

PEMROGRAMAN WEB LANJUT
JOB SHEET 3 - MIGRATION, SEEDER, DB FAÇADE,
QUERY BUILDER, dan ELOQUENT ORM



Disusun oleh:

Stefanus Ageng Budi Utomo

(2241720126)

JURUSAN TEKNOLOGI INFORMASI
POLITEKNIK NEGERI MALANG
TAHUN AJARAN 2023/2024

A. Pengaturan Database

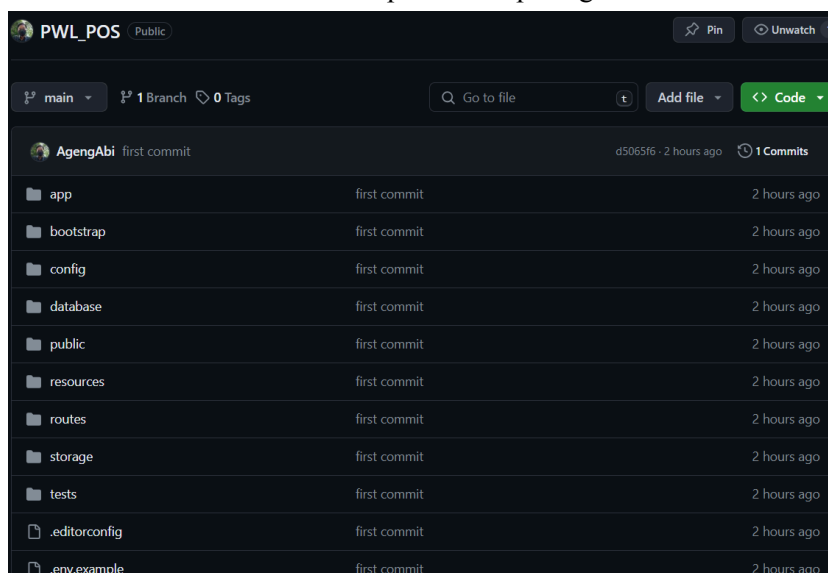
Database atau basis data menjadi komponen penting dalam membangun sistem. Hal ini dikarenakan database menjadi tempat untuk menyimpan data-data transaksi yang ada pada sistem. Koneksi ke database perlu kita atur agar sesuai dengan database yang kita gunakan.

Praktikum 1- Pengaturan database

1. Buka aplikasi phpMyAdmin, dan buat database baru dengan nama PWL_POS
2. Buka aplikasi VSCode dan buka folder project PWL_POS yang sudah kita buat.
3. Copy file .env.example menjadi .env.
4. Buka file .env, dan pastikan konfigurasi APP_KEY bernilai. Jika belum bernilai silahkan kalian generate menggunakan php artisan.
5. Edit file .env dan sesuaikan dengan database yang telah dibuat

```
.env
1 APP_NAME=Laravel
2 APP_ENV=local
3 APP_KEY=base64:nD/aXG4qPCe6ZHK0Vs90b/zEqvdoL7Dmq1IOck1jeus=
4 APP_DEBUG=true
5 APP_URL=http://localhost
6
7 LOG_CHANNEL=stack
8 LOG_DEPRECATIONS_CHANNEL=null
9 LOG_LEVEL=debug
10
11 DB_CONNECTION=mysql
12 DB_HOST=127.0.0.1
13 DB_PORT=3306
14 DB_DATABASE=PWL_POS
15 DB_USERNAME=root
16 DB_PASSWORD=
```

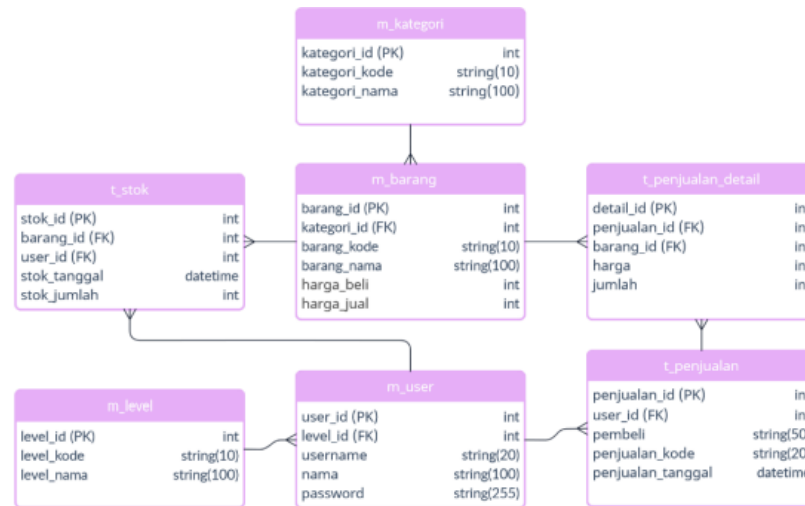
6. Laporkan hasil Praktikum-1 ini dan commit perubahan pada git.



B. Migration

Migration pada Laravel merupakan sebuah fitur yang dapat membantu kita mengelola database secara efisien dengan menggunakan kode program. Migration membantu kita dalam membuat (create), mengubah (edit), dan menghapus (delete) struktur tabel dan kolom pada database yang sudah kita buat dengan cepat dan mudah. Dengan Migration, kita juga dapat melakukan perubahan pada struktur database tanpa harus menghapus data yang ada.

Salah satu keunggulan menggunakan migration adalah mempermudah proses instalasi aplikasi kita, Ketika aplikasi yang kita buat akan diimplementasikan di server/komputer lain.



Dalam membuat file migration di Laravel, yang perlu kita perhatikan adalah struktur table yang ingin kita buat.

TIPS MIGRATION

Buatlah file migration untuk table yang tidak memiliki relasi (table yang tidak ada foreign key) dulu, dan dilanjutkan dengan membuat file migrasi yang memiliki relasi yang sedikit, dan dilanjutkan ke file migrasi dengan table yang memiliki relasi yang banyak.

Dari tips di atas, kita dapat melakukan cek untuk desain database yang sudah ada dengan mengetahui jumlah foreign key yang ada. Dan kita bisa menentukan table mana yang akan kita buat migrasinya terlebih dahulu.

No Urut	Nama Tabel	Jumlah FK
1	m_level	0
2	m_kategori	0
3	m_user	1
4	m_barang	1
5	t_penjualan	1

6	t_stok	2
7	t_penjualan_detail	2

INFO

Secara default Laravel sudah ada table users untuk menyimpan data pengguna, tapi pada praktikum ini, kita gunakan table sesuai dari file Studi Kasus PWL.pdf yaitu m_user.

