



Mata Kuliah : Pemrograman Web Lanjut (PWL)  
Program Studi : D4 – Teknik Informatika / D4 – Sistem Informasi Bisnis  
Semester : 4 (empat) / 6 (enam)  
Pertemuan ke- : 1 (satu)

## **JOBSHEET 03**

### **MIGRATION, SEEDER, DB FAÇADE, QUERY BUILDER, dan**

### **ELOQUENT ORM**

Sebelumnya kita sudah membahas mengenai *Routing*, *Controller*, dan *View* yang ada di Laravel. Sebelum kita masuk pada pembuatan aplikasi berbasis website, alangkah baiknya kita perlu menyiapkan Basis data sebagai tempat menyimpan data-data pada aplikasi kita nanti. Selain itu, umumnya kita perlu menyiapkan juga data awal yang kita gunakan sebelum membuat aplikasi, seperti data user administrator, data pengaturan sistem, dll.

Untuk itu, kita memerlukan teknik untuk merancang/membuat table basis data sebelum membuat aplikasi. Laravel memiliki fitur dalam pengelolaan basis data seperti, migration, seeder, model, dll.

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. 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<sup>1</sup>** - pengaturan database:

The screenshot shows the phpMyAdmin interface for a MySQL server on localhost. The 'Databases' tab is selected. A 'Create database' button is visible. In the input field, 'PWL\_POS' is typed. To the right of the input field is a 'Create' button.

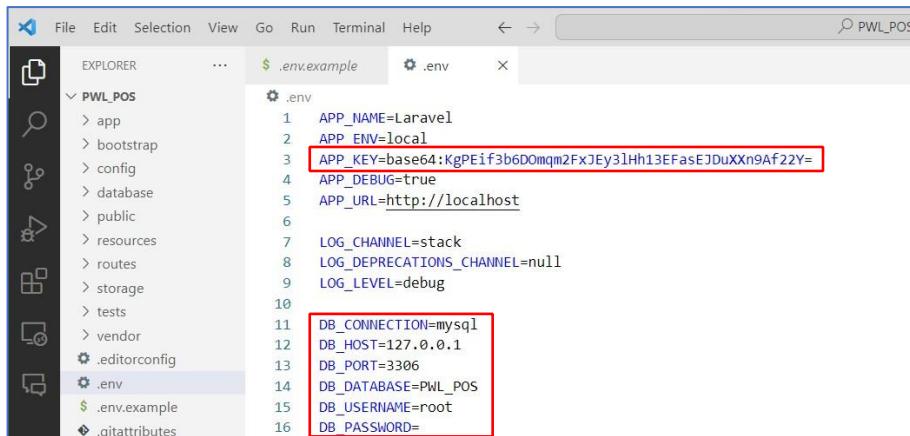
2. Buka aplikasi VSCode dan buka folder project **PWL\_POS** yang sudah kita buat
3. Copy file **.env.example** menjadi **.env**
4. Buka file **.env**, dan pastikan konfigurasi **APP\_KEY** bernilai. Jika belum bernilai silahkan kalian *generate* menggunakan **php artisan**.

The screenshot shows the VSCode code editor with the '.env' file open. The file contains the following Laravel configuration variables:

```
APP_NAME=Laravel
APP_ENV=local
APP_KEY=
APP_DEBUG=true
APP_URL=http://localhost
LOG_CHANNEL=stack
LOG_DEPRECATIONS_CHANNEL=null
LOG_LEVEL=debug
DB_CONNECTION=mysql
DB_HOST=127.0.0.1
DB_PORT=3306
DB_DATABASE=PWL_POS
DB_USERNAME=root
DB_PASSWORD=
```

