LAPORAN PRAKTIKUM

PERANCANGAN DAN PEMROGRAMAN WEB

MODUL 12

(LARAVEL: MIGRATION)

Oleh:

(Rengganis Tantri Pramudita) (2311104065)

PROGRAM STUDI S1 REKAYASA PERANGKAT LUNAK

DIREKTORAT KAMPUS PURWOKERTO

UNIVERSITAS TELKOM

2025

BAB I

PENDAHULUAN

A. Dasar Teori

Laravel merupakan framework PHP yang menyediakan berbagai fitur untuk mempermudah pengelolaan database, salah satunya melalui DB Facade. DB Facade memungkinkan pengembang untuk menjalankan perintah SQL mentah (Raw SQL Queries) secara langsung di dalam aplikasi Laravel. Dengan metode ini, pengembang dapat menuliskan perintah SQL seperti INSERT, SELECT, UPDATE, dan DELETE sebagaimana pada MySQL biasa, namun tetap berada dalam lingkungan Laravel yang terintegrasi dengan konfigurasi database dan sistem routing.

B. Tujuan

Tujuan dari praktikum ini adalah untuk memahami cara menyimpan data donasi tunai ke dalam database menggunakan metode Raw SQL Queries pada framework Laravel. Selain itu, praktikum ini bertujuan melatih mahasiswa agar mampu menghubungkan route, controller, dan database, serta memahami alur penyimpanan data dari aplikasi Laravel hingga tersimpan dengan benar di dalam tabel database.

BAB II

HASIL

A. GUIDED (Praktikum Terbimbing)

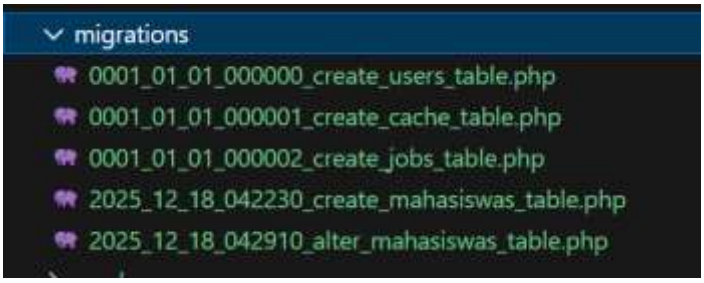
- Konfigurasi database laravel

  Sebelum menjalankan migration dan database, jalankan Apache dan MySQL di XAMPP.

  Silahkan buka file .env kemudian lakukan pengaturan database seperti

```
22
23  DB_CONNECTION=mysql
24  DB_HOST=127.0.0.1
25  DB_PORT=3306
26  DB_DATABASE=2311104065_rengganis
27  DB_USERNAME=root
28  DB_PASSWORD=
```

- File migration bawaan laravel



  Penjelasan:

  berfungsi sebagai kontrol versi untuk struktur database. Setiap file berisi kode PHP untuk membuat atau mengubah tabel; terlihat ada migrasi standar untuk sistem (users, cache, jobs) serta migrasi khusus untuk tabel mahasiswas. Penamaan file menggunakan format timestamp di bagian depan untuk memastikan urutan eksekusi yang tepat, mulai dari pembuatan tabel hingga proses perubahan struktur (*alter*) yang dilakukan kemudian.

- Implementasi table migration

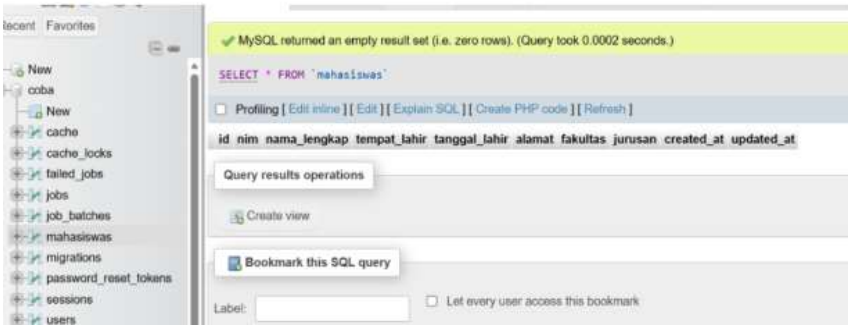
```php
<?php

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

class CreateMahasiswasTable extends Migration
{
    /**
     * Run the migrations.
     */
    public function up()
    {
        Schema::create('mahasiswas', function (Blueprint $table) {
            $table->id();
            $table->char('nim', 8)->unique();
            $table->string('nama');
            $table->string('tempat_lahir');
            $table->date('tanggal_lahir');
            $table->string('fakultas');
            $table->string('jurusan');
            $table->decimal('ipk', 3, 2)->default(1.00);
            $table->timestamps();
        });
    }

    /**
     * Reverse the migrations.
     */
    public function down()
```

```
    {
        Schema::dropIfExists('mahasiswas');
    }
}
```