Pembuatan file migrasi bisa menggunakan 2 cara, yaitu

- Menggunakan artisan untuk membuat file migration

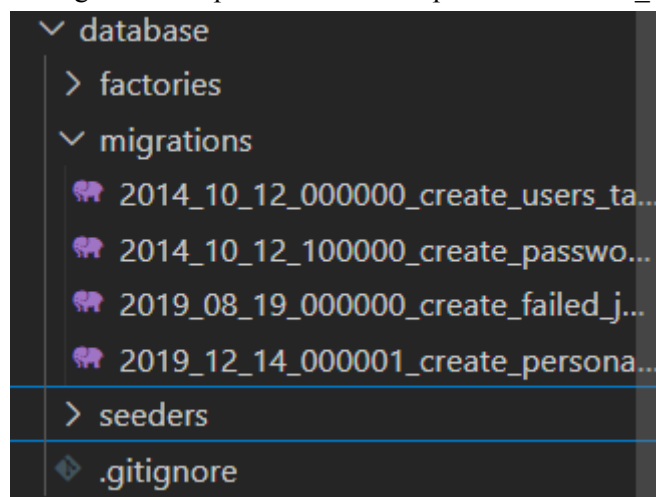
```
php artisan make:migration <nama-file-tabel> --create=<nama-tabel>
```

- Menggunakan artisan untuk membuat file model + file migration

```
php artisan make:model <nama-model> -m
```

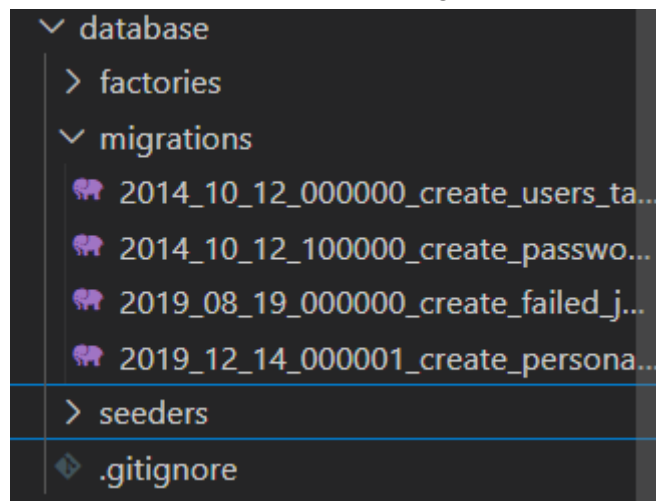
Perintah -m di atas adalah shorthand untuk opsi membuat file migrasi berdasarkan model yang dibuat.

Pada Laravel, file-file migration ataupun seeder berada pada folder PWL_POS/database



Pratikum 2.1 - Pembuatan file migrasi tanpa relasi

1. Buka terminal VSCode kalian, untuk melihat default migration dari laravel



2. Kita abaikan dulu yang di kotak merah (jangan di hapus)
3. Kita buat file migrasi untuk table m_level dengan perintah

```
php artisan make:migration create_m_level_table --create=m_level
```

```
2024_03_06_041808_create_m_level_table.php U ●
database > migrations > 2024_03_06_041808_create_m_level_table.php > class > up
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::create('m_level', function (Blueprint $table) {
15             $table->id();
16             $table->timestamps();
17         });
18     }
19
20     /**
21      * Reverse the migrations.
22      */
23     public function down(): void
24     {
25         Schema::dropIfExists('m_level');
26     }
27 };
```

4. Kita perhatikan bagian yang di kotak merah, bagian tersebut yang akan kita modifikasi sesuai desain database yang sudah ada

```

2024_03_06_041808_create_m_level_table.php U X
database > migrations > 2024_03_06_041808_create_m_level_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::create('m_level', function (Blueprint $table) {
15             $table->id('level_id');
16             $table->string('level_kode', 10)->unique();
17             $table->string('level_nama', 100);
18             $table->timestamps();
19         });
20     }
21
22     /**
23      * Reverse the migrations.
24      */
25     public function down(): void
26     {
27         Schema::dropIfExists('m_level');
28     }
29 };

```

5. Simpan kode pada tahapan 4 tersebut, kemudian jalankan perintah ini pada terminal VSCode untuk melakukan migrasi

php artisan migrate

```

PS D:\VAD\Kuliah\semester 4\peminatan Web Lanjut\code\VM_Pos php artisan migrate
[INFO] Preparing database.
Creating migration table ..... 16ms DONE
[INFO] Running migrations.
2014_10_12_000000 create_users_table ..... 44ms DONE
2014_10_12_100000 create_password_reset_tokens_table ..... 11ms DONE
2019_08_19_000000 create_failed_jobs_table ..... 41ms DONE
2019_12_14_000001 create_personal_access_tokens_table ..... 47ms DONE
2024_03_06_041808 create_m_level_table ..... 30ms DONE

```

6. Kemudian kita cek di phpMyAdmin apakah table sudah ter-generate atau belum

Table	Action	Rows	Type	Collation	Size	Overhead
<input type="checkbox"/> failed_jobs	★ Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_unicode_ci	16.0 KiB	-
<input type="checkbox"/> migrations	★ Browse Structure Search Insert Empty Drop	5	InnoDB	utf8mb4_unicode_ci	16.0 KiB	-
<input type="checkbox"/> m_level	★ Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_unicode_ci	16.0 KiB	-
<input type="checkbox"/> password_reset_tokens	★ Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_unicode_ci	16.0 KiB	-
<input type="checkbox"/> personal_access_tokens	★ Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_unicode_ci	16.0 KiB	-
<input type="checkbox"/> users	★ Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_unicode_ci	16.0 KiB	-
6 tables	Sum	5	InnoDB	utf8mb4_0900_ai_ci	96.0 KiB	0 B

7. Ok, table sudah dibuat di database
8. Buat table database dengan migration untuk table m_kategori yang sama-sama tidak memiliki foreign key
9. Laporkan hasil Praktikum-2.1 ini dan commit perubahan pada git.

```
2024_03_06_041808_create_m_level_table.php U 2024_03_06_060932_create_m_kategori_table.php U X
database > migrations > 2024_03_06_060932_create_m_kategori_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     /**
10      * Run the migrations.
11      */
12     public function up(): void
13     {
14         Schema::create('m_kategori', function (Blueprint $table) {
15             $table->id('kategori_id');
16             $table->string('kategori_kode', 10)->unique();
17             $table->string('kategori_nama', 100);
18             $table->timestamps();
19         });
20     }
21 }
22 /**
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS powershell + - - - - -

2024_03_06_041808_create_m_level_table 38ms DONE

PS D:\Abi\kuliah\semester 4\Pemrograman Web Lanjut\code\PWL_POS> php artisan make:migration create_m_kategori_table --create=m_kategori


INFO Migration [D:\Abi\kuliah\semester 4\Pemrograman Web Lanjut\code\PWL_POS\database\Migrations\2024_03_06_060932_create_m_kategori_table.php] created successfully.

PS D:\Abi\kuliah\semester 4\Pemrograman Web Lanjut\code\PWL_POS> php artisan migrate

INFO Running migrations.

2024_03_06_060932_create_m_kategori_table 69ms DONE

PS D:\Abi\kuliah\semester 4\Pemrograman Web Lanjut\code\PWL_POS>

 **PWL_POS** Public

Pin Unwatch 1


main

1 Branch 0 Tags

Go to file

Add file

<> Code

 **AgengAbi** feat: add migration table level and category fa4b1d2 · 6 minutes ago 2 Commits

app	first commit	4 hours ago
bootstrap	first commit	4 hours ago
config	first commit	4 hours ago
database	feat: add migration table level and category	6 minutes ago

Pratikum 2.2 - Pembuatan file migrasi dengan relasi

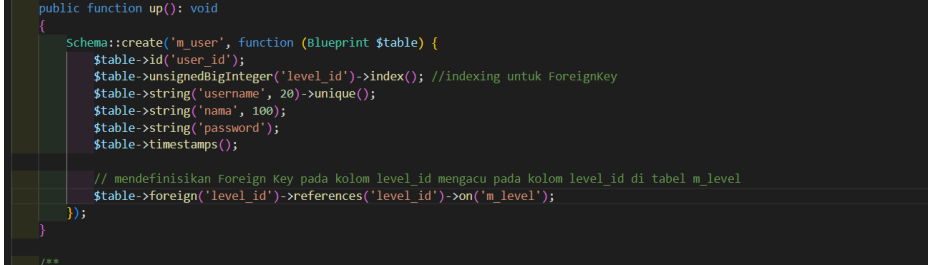
1. Buka terminal VSCode kalian, dan buat file migrasi untuk table m_user
2. Buka file migrasi untuk table m_user, dan modifikasi seperti berikut.