5. Edit file **.env** dan sesuaikan dengan database yang telah dibuat

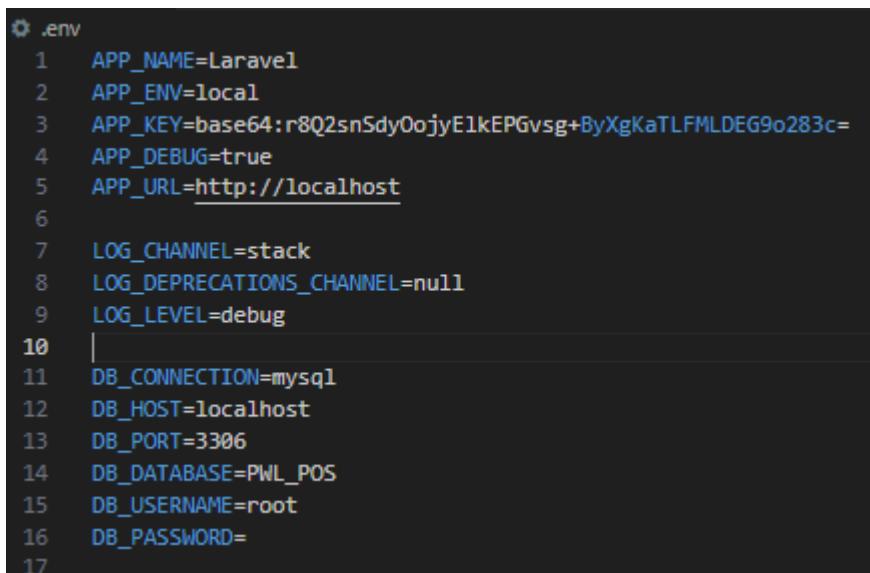
<sup>1</sup> . Buka aplikasi phpMyAdmin, dan buat database baru dengan nama **PWL\_POS**



```
File Edit Selection View Go Run Terminal Help ← → ⚡ PWL_POS
EXPLORER ... $ .env.example .env
PWL_POS
  app
  bootstrap
  config
  database
  public
  resources
  routes
  storage
  tests
  vendor
  .editorconfig
  .env
  .env.example
  .gitattributes

.env
1 APP_NAME=Laravel
2 APP_ENV=local
3 APP_KEY=base64:KgPEif3b6D0mqm2FxJEy3lHh13EFasEJDuXXn9Af22Y=
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=
17
```

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



```
.env
1 APP_NAME=Laravel
2 APP_ENV=local
3 APP_KEY=base64:r8Q2snSdyOojyElkEPGvsg+ByXgKaTLFMLDEG9o283c=
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=localhost
13 DB_PORT=3306
14 DB_DATABASE=PWL_POS
15 DB_USERNAME=root
16 DB_PASSWORD=
17
```

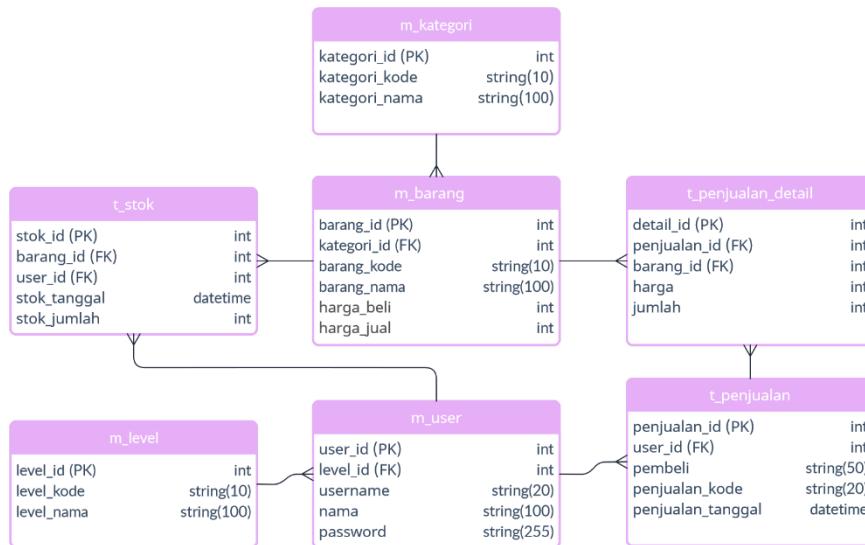
## B. MIGRATION

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.