Penjelasan:

Kode tersebut merupakan file migrasi Laravel untuk membuat tabel mahasiswas dengan struktur data yang lengkap, mulai dari kolom identitas hingga informasi akademik. Metode up() mendefinisikan kolom-kolom spesifik seperti nim yang bersifat unik, ipk dengan format desimal, serta kolom pendukung seperti timestamps untuk pencatatan waktu, sementara metode down() bertugas menghapus tabel tersebut jika dilakukan *rollback*. Secara keseluruhan, migrasi ini memastikan skema database dapat dibuat secara otomatis dan konsisten dengan tipe data yang tepat, seperti penggunaan char untuk panjang karakter tetap pada NIM dan date untuk tanggal lahir.

Outputnya



- Migration Rollback

  Pada saat menjalankan perintah php artisan migrate, file migration akan di eksekusi secara berurutan sesuai tanggal timestamp. File dengan timestamp terbaru akan dijalankan paling akhir. Bisa menggunakan perintah *php artisan migrate:rollback --step=1*. Perintah ini akan me-rollback 1 file migration terakhir dan juga menghapus tabelnya.

- Modifikasi file migration

  Perubahan pada kolom nim dan ipk

```php
Schema::create('mahasiswas', function (Blueprint $table) {
        $table->id();
        $table->char('nim',8)->unique();
        $table->string('nama');
        $table->string('tempat_lahir');
        $table->date('tanggal_lahir');
        $table->string('fakultas');
        $table->string('jurusan');
        $table->decimal('ipk',3,2)->default(1.00);
        $table->timestamps();
    });
}
```

- Alter table migration

Penjelasan:

proses instalasi paket **doctrine/dbal** melalui Composer yang diperlukan agar Laravel dapat melakukan **alter table** atau memodifikasi kolom pada migrasi yang sudah ada. Dalam konteks file Anda, paket ini diinstal agar file alter_mahasiswas_table.php dapat menjalankan fungsi perubahan struktur, seperti mengubah tipe data atau mengganti nama kolom, tanpa menghapus data yang sudah ada. Meskipun muncul peringatan teknis terkait modul PHP, proses tersebut berhasil memasang dependensi yang dibutuhkan untuk mengelola skema database secara lebih fleksibel.

Implementasi

```
<?php


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

```
return new class extends Migration {
    public function up()
    {
        Schema::table('mahasiswas', function (Blueprint $table) {
            $table->renameColumn('nama', 'nama_lengkap');
            $table->text('alamat')->after('tanggal_lahir');
            $table->dropColumn('ipk');
        });
    }

    public function down()
    {
        Schema::table('mahasiswas', function (Blueprint $table) {
            $table->renameColumn('nama_lengkap', 'nama');
            $table->dropColumn('alamat');
            $table->decimal('ipk', 3, 2)->default(1.00);
        });
    }
};
```

Penjelasan:

merupakan file migrasi alter table yang berfungsi untuk memodifikasi struktur tabel mahasiswas yang sudah ada sebelumnya. Di dalam metode up(), dilakukan tiga perubahan utama yaitu mengubah nama kolom nama menjadi nama_lengkap, menambahkan kolom alamat dengan tipe data teks tepat setelah kolom tanggal_lahir, serta menghapus kolom ipk. Sebaliknya, metode down() berisi instruksi untuk membatalkan semua perubahan tersebut (mengembalikan nama kolom, menghapus kolom alamat, dan menambahkan kembali kolom IPK) guna memastikan integritas database tetap terjaga jika Anda melakukan *rollback* migrasi.

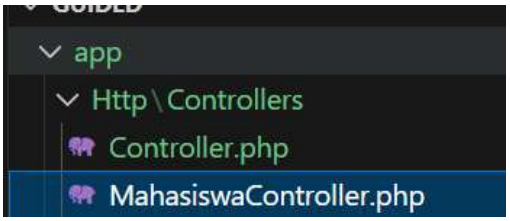
- Membuat route untuk crud

