategoriModel;
10 use Illuminate\Database\Eloquent\casts\Attribute;
11
12 class BarangModel extends Model
13 {
14     use HasFactory;
15
16     protected $table = 'm_barang';
17     protected $primaryKey = 'barang_id';
18
19     protected $fillable = ['kategori_id', 'barang_kode', 'barang_nama', 'harga_beli', 'harga_jual', 'image'];
20
21     public function kategori(): BelongsTo
22     {
23         return $this->belongsTo(related: KategoriModel::class, foreignKey: 'kategori_id', ownerKey: 'kategori_id');
24     }
25
26     protected function image(): Attribute
27     {
28         return Attribute::make(
29             get: fn ($image) => url(path: '/storage/posts/' . $image),
30         );
31     }
32 }
33
```



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>

```
45 public function update(Request $request, BarangModel $barang)
46 {
47     $request->validate([
48         'barang_kode' => 'required|string|min:3|unique:m_barang,barang_kode',
49         'barang_nama' => 'required|string|max:100',
50         'harga_beli' => 'required|integer',
51         'harga_jual' => 'required|integer',
52         'kategori_id' => 'required|integer',
53         'image' => 'required|image|mimes:jpeg,png,jpg,gif,svg|max:2048'
54     ]);
55
56     $barang->update([
57         'barang_kode' => $request->barang_kode,
58         'barang_nama' => $request->barang_nama,
59         'harga_beli' => $request->harga_beli,
60         'harga_jual' => $request->harga_jual,
61         'kategori_id' => $request->kategori_id,
62         'image' => $request->image->hashName()
63     ]);
64
65     return BarangModel::find($barang);
66 }
67
68 public function destroy(BarangModel $barang)
69 {
70     $barang->delete();
71     return response()->json([
72         'success' => true,
73         'message' => 'Data Berhasil Dihapus'
74     ]);
75 }
76 }
```

```
app > Http > Controllers > Api > BarangController.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
9 class BarangController extends Controller
10 {
11     public function index()
12     {
13         return BarangModel::all();
14     }
15
16     public function store(Request $request)
17     {
18         $request->validate([
19             'barang_kode' => 'required|string|min:3|unique:m_barang,barang_kode',
20             'barang_nama' => 'required|string|max:100',
21             'harga_beli' => 'required|integer',
22             'harga_jual' => 'required|integer',
23             'kategori_id' => 'required|integer',
24             'image' => 'required|image|mimes:jpeg,png,jpg,gif,svg|max:2048'
25         ]);
26
27         $barang = BarangModel::create([
28             'barang_kode' => $request->barang_kode,
29             'barang_nama' => $request->barang_nama,
30             'harga_beli' => $request->harga_beli,
31             'harga_jual' => $request->harga_jual,
32             'kategori_id' => $request->kategori_id,
33             'image' => $request->image->hashName()
34         ]);
35
36         return response()->json([
37             'data' => $barang,
38             'status' => 201
39         ]);
40
41     public function show(BarangModel $barang)
42     {
43         return response()->json([
44             'data' => $barang,
45             'status' => 200
46         ]);
47     }
48 }
```



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>

2

Hasil

The image displays two screenshots of the Postman API client interface. The top screenshot shows a POST request to `localhost:8000/api/barang` with a status of 201 Created. The request body is a JSON object containing details for a new item. The bottom screenshot shows a GET request to `localhost:8000/api/barang/14` with a status of 200 OK, returning the details of the item with ID 14.

POST Request Details:

- URL: `localhost:8000/api/barang`
- Method: POST
- Body (form-data):
 - `kategori_id`: 2
 - `barang_kode`: BRG12
 - `barang_nama`: KOKO Crunch Duo
 - `harga_beli`: 20000
 - `harga_jual`: 50000
 - `image`: `kokoDuo170g_0.png`
- Status: 201 Created
- Time: 709 ms
- Size: 624 B

GET Request Details:

- URL: `localhost:8000/api/barang/14`
- Method: GET
- Status: 200 OK
- Time: 518 ms
- Size: 613 B

Response Body (JSON):

```
{
  "barang_id": 14,
  "kategori_id": 2,
  "barang_kode": "BRG12",
  "barang_nama": "KOKO Crunch Duo",
  "harga_beli": 20000,
  "harga_jual": 50000,
  "created_at": "2024-05-15T06:56:06.000000Z",
  "updated_at": "2024-05-15T06:56:06.000000Z",
  "image": "http://localhost:8000/storage/posts/NwobczlTgvbF8d9040c0w055KHZLSlaylcvf1.png"
}
```