```
return new class extends Migration
{
    /**
     * Run the migrations.
     */
    public function up(): void
    {
        Schema::create('m_user', function (Blueprint $table) {
            $table->id('user_id');
            $table->unsignedBigInteger('level_id')->index(); // indexing untuk ForeignKey
            $table->string('username', 20)->unique(); // unique untuk memastikan tidak ada username yang sama
            $table->string('nama', 100);
            $table->string('password');
            $table->timestamps();

            // Mendefinisikan Foreign Key pada kolom level_id mengacu pada kolom level_id di tabel m_level
            $table->foreign('level_id')->references('level_id')->on('m_level');
        });

        /**
         * Reverse the migrations.
         */
        public function down(): void
        {
            Schema::dropIfExists('m_user');
        }
    }
};
```

3. Simpan kode program Langkah 2, dan jalankan perintah php artisan migrate. Amati apa yang terjadi pada database.



```
public function up(): void
{
    Schema::create('m_user', function (Blueprint $table) {
        $table->id('user_id');
        $table->unsignedBigInteger('level_id')->index(); //indexing untuk ForeignKey
        $table->string('username', 20)->unique();
        $table->string('nama', 100);
        $table->string('password');
        $table->timestamps();

        // mendefinisikan Foreign Key pada kolom level_id mengacu pada kolom level_id di tabel m_level
        $table->foreign('level_id')->references('level_id')->on('m_level');
    });
}
```

```
2024_03_06_060932_create_m_kategori_table ..... 69ms DONE

Abi\kuliahs\semester 4\Pemrograman Web Lanjut\code\PMU_POS> php artisan make:migration create_m_user_table --create=m_user

0 Migration [D:\Abi\kuliahs\semester 4\Pemrograman Web Lanjut\code\PMU_POS\database\migrations\2024_03_06_070052_create_m_user_table.php] creat

Abi\kuliahs\semester 4\Pemrograman Web Lanjut\code\PMU_POS> php artisan migrate

0 Running migrations.

2024_03_06_070052_create_m_user_table ..... 141ms DONE
```

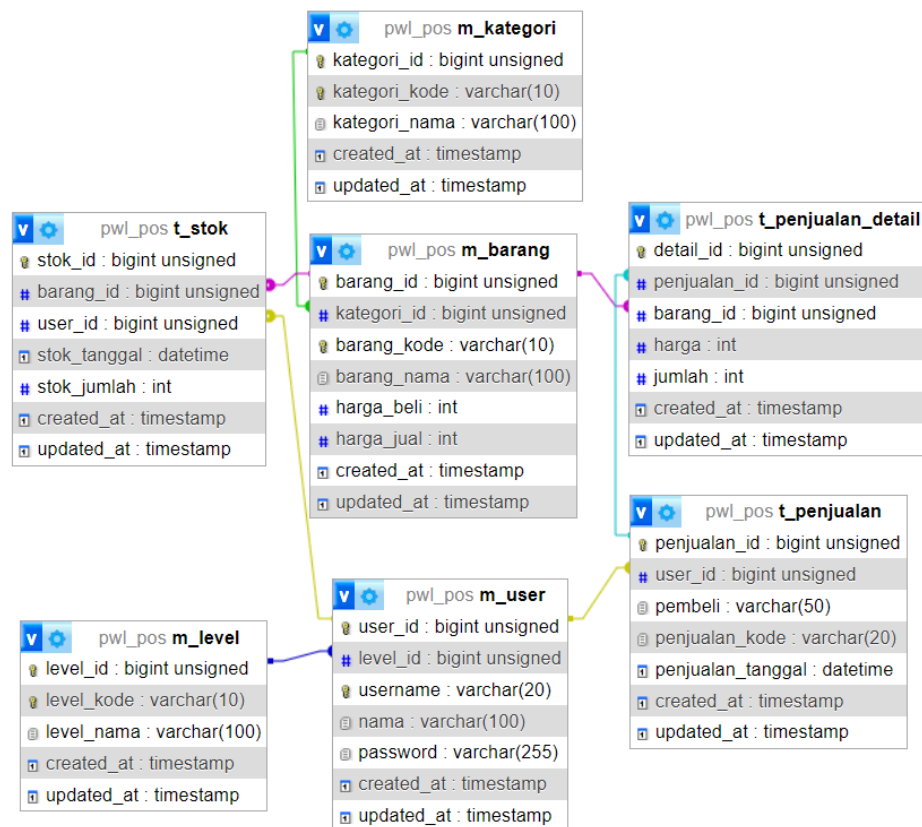
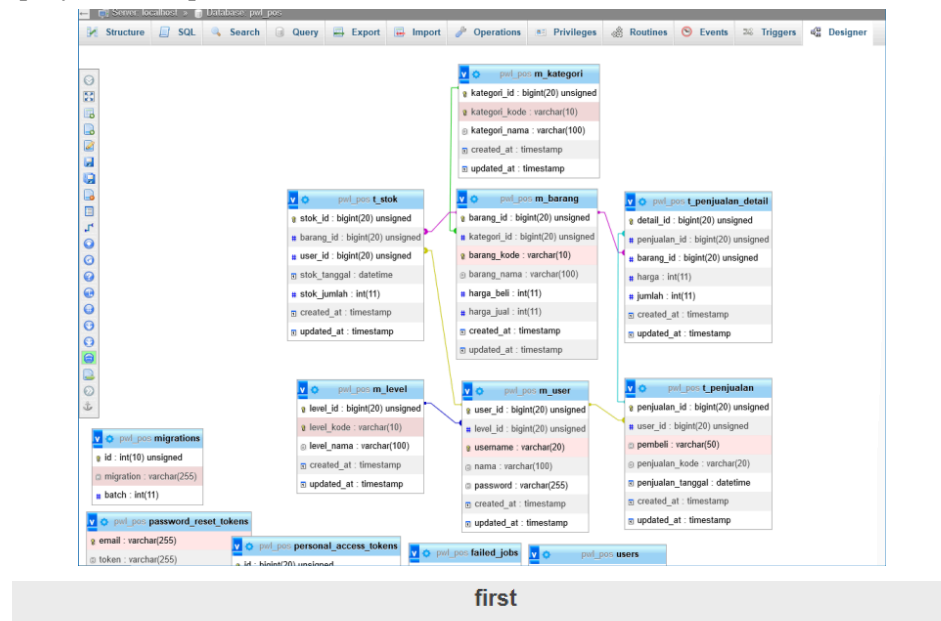
Table	Action	Rows	Type	Collation	Size	Overhead
failed_jobs	Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_unicode_ci	16.0 KiB	-
migrations	Browse Structure Search Insert Empty Drop	7	InnoDB	utf8mb4_unicode_ci	16.0 KiB	-
m_kategori	Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_unicode_ci	16.0 KiB	-
m_level	Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_unicode_ci	16.0 KiB	-
m_user	Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_unicode_ci	32.0 KiB	-
password_reset_tokens	Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_unicode_ci	16.0 KiB	-
personal_access_tokens	Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_unicode_ci	16.0 KiB	-
users	Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_unicode_ci	16.0 KiB	-
8 tables	Sum	7	InnoDB	utf8mb4_0900_ai_ci	144.0 KiB	0 B

4. Buat table database dengan migration untuk table-tabel yang memiliki foreign key.

m_barang
t_penjualan
t_stok
t_penjualan_detail