```php
<?php

use Illuminate\Support\Facades\Route;
use App\Http\Controllers\MahasiswaController;

Route::get('/', function () {
    return view('welcome');
});

Route::get('/insert-data', [MahasiswaController::class,'insertData']);
Route::get('/select-data', [MahasiswaController::class,'selectData']);
Route::get('/update-data', [MahasiswaController::class,'updateData']);
Route::get('/delete-data', [MahasiswaController::class,'deleteData']);
```

Penjelasan:

pada file web.php, Anda telah mendefinisikan rute-rute khusus yang menghubungkan URL seperti /insert-data, /select-data, /update-data, dan /delete-data ke metode-metode di dalam MahasiswaController. Dengan mengintegrasikan kemampuan manipulasi skema database dan sistem routing ini, aplikasi kini siap untuk menjalankan fungsi pengelolaan data mahasiswa secara dinamis melalui browser.

- Membuat mahasiswa controller



- Inseert Data menggunakan Query Builder

Penjelasan:

Fungsi ini menggunakan perintah insert untuk memasukkan satu baris data mahasiswa baru ke dalam tabel mahasiswas dengan mendefinisikan setiap kolom dalam bentuk array. Hasil dari perintah ini berupa nilai boolean (true/false) yang menunjukkan apakah data berhasil disimpan ke database.

- Update Data menggunakan Query Builder

```
        public function updateData()
        {
            $result = DB::table('mahasiswas')
                ->where('nim', '2211104065')
                ->update(['nama_lengkap' => 'Rengganis Putri']);
            dump($result);
}
```

Penjelasan:

Fungsi ini bertujuan untuk mengubah data yang sudah ada di database dengan mencari baris berdasarkan kolom nim menggunakan metode where. Setelah baris

ditemukan, metode update akan mengganti nilai pada kolom nama_lengkap tanpa menghapus data kolom lainnya.

- Delete Data menggunakan Query Builder

```
    public function deleteData()
    {
        $result = DB::table('mahasiswas')
            ->where('nim', '2211104065')
            ->delete();
        dump($result);
}
```

Penjelasan:

Fungsi ini digunakan untuk menghapus data tertentu dari tabel dengan memberikan kondisi spesifik melalui metode where. Penggunaan where sangat penting di sini agar perintah delete hanya menghapus data mahasiswa yang dimaksud dan tidak menghapus seluruh isi tabel secara tidak sengaja.

- Select Data menggunakan Query Builder

```
    public function selectData()
    {
        $mahasiswa = DB::table('mahasiswas')->get();
        dump($mahasiswa);
}
```

Penjelasan:

Fungsi ini menggunakan metode get untuk menarik atau menampilkan seluruh baris data yang tersimpan di dalam tabel mahasiswas. Data yang diambil biasanya dikembalikan dalam bentuk *Collection* yang berisi objek-objek data untuk kemudian ditampilkan ke layar atau dikirim ke tampilan (view).

- Input Data menggunakan Eloquent ORM

Modifikasi isi \app\Models\Mahasiswa.php seperti berikut:

```
```

```php
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;
use Illuminate\Support\Facades\DB;
// Import Model Mahasiswa agar bisa digunakan
use App\Models\Mahasiswa;

class MahasiswaController extends Controller
{
    public function index()
    {
        return "Index untuk mahasiswa";
    }

    // Menambah data menggunakan Eloquent (sesuai gambar terakhir)
    public function insertEloquent()
    {
        $mhs = Mahasiswa::create([
            'nim'          => '20104080',
            'nama_lengkap'  => 'Rizky Ananda',
            'tempat_lahir'  => 'Malang',
            'tanggal_lahir' => '2002-09-12',
            'alamat'       => 'Jl. Kenanga No. 7',
            'fakultas'     => 'Fakultas Informatika',
            'jurusan'      => 'Software Engineering',
        ]);

        dump($mhs);
    }
```

```
// Menambah data menggunakan Query Builder
public function insertData()
{
    $result = DB::table('mahasiswas')->insert([
        'nim'          => '2211104065',
        'nama_lengkap'  => 'Rengganis',
        'tanggal_lahir' => '1998-09-28',
        'tempat_lahir'  => 'Purwokerto',
        'alamat'        => 'Arca',
        'fakultas'      => 'Informatika',
        'jurusan'       => 'RPL',
    ]);

    dump($result);
}

// Menampilkan data
public function selectData()
{
    $allMahasiswa = DB::table('mahasiswas')->get();
    dump($allMahasiswa);
}