Sesuai dengan topik pembelajaran kita untuk membangun sistem *Point of Sales (PoS)* sederhana, maka kita perlu membuat migration sesuai desain database yang sudah didefinisikan pada file

**Studi Kasus PWL.pdf**



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 dilanjut 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	<a href="#">m_level</a>	0
2	<a href="#">m_kategori</a>	0
3	<a href="#">m_user</a>	1
4	<a href="#">m_barang</a>	1
5	<a href="#">t_penjualan</a>	1
6	<a href="#">t_stok</a>	2
7	<a href="#">t_penjualan_detail</a>	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*

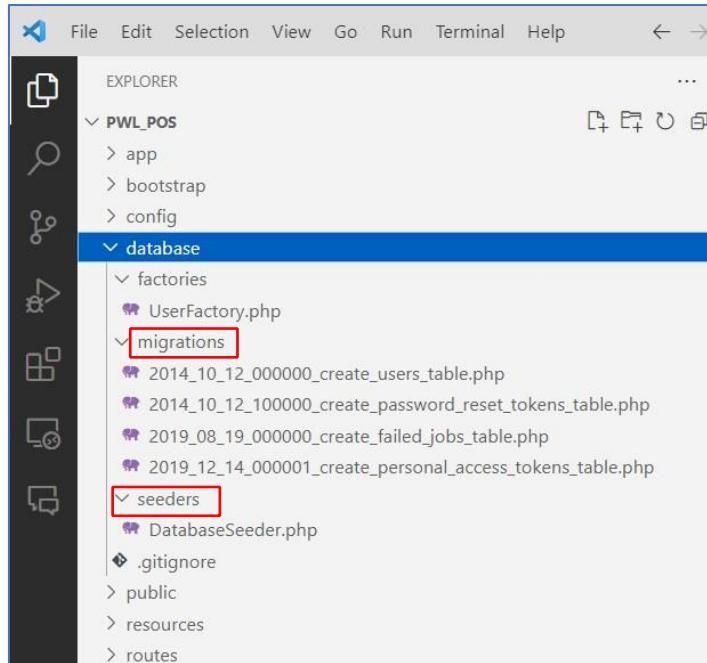
```
php artisan make:migration <nama-file-tabel> --create=<nama-tabel>
```

- 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](#)



### Praktikum 2.1 - Pembuatan file migrasi tanpa relasi

---

- Buka *terminal* VSCode kalian, untuk yang di kotak merah adalah default dari laravel



The screenshot shows a file explorer window with the following structure:

- PWL\_POS**
  - app
  - bootstrap
  - config
  - database
    - migrations
      - 2014\_10\_12\_000000\_create\_users\_table.php
      - 2014\_10\_12\_100000\_create\_password\_reset\_tokens\_table.php
      - 2019\_08\_19\_000000\_create\_failed\_jobs\_table.php
      - 2019\_12\_14\_000001\_create\_personal\_access\_tokens\_table.php
    - seeders
  - public
  - resources
  - routes
  - storage
  - tests
  - vendor

**.env**

```
APP_NAME=Laravel
APP_ENV=local
APP_KEY=base64:KgPEif3b6D0mqm2FxJEy3lHh13EF
APP_DEBUG=true
APP_URL=http://localhost
LOG_CHANNEL=stack
LOG_DEPRECATIONS_CHANNEL=null
LOG_LEVEL=debug
DB_CONNECTION=mysql
DB_HOST=127.0.0.1
DB_PORT=3306
DB_DATABASE=PWL_POS
DB_USERNAME=root
DB_PASSWORD=
BROADCAST_DRIVER=log
CACHE_DRIVER=file
```

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
```

The screenshot shows a code editor displaying a PHP migration file:

```
<?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::create('m_level', function (Blueprint $table) {
            $table->id();
            $table->timestamps();
        });
    }

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

**Hasil :**



```
PS C:\laragon\www\PWL2025\Week3\Jobsheet3> php artisan make:migration create_m_level_table --create=m_level
[INFO] Migration [C:\laragon\www\PWL2025\Week3\Jobsheet3\database\migrations\2025_03_05_121314_create_m_level_table.php] created successfully.

PS C:\laragon\www\PWL2025\Week3\Jobsheet3>

database > migrations > 2025_03_05_121314_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): void {
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  };
28
```

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

```
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  };
```

**Hasil :