```
2024_03_06_070052_create_m_user_table ..... 147ms DONE
2024_03_06_073239_create_m_barang_table ..... 74ms DONE
2024_03_06_073305_create_t_penjualan_table ..... 78ms DONE
2024_03_06_073322_create_t_stok_table ..... 184ms DONE
2024_03_06_073345_create_t_penjualan_detail_table ..... 329ms DONE
```


5. Jika semua file migrasi sudah di buat dan dijalankan maka bisa kita lihat tampilan designer pada phpMyAdmin seperti berikut



6. Laporkan hasil Praktikum-2.2 ini dan commit perubahan pada git.



Praktikum 3 - Membuat file seeder

1. Kita akan membuat file seeder untuk table `m_level` dengan mengetikkan perintah

```
php artisan make:seeder LevelSeeder
```

2. Selanjutnya, untuk memasukkan data awal, kita modifikasi file tersebut di dalam function `run()`

```
LevelSeeder.php X
database > seeders > LevelSeeder.php > ...
1  <?php
2
3  namespace Database\Seeders;
4
5  use Illuminate\Database\Console\Seeds\WithoutModelEvents;
6  use Illuminate\Database\Seeder;
7  use Illuminate\Support\Facades\DB;
8
9  class LevelSeeder extends Seeder
10 {
11     /**
12      * Run the database seeds.
13      */
14     public function run(): void
15     {
16         $data = [
17             ['level_id' => 1, 'level_kode' => 'ADM', 'level_nama' => 'Administrator'],
18             ['level_id' => 2, 'level_kode' => 'MNG', 'level_nama' => 'Manager'],
19             ['level_id' => 3, 'level_kode' => 'STF', 'level_nama' => 'Staff/Kasir'],
20         ];
21         DB::table('m_level')->insert($data);
22     }
23 }
```

3. Selanjutnya, kita jalankan file seeder untuk table `m_level` pada terminal

```
php artisan db:seed --class=LevelSeeder
```

```
PS D:\Abi\kuliah\semester 4\Pemrograman Web Lanjut\code\PWL_POS> php artisan db:seed --class=LevelSeeder
INFO Seeding database.
```

4. Ketika seeder berhasil dijalankan maka akan tampil data pada table `m_level`

				level_id	level_kode	level_nama	created_at	updated_at
<input type="checkbox"/>	Edit	Copy	Delete	1	ADM	Administrator	NULL	NULL
<input type="checkbox"/>	Edit	Copy	Delete	2	MNG	Manager	NULL	NULL
<input type="checkbox"/>	Edit	Copy	Delete	3	STF	Staff/Kasir	NULL	NULL

5. Sekarang kita buat file seeder untuk table `m_user` yang me-refer ke table `m_level`

```
php artisan make:seeder UserSeeder
```

6. Modifikasi file class UserSeeder seperti berikut

```
9 class UserSeeder extends Seeder
10 {
11     public function run(): void
12     {
13         $data = [
14             [
15                 'user_id' => 1,
16                 'level_id' => 1,
17                 'username' => 'admin',
18                 'nama' => 'Administrator',
19                 'password' => Hash::make('12345'), // class untuk mengenkripsi/hash password
20             ],
21             [
22                 'user_id' => 2,
23                 'level_id' => 2,
24                 'username' => 'manager',
25                 'nama' => 'Manager',
26                 'password' => Hash::make('12345'),
27             ],
28             [
29                 'user_id' => 3,
30                 'level_id' => 3,
31                 'username' => 'staff',
32                 'nama' => 'Staff/Kasir',
33                 'password' => Hash::make('12345'),
34             ],
35         ];
36         DB::table('m_user')->insert($data);
37     }
38 }
```

7. Jalankan perintah untuk mengeksekusi class UserSeeder

```
php artisan db:seed --class=UserSeeder
```

```
PS D:\Abi\kuliah\semester 4\Pemrograman Web Lanjut\code\PWL_POS> php artisan db:seed --class=UserSeeder
INFO Seeding database.
```

8. Perhatikan hasil seeder pada table m_user

	user_id	level_id	username	nama	password
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	1	1	admin	Administrator	\$2y\$12\$Sw.Kekmg38lmpHl0KJ/2qeN
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	2	2	manager	Manager	\$2y\$12\$Kj0bo5uFsmKhbrKlr5pgvOV
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	3	3	staff	Staff/Kasir	\$2y\$12\$puVuEnwFowqL24KB6K7Vh

9. Ok, data seeder berhasil di masukkan ke database.
10. Sekarang coba kalian masukkan data seeder untuk table yang lain, dengan ketentuan seperti berikut

No	Nama Tabel	Jumlah Data	Keterangan
1	m_kategori	5	5 kategori barang
2	m_barang	10	10 barang yang berbeda
3	t_stok	10	Stok untuk 10 barang
4	t_penjualan	10	10 transaksi penjualan
5	t_penjualan_detail	30	3 barang untuk setiap transaksi penjualan

```
PS D:\Abi\kuliah\semester 4\Pemrograman Web Lanjut\code\PWL_POS> php artisan db:seed --class=KategoriSeeder
INFO Seeding database.
```

```
PS D:\Abi\kuliah\semester 4\Pemrograman Web Lanjut\code\PWL_POS> php artisan db:seed --class=BarangSeeder
INFO Seeding database.
```

```
PS D:\Abi\kuliah\semester 4\Pemrograman Web Lanjut\code\PWL_POS> php artisan db:seed StokSeeder  
INFO Seeding database.
```

```
PS D:\Abi\kuliah\semester 4\Pemrograman Web Lanjut\code\PWL_POS> php artisan db:seed --class=PenjualanSeeder  
INFO Seeding database.
```

```
PS D:\Abi\kuliah\semester 4\Pemrograman Web Lanjut\code\PWL_POS> php artisan db:seed --class=PenjualanDetailSeeder  
INFO Seeding database.
```

11. Jika sudah, laporkan hasil Praktikum-3 ini dan commit perubahan pada git

The screenshot shows the GitHub interface for a repository named 'PWL_POS'. The repository is public and has 1 branch (main) and 0 tags. The commit history is displayed, showing a commit by 'AgengAbi' with the message 'feat: add seeder'. The commit hash is 'f51e4d5' and it was made 'now'. The commit history table lists the files changed in each commit:

File	Commit Message	Time
app	first commit	2 days ago
bootstrap	first commit	2 days ago
config	first commit	2 days ago
database	feat: add seeder	now

DB FECADE

Pratikum 4 - Implementasi DB Fecade

1. Kita buat controller dahulu untuk mengelola data pada table m_level

```
php artisan make:controller LevelController
```

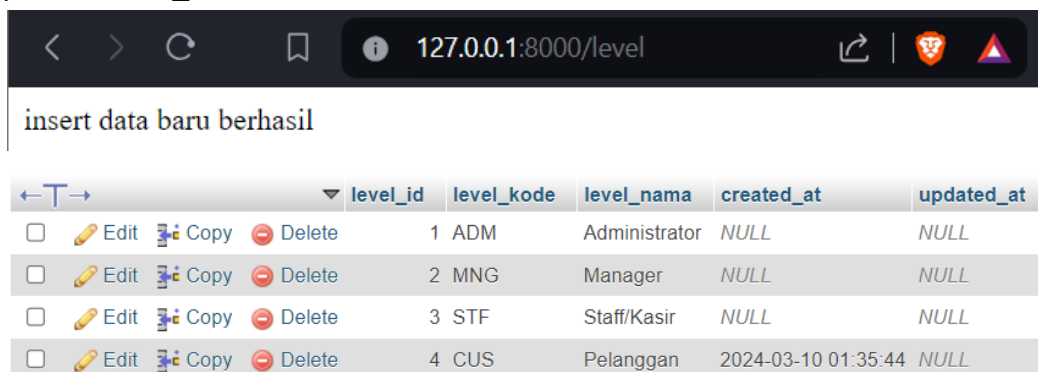
2. Kita modifikasi dulu untuk routing-nya, ada di PWL_POS/routes/web.php

```
Route::get('/', function () {  
    return view('welcome');  
});  
  
Route::get('/level', [LevelController::class, 'index']);
```

3. Selanjutnya, kita modifikasi file LevelController untuk menambahkan 1 data ke table m_level

```
<?php  
  
namespace App\Http\Controllers;  
  
use Illuminate\Http\Request;  
use Illuminate\Support\Facades\DB;  
  
class LevelController extends Controller  
{  
    public function index()  
    {  
        DB::insert('insert into m_level(level_code, level_nama, create_at) values  
            (?, ?, ?)', ['CUS', 'Pelanggan', now()]);  
  
        return 'insert data baru berhasil';  
    }  
}
```

4. Kita coba jalankan di browser dengan url localhost/PWL_POS/public/level dan amati apa yang terjadi pada table m_level di database, screenshot perubahan yang ada pada table m_level



	level_id	level_kode	level_nama	created_at	updated_at
<input type="checkbox"/> Edit Copy Delete	1	ADM	Administrator	NULL	NULL
<input type="checkbox"/> Edit Copy Delete	2	MNG	Manager	NULL	NULL
<input type="checkbox"/> Edit Copy Delete	3	STF	Staff/Kasir	NULL	NULL
<input type="checkbox"/> Edit Copy Delete	4	CUS	Pelanggan	2024-03-10 01:35:44	NULL

5. Selanjutnya, kita modifikasi lagi file LevelController untuk meng-update data di table m_level seperti berikut

```

class LevelController extends Controller
{
    public function index()
    {
        // DB::insert('insert into m_level(level_kode, level_nama, created_at)
        // values(?,?,?)', ['CUS', 'Pelanggan', now()]);

        // return 'insert data baru berhasil';

        $row = DB::update('update m_level set level_nama = ? where level_kode = ?
        ', ['Customer', 'CUS']);
        return 'Update data berhasil. Jumlah data yang diupdate: ' . $row . '
        baris';
    }
}

```

6. Kita coba jalankan di browser dengan url localhost/PWL_POS/public/level lagi dan amati apa yang terjadi pada table m_level di database, screenshot perubahan yang ada pada table m_level

127.0.0.1:8000/level

Update data berhasil. Jumlah data yang diupdate: 1 baris

	level_id	level_kode	level_nama	created_at	updated_at
<input type="checkbox"/> Edit Copy Delete	1	ADM	Administrator	NULL	NULL
<input type="checkbox"/> Edit Copy Delete	2	MNG	Manager	NULL	NULL
<input type="checkbox"/> Edit Copy Delete	3	STF	Staff/Kasir	NULL	NULL
<input type="checkbox"/> Edit Copy Delete	4	CUS	Customer	2024-03-10 01:35:44	NULL

7. Kita coba modifikasi lagi file LevelController untuk melakukan proses hapus data

```

class LevelController extends Controller
{
    public function index()
    {
        // DB::insert('insert into m_level(level_kode, level_nama, created_at)
        // values(?,?,?)', ['CUS', 'Pelanggan', now()]);

        // return 'insert data baru berhasil';

        // $row = DB::update('update m_level set level_nama = ? where level_kode
        // = ?', ['Customer', 'CUS']);
        // return 'Update data berhasil. Jumlah data yang diupdate: ' . $row . '
        // baris';

        $row = DB::delete('delete from m_level where level_kode=?', ['CUS']);
        return 'Delete data berhasil. Jumlah data yang dihapus ' . $row . '
        baris';
    }
}

```

8. Method terakhir yang kita coba adalah untuk menampilkan data yang ada di table m_level. Kita modifikasi file LevelController seperti berikut

```

class LevelController extends Controller
{
    public function index()
    {
        // DB::insert('insert into m_level(level_kode, level_nama, created_at)
        // values(?,?,?)', ['CUS', 'Pelanggan', now()]);
        // return 'insert data baru berhasil';

        // $row = DB::update('update m_level set level_nama = ? where level_kode
        // = ?', ['Customer', 'CUS']);
        // return 'Update data berhasil. Jumlah data yang diupdate: ' . $row . '
        // baris';

        // $row = DB::delete('delete from m_level where level_kode=?', ['CUS']);
        // return 'Delete data berhasil. Jumlah data yang dihapus ' . $row . '
        // baris';