// Mengupdate data
public function updateData()
{
    $result = DB::table('mahasiswas')
        ->where('nim', '2211104065')
        ->update(['nama_lengkap' => 'Rengganis Update']);

    dump($result);
}
```

```
        // Menghapus data
        public function deleteData()
        {
            $result = DB::table('mahasiswas')
                ->where('nim', '2211104065')
                ->delete();

            dump($result);
        }
}
```

Penjelasasn:

Kode tersebut menunjukkan penggunaan Eloquent ORM di Laravel untuk menambahkan data ke dalam tabel mahasiswas melalui Model Mahasiswa. Dengan menggunakan metode Mahasiswa::create(), Anda dapat memasukkan data secara lebih efisien dan rapi dibandingkan Query Builder, karena Laravel secara otomatis memetakan array berisi NIM, nama, dan detail lainnya ke kolom-kolom database yang sesuai. Setelah proses eksekusi, fungsi dump($mhs) digunakan untuk menampilkan objek data yang baru saja disimpan guna memastikan bahwa operasi *insert* berhasil dan atribut seperti id atau timestamps telah terisi secara otomatis oleh sistem.

B. UNGUIDED (Tugas Mandiri)

12.1

Buatlah file migration untuk database yang akan digunakan pada tugas besar di kelas teori.

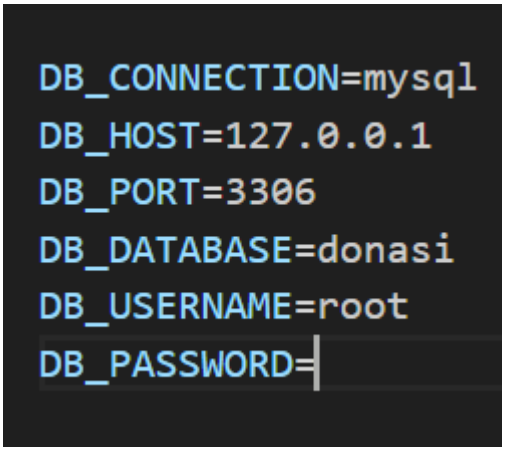
- Buat project Laravel dengan nama unguided

- Buat database dengan nama donasi di phpMyAdmin

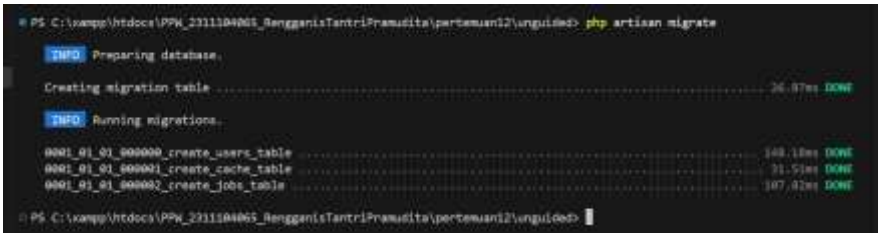
Database	Collation	Action
akademic	utf8mb4_general_ci	Check privileges
akademik	utf8mb4_general_ci	Check privileges
donasi	utf8mb4_general_ci	Check privileges
ecommerce	utf8mb4_general_ci	Check privileges
information_schema	utf8_general_ci	Check privileges
login_sistem	utf8mb4_general_ci	Check privileges
mysql	utf8mb4_general_ci	Check privileges
panti_wredha_bdk	utf8mb4_general_ci	Check privileges
performance_schema	utf8_general_ci	Check privileges
phpmyadmin	utf8_bin	Check privileges
test	latin1_swedish_ci	Check privileges

Total: 11

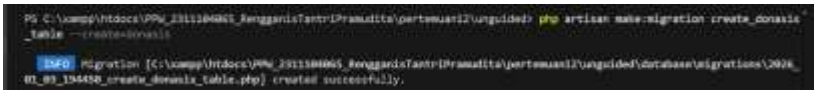
Ubah di file .env menjadi seperti ini



Lakukan *php artisan migrate*



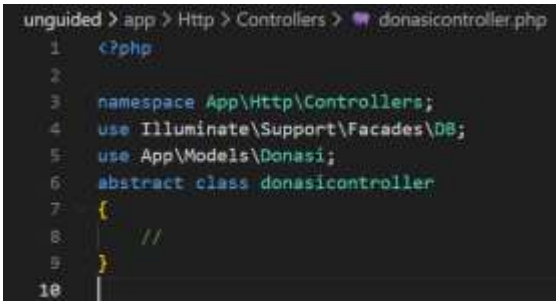
- Buat tabel donasi dengan *php artisan make:migration create_donasis_table --create=donasis*



Isi

Lalu lakukan *php artisan migrate*

- Buat donasi controller



12.2

Buatlah fungsi-fungsi berikut untuk menyelesaikan tugas besar di matakuliah teori:

1. Buat fungsi untuk insert data menggunakan raw SQL Queries
   - Isi DonasiController.php dengan ini

```
<?php

namespace App\Http\Controllers;

use Illuminate\Support\Facades\DB;
use Illuminate\Http\Request;

class DonasiController extends Controller
{
   public function insertSql()
   {
      $result = DB::insert("
         INSERT INTO donasis
            (nama_donatur, email, jumlah_donasi, metode_pembayaran, pesan,
tanggal_donasi)
         VALUES
         ('Rengganis Tantri', 'rengganis@gmail.com', 50000,
          'Tunai', 'Semoga bermanfaat', CURDATE())
      ");

      dump($result);
   }
}
```

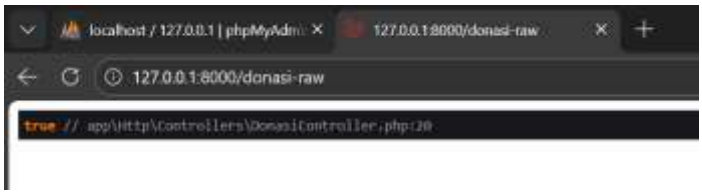
Penjelasan:

Kode tersebut menunjukkan cara memasukkan data ke dalam database menggunakan Raw SQL Query di Laravel melalui fasad DB::insert. Alih-alih

menggunakan Query Builder atau Eloquent, Anda menulis perintah SQL standar secara manual untuk mengisi kolom seperti nama, email, dan jumlah donasi pada tabel donasis. Fungsi ini juga memanfaatkan fungsi bawaan database CURDATE() untuk mengisi tanggal secara otomatis, dan hasil eksekusinya akan ditampilkan melalui dump($result) berupa nilai boolean untuk memastikan apakah data tersebut berhasil ditambahkan.

- Buat route

Output:



2. Buat fungsi untuk insert data menggunakan Query Builder

- Isi DonasiController.php dengna ini

```
public function insertQB()
{
    $result = DB::table('donasis')->insert([
        'nama_donatur' => 'Aulia Putri',
        'email' => 'aulia@gmail.com',
        'jumlah_donasi' => 100000,
        'metode_pembayaran' => 'Transfer',
        'pesan' => 'Untuk kemanusiaan',
        'tanggal_donasi' => now(),
    ]);

    dump($result);
```

```
        }
    }
```

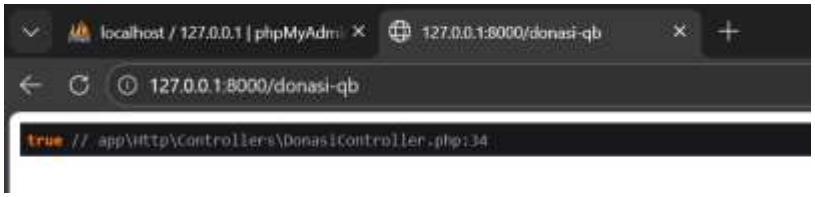
Penjelasan:

Kode tersebut merupakan fungsi insertQB() yang menggunakan metode Query Builder dari Laravel untuk memasukkan data donasi baru ke dalam tabel donasis. Berbeda dengan penggunaan Raw SQL, metode DB::table()->insert() ini menggunakan *associative array* untuk menentukan nilai pada setiap kolom secara lebih aman dan bersih, termasuk penggunaan fungsi now() untuk mengisi kolom tanggal_donasi dengan waktu saat ini. Hasil dari operasi ini kemudian ditampilkan menggunakan fungsi dump($result), yang akan mengembalikan nilai true jika data berhasil disimpan ke dalam database.

- Buat route

```
Route::get('/donasi-qb', [DonasiController::class, 'insertQB']);
```

Output:



3. Buat fungsi untuk insert data menggunakan Eloquent ORM

- Buat model Donasi

Isi dengan kode ini