        $data = DB::select('select * from m_level');
        return view('level', ['data' => $data]);
    }
}

```

9. Coba kita perhatikan kode yang diberi tanda kotak merah, berhubung kode tersebut memanggil view('level'), maka kita buat file view pada VSCode di PWL_POS/resources/view/level.blade.php

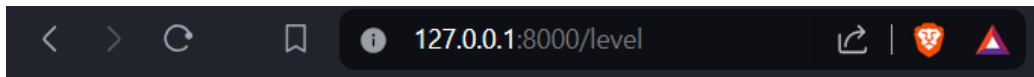
```

<html lang="en">
<head>
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title>Data Level Pengguna</title>
</head>

<body>
    <h1>Data Level Pengguna</h1>
    <table border="1" cellpadding="2" cellspacing="0">
        <tr>
            <th>ID</th>
            <th>Kode Level</th>
            <th>Nama Level</th>
        </tr>
        @foreach ($data as $d)
            <tr>
                <td>{{ $d->level_id }}</td>
                <td>{{ $d->level_kode }}</td>
                <td>{{ $d->level_nama }}</td>
            </tr>
        @endforeach
    </table>
</body>

```

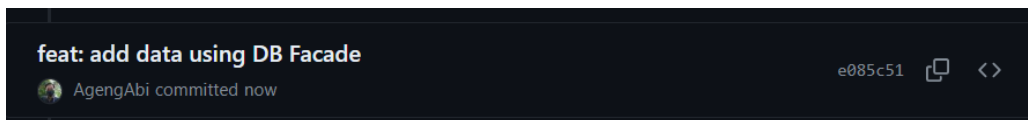
10. Silahkan dicoba pada browser dan amati apa yang terjadi



Data Level Pengguna

ID	Kode Level	Nama Level
1	ADM	Administrator
2	MNG	Manager
3	STF	Staff/Kasir

11. Laporkan hasil Praktikum-4 ini dan commit perubahan pada git.



QUERY BUILDER

Praktikum 5 - Implementasi Query Builder

1. Kita buat controller dahulu untuk mengelola data pada table m_kategori

```
php artisan make:controller KategoriController
```

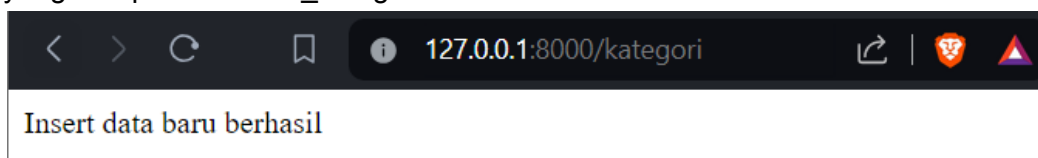
2. Kita modifikasi dulu untuk routing-nya, ada di PWL_POS/routes/web.php

```
Route::get('/', function () {  
    return view('welcome');  
});  
  
Route::get('/level', [LevelController::class, 'index']);  
Route::get('/kategori', [KategoriController::class,  
    'index']);
```

3. Selanjutnya, kita modifikasi file KategoriController untuk menambahkan 1 data ke table m_kategori

```
class KategoriController extends Controller  
{  
    public function index()  
    {  
        $data = [  
            'kategori_kode' => 'SNK',  
            'kategori_nama' => 'Snack/Makanan Ringan',  
            'created_at' => now(),  
        ];  
        DB::table('m_kategori')->insert($data);  
        return 'Insert data baru berhasil';  
    }  
}
```

4. Kita coba jalankan di browser dengan url localhost/PWL_POS/public/kategori dan amati apa yang terjadi pada table m_kategori di database, screenshot perubahan yang ada pada table m_kategori



<div><div><div><div></div><div></div></div><div><div></div><div></div></div></div><div></div></div>				kategori_id	kategori_kode	kategori_nama	created_at	updated_at
<div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div>Edit</div><div>Copy</div><div>Delete</div></div>	1	KBE1	laptop	NULL	NULL			
<div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div>Edit</div><div>Copy</div><div>Delete</div></div>	2	KBE2	smartphone	NULL	NULL			
<div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div>Edit</div><div>Copy</div><div>Delete</div></div>	3	KBE3	camera	NULL	NULL			
<div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div>Edit</div><div>Copy</div><div>Delete</div></div>	4	KBE4	headphones	NULL	NULL			
<div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div>Edit</div><div>Copy</div><div>Delete</div></div>	5	KBE5	keyboard	NULL	NULL			
<div><div><div></div><div></div></div><div><div></div><div></div></div></div> <div><div>Edit</div><div>Copy</div><div>Delete</div></div>	6	SNK	Snack/Makanan Ringan	2024-03-10 02:02:39	NULL			

5. Selanjutnya, kita modifikasi lagi file KategoriController untuk meng-update data di table m_kategori seperti berikut

```
class KategoriController extends Controller
{
    public function index()
    {
        // $data = [
        //     'kategori_kode' => 'SNK',
        //     'kategori_nama' => 'Snack/Makanan Ringan',
        //     'created_at' => now(),
        // ];
        // DB::table('m_kategori')->insert($data);
        // return 'Insert data baru berhasil';

        $row = DB::table('m_kategori')->where('kategori_kode', 'SNK')->update(
            ['kategori_nama' => 'Camilan']);
        return 'Update data berhasil. Jumlah data yang diupdate: ' . $row . '
        baris';
    }
}
```

6. Kita coba jalankan di browser dengan url localhost/PWL_POS/public/kategori lagi dan amati apa yang terjadi pada table m_kategori di database, screenshot perubahan yang ada pada table m_kategori

127.0.0.1:8000/kategori

Update data berhasil. Jumlah data yang diupdate: 1 baris

				kategori_id	kategori_kode	kategori_nama	created_at	updated_at
<input type="checkbox"/>	Edit	Copy	Delete	1	KBE1	laptop	NULL	NULL
<input type="checkbox"/>	Edit	Copy	Delete	2	KBE2	smartphone	NULL	NULL
<input type="checkbox"/>	Edit	Copy	Delete	3	KBE3	camera	NULL	NULL
<input type="checkbox"/>	Edit	Copy	Delete	4	KBE4	headphones	NULL	NULL
<input type="checkbox"/>	Edit	Copy	Delete	5	KBE5	keyboard	NULL	NULL
<input type="checkbox"/>	Edit	Copy	Delete	6	SNK	Camilan	2024-03-10 02:02:39	NULL

7. Kita coba modifikasi lagi file KategoriController untuk melakukan proses hapus data

```

class KategoriController extends Controller
{
    public function index()
    {
        // $data = [
        //     'kategori_kode' => 'SNK',
        //     'kategori_nama' => 'Snack/Makanan Ringan',
        //     'created_at' => now(),
        // ];
        // DB::table('m_kategori')->insert($data);
        // return 'Insert data baru berhasil';

        // $row = DB::table('m_kategori')->where('kategori_kode', 'SNK')->update
        // (['kategori_nama' => 'Camilan']);
        // return 'Update data berhasil. Jumlah data yang diupdate: ' . $row . '
        // baris';

        $row = DB::table('m_kategori')->where('kategori_kode', 'SNK')->delete();
        return 'Delete data berhasil. Jumlah data yang dihapus: ' . $row . '
        baris';
    }
}

```

127.0.0.1:8000/kategori

Delete data berhasil. Jumlah data yang dihapus: 1 baris

					kategori_id	kategori_kode	kategori_nama	created_at	updated_at
<input type="checkbox"/>	Edit	Copy	Delete		1	KBE1	laptop	NULL	NULL
<input type="checkbox"/>	Edit	Copy	Delete		2	KBE2	smartphone	NULL	NULL
<input type="checkbox"/>	Edit	Copy	Delete		3	KBE3	camera	NULL	NULL
<input type="checkbox"/>	Edit	Copy	Delete		4	KBE4	headphones	NULL	NULL
<input type="checkbox"/>	Edit	Copy	Delete		5	KBE5	keyboard	NULL	NULL

8. Method terakhir yang kita coba adalah untuk menampilkan data yang ada di table m_kategori. Kita modifikasi file KategoriController seperti berikut

```

class KategoriController extends Controller
{
    public function index()
    {
        // ...
        $data = DB::table('m_kategori')->get();
        return view('kategori', ['data' => $data]);
    }
}

```

9. Coba kita perhatikan kode yang diberi tanda kotak merah, berhubung kode tersebut memanggil view('kategori'), maka kita buat file view pada VSCode di PWL_POS/resources/view/kategori.blade.php

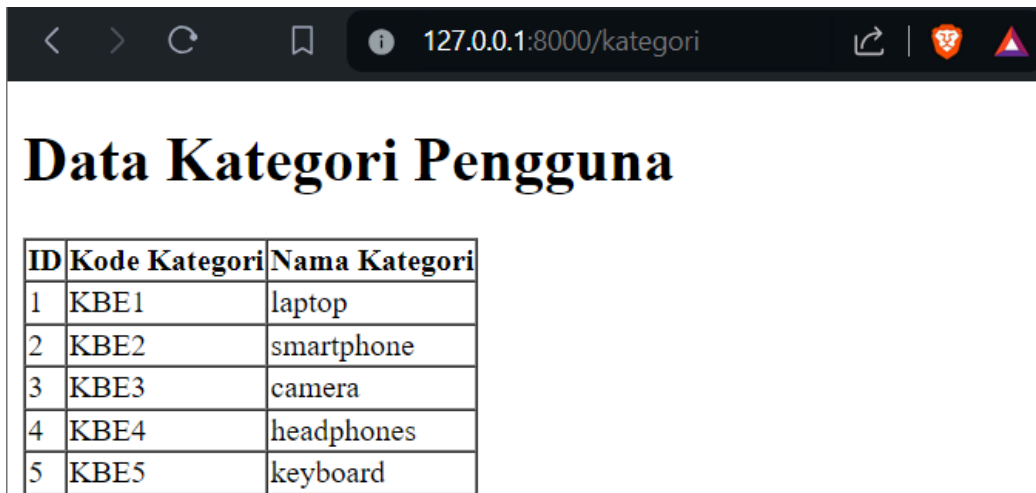
```

<html lang="en">
<head>
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <meta http-equiv="X-UA-Compatible" content="ie=edge">
  <title>Data Kategori Barang</title>
</head>

<body>
  <h1>Data Kategori Pengguna</h1>
  <table border="1" cellpadding="2" cellspacing="0">
    <tr>
      <th>ID</th>
      <th>Kode Kategori</th>
      <th>Nama Kategori</th>
    </tr>
    @foreach ($data as $d)
      <tr>
        <td>{{ $d->kategori_id }}</td>
        <td>{{ $d->kategori_kode }}</td>
        <td>{{ $d->kategori_nama }}</td>
      </tr>
    @endforeach
  </table>
</body>

```

10. Silahkan dicoba pada browser dan amati apa yang terjadi.



ID	Kode Kategori	Nama Kategori
1	KBE1	laptop
2	KBE2	smartphone
3	KBE3	camera
4	KBE4	headphones
5	KBE5	keyboard

11. Laporkan hasil Praktikum-5 ini dan commit perubahan pada git



```

feat: add data using query builder
861d0ec
AgengAbi committed now

```

ELOQUENT ORM

Praktikum 6 - Implementasi Eloquent ORM

1. Kita buat file model untuk tabel m_user dengan mengetikkan perintah

```
php artisan make:model UserModel
```

2. Setelah berhasil generate model, terdapat 2 file pada folder model yaitu file User.php bawaan dari laravel dan file UserModel.php yang telah kita buat. Kali ini kita akan menggunakan file UserModel.php
3. Kita buka file UserModel.php dan modifikasi seperti berikut

```
<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;

class UserModel extends Model
{
    use HasFactory;

    protected $table = 'm_user'; // Mendefinisiakn nama tabel yang
    digunakan oleh model ini
    protected $primaryKey = 'user_id'; // Mendefinisikan primary key dari tabel
    yang digunakan
}
```

4. Kita modifikasi route web.php untuk mencoba routing ke controller UserController

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

Route::get('/level', [LevelController::class, 'index']);
Route::get('/kategori', [KategoriController::class, 'index']);
Route::get('/user', [UserController::class, 'index']);
```

5. Sekarang, kita buat file controller UserController dan memodifikasinya seperti berikut

```

<?php

namespace App\Http\Controllers;

use App\Models\UserModel;
use Illuminate\Http\Request;

class UserController extends Controller
{
    public function index()
    {
        $user = UserModel::all(); // ambil semua data dari tabel m_user
        return view('user', ['data' => $user]);
    }
}

```

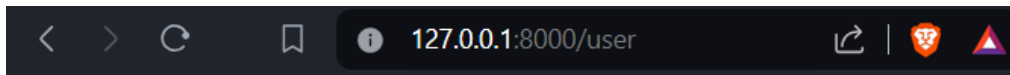
6. Kemudian kita buat view user.blade.php

```

<html lang="en">
<body>
    <table border="1" cellpadding="2" cellspacing="0">
        <tr>
            <th>Username</th>
            <th>Nama</th>
            <th>ID Level Pengguna</th>
        </tr>
        @foreach ($data as $d)
            <tr>
                <td>{{ $d->user_id }}</td>
                <td>{{ $d->username }}</td>
                <td>{{ $d->nama }}</td>
                <td>{{ $d->level_id }}</td>
            </tr>
        @endforeach
    </table>
</body>

```

7. Jalankan di browser, catat dan laporkan apa yang terjadi



Data User

ID	Username	Nama	ID Level Pengguna
1	admin	Administrator	1
2	manager	Manager	2
3	staff	Staff/Kasir	3

- Semua data dari tabel m_user diambil dan ditampilkan melalui UserModel.
8. Setelah itu, kita modifikasi lagi file UserController

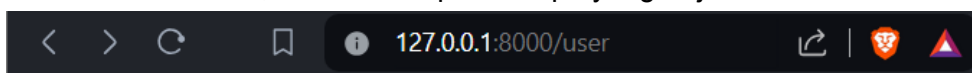
```
<?php

namespace App\Http\Controllers;

use App\Models\UserModel;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Hash;

class UserController extends Controller
{
    public function index()
    {
        $data = [
            'username' => 'customer-1',
            'nama' => 'Pelanggan',
            'password' => Hash::make('12345'),
            'level_id' => 4,
        ];
        UserModel::insert($data);
        $user = UserModel::all(); // ambil semua data dari tabel m_user
        return view('user', ['data' => $user]);
    }
}
```

9. Jalankan di browser, amati dan laporkan apa yang terjadi



Data User

ID	Username	Nama	ID Level Pengguna
1	admin	Administrator	1
2	manager	Manager	2
3	staff	Staff/Kasir	3
6	customer-1	Pelanggan	5

10. Kita modifikasi lagi file UserController menjadi seperti berikut

```

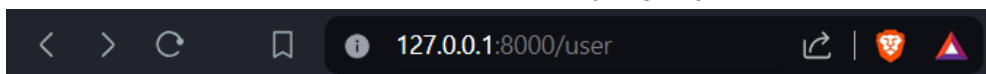
class UserController extends Controller
{
    public function index()
    {
        // $data = [
        //     'username' => 'customer-1',
        //     'nama' => 'Pelanggan',
        //     'password' => Hash::make('12345'),
        //     'level_id' => 5,
        // ];
        // UserModel::insert($data);

        $data = [
            'nama' => 'Pelanggan Pertama',
        ];
        UserModel::where('username', 'customer-1')->update($data);

        $user = UserModel::all(); // ambil semua data dari tabel m_user
        return view('user', ['data' => $user]);
    }
}

```

11. Jalankan di browser, amati dan laporkan apa yang terjadi



Data User

ID	Username	Nama	ID Level Pengguna
1	admin	Administrator	1
2	manager	Manager	2
3	staff	Staff/Kasir	3
6	customer-1	Pelanggan Pertama	5

- data pada tabel tersebut berubah sesuai dengan yang diinputkan di controller.

12. Jika sudah, laporkan hasil Praktikum-6 ini dan commit perubahan pada git



Pertanyaan Praktikum

1. Pada Praktikum 1 - Tahap 5, apakah fungsi dari APP_KEY pada file setting .env Laravel?
 - APP_KEY pada file .env Laravel adalah kunci rahasia yang digunakan untuk mengamankan data yang disimpan dalam session dan digunakan untuk enkripsi dan dekripsi data.
2. Pada Praktikum 1, bagaimana kita men-generate nilai untuk APP_KEY?
 - kita bisa men-generatannya menggunakan perintah
 - `php artisan key:generate`
3. Pada Praktikum 2.1 - Tahap 1, secara default Laravel memiliki berapa file migrasi? dan untuk apa saja file migrasi tersebut?

Terdapat 4 file migrasi, yaitu :

 1. `2014_10_12_000000_create_users_table.php`: untuk membuat tabel users.
 2. `2014_10_12_100000_create_password_reset_tokens_table.php`: untuk membuat tabel `password_resets_tokens` yang digunakan untuk menyimpan token yang diberikan kepada pengguna yang ingin mereset password mereka.
 3. `2019_08_19_000000_create_failed_jobs_table.php`: untuk membuat tabel `failed_jobs` yang berfungsi sebagai tempat menyimpan informasi tentang job yang gagal dieksekusi pada sistem aplikasi.
 4. `2019_12_14_000001_create_personal_access_tokens_table.php`: untuk membuat tabel `personal_access_tokens` yang digunakan untuk menyimpan data token akses pribadi untuk otentikasi.
4. Secara default, file migrasi terdapat kode `$table->timestamps();`, apa tujuan/output dari fungsi tersebut?
 - Fungsi `$table->timestamps()` pada file migrasi digunakan untuk membuat dua kolom tambahan dalam tabel basis data, yaitu `created_at` dan `updated_at`. Kolom `created_at` akan diisi dengan tanggal dan waktu ketika sebuah record dibuat, sedangkan kolom `updated_at` akan diisi dengan tanggal dan waktu ketika sebuah record diperbarui
5. Pada File Migrasi, terdapat fungsi `$table->id();` Tipe data apa yang dihasilkan dari fungsi tersebut?
 - UNSIGNED BIGINT
6. Apa bedanya hasil migrasi pada table `m_level`, antara menggunakan `$table->id();` dengan menggunakan `$table->id('level_id');` ?
 - Perbedaan antara menggunakan `$table->id();` dan `$table->id('level_id');` adalah pada nama kolom primary key yang dihasilkan. `$table->id();` akan membuat kolom primary key dengan nama `id`, sedangkan `$table->id('level_id');` akan membuat kolom primary key dengan nama `level_id`.
7. Pada migration, Fungsi `->unique()` digunakan untuk apa?
 - Fungsi `->unique()` pada migration digunakan untuk menetapkan kolom sebagai nilai yang unik, yang berarti nilainya harus unik atau tidak boleh ada nilai yang sama dalam kolom tersebut.
8. Pada Praktikum 2.2 - Tahap 2, kenapa kolom `level_id` pada tabel `m_user` menggunakan `$table->unsignedBigInteger('level_id')`, sedangkan kolom `level_id` pada tabel `m_level` menggunakan `$table->id('level_id')` ?

- Kolom `level_id` pada tabel `m_user` menggunakan `$table->unsignedBigInteger('level_id')` karena kolom ini berperan sebagai foreign key yang merujuk ke kolom `id` pada tabel `m_level`, Sedangkan kolom `level_id` pada tabel `m_level` menggunakan `$table->id('level_id')` sebagai primary key.
9. Pada Praktikum 3 - Tahap 6, apa tujuan dari Class Hash? dan apa maksud dari kode program `Hash::make('1234');`?
 - Penggunaan Class Hash pada kode program `Hash::make('1234');` adalah untuk enkripsi data informasi personal.
 10. Pada Praktikum 4 - Tahap 3/5/7, pada query builder terdapat tanda tanya (?), apa kegunaan dari tanda tanya (?) tersebut?
 - Kegunaan tanda tanya (?) pada query builder adalah sebagai placeholder untuk parameter yang akan diisi dan dieksekusi. Placeholder ini nantinya akan digantikan oleh nilai yang diberikan saat query dieksekusi. Tujuannya untuk membuat query aman terhadap serangan SQL injection dan kesalahan memasukan nilai.
 11. Pada Praktikum 6 - Tahap 3, apa tujuan penulisan kode `protected $table = 'm_user';` dan `protected $primaryKey = 'user_id';` ?
 - `protected $table = 'm_user';`. Baris ini digunakan untuk mendefinisikan nama tabel yang akan digunakan oleh model.
 - `protected $primaryKey = 'user_id';`. Baris ini digunakan untuk mendefinisikan primary key dari tabel yang digunakan oleh model.
 12. Menurut kalian, lebih mudah menggunakan mana dalam melakukan operasi CRUD ke database (DB Façade / Query Builder / Eloquent ORM) ? jelaskan
 - Eloquent ORM karena dapat melakukan query dengan sintaks yang lebih mudah dibaca dan dimengerti.