```
<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Model;

class Donasi extends Model
{
    protected $table = 'donasis';

    protected $fillable = [
        'nama_donatur',
```

```
            'email',
            'jumlah_donasi',
            'metode_pembayaran',
            'pesan',
            'tanggal_donasi',
        ];
    }
```

Penjelasan:

Kode tersebut merupakan file Model Eloquent bernama Donasi yang berfungsi sebagai representasi tabel donasis dalam aplikasi Laravel. Dengan mendefinisikan properti $table, Anda secara eksplisit menghubungkan model ini ke tabel yang sesuai di database, sementara properti $fillable bertugas sebagai pengaman (*Mass Assignment*) yang menentukan kolom mana saja yang diizinkan untuk diisi secara massal melalui metode seperti create() atau update(). Penggunaan model ini memudahkan Anda untuk melakukan operasi database dengan sintaks yang lebih humanis dan berorientasi objek dibandingkan menggunakan query SQL manual.

- Isi di DonasiController.php

```
    public function insertEloquent()
    {
        $donasi = Donasi::create([
            'nama_donatur' => 'Rizky Ananda',
            'email' => 'rizky@gmail.com',
            'jumlah_donasi' => 75000,
            'metode_pembayaran' => 'Tunai',
            'pesan' => 'Ikhlas',
            'tanggal_donasi' => now(),
        ]);

        dump($donasi);
```

Penjelasan:

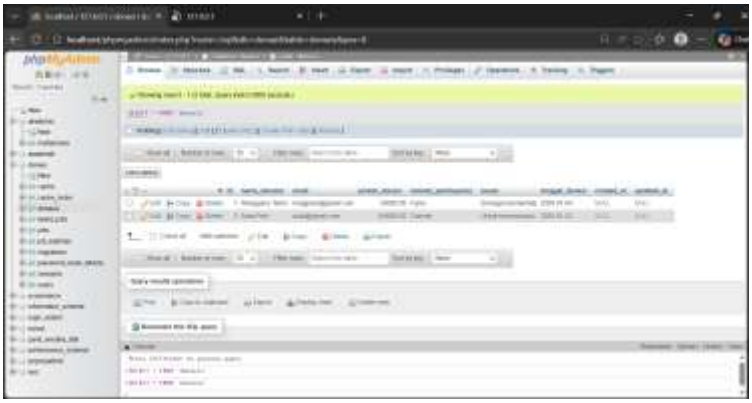
Kode tersebut merupakan fungsi insertEloquent() yang menggunakan Model Eloquent untuk menambahkan data ke tabel donasis secara lebih praktis dibandingkan Query Builder. Dengan memanggil metode Donasi::create(), Laravel

secara otomatis memetakan data dalam array ke kolom-kolom database yang telah diizinkan melalui properti $fillable pada Model. Penggunaan perintah now() memastikan kolom tanggal terisi dengan waktu saat ini, dan fungsi dump($donasi) akan menampilkan objek model lengkap hasil dari proses penyimpanan tersebut, termasuk atribut otomatis seperti ID dan waktu pembuatan.

- Buat route

```
Route::get('/donasi-eloquent', [DonasiController::class,
'insertEloquent']);
```

Output di database phpMyAdmin



Penjelasan:

Berdasarkan hasil pengujian, proses insert data donasi tunai menggunakan Raw SQL Query pada Laravel berhasil dijalankan dengan baik. Hal ini dibuktikan dengan munculnya data baru pada tabel donasis di database melalui phpMyAdmin. Data yang tersimpan meliputi nama donatur, email, jumlah donasi, metode pembayaran, pesan, serta tanggal donasi. Keberhasilan ini menunjukkan bahwa perintah SQL INSERT INTO yang dijalankan melalui DB Facade Laravel mampu menyimpan data donasi secara langsung ke database sesuai dengan struktur tabel yang telah dibuat menggunakan migration.

BAB III

PENUTUP

A. Kesimpulan

Berdasarkan hasil praktikum yang telah dilakukan, dapat disimpulkan bahwa metode Raw SQL Queries pada Laravel dapat digunakan secara efektif untuk melakukan proses insert data ke dalam database. Data donasi tunai berhasil disimpan pada tabel donasis dan dapat ditampilkan melalui phpMyAdmin. Hal ini menunjukkan bahwa integrasi antara Laravel, DB Facade, dan MySQL berjalan dengan baik, serta memberikan pemahaman kepada mahasiswa mengenai cara pengelolaan data menggunakan query SQL mentah dalam aplikasi berbasis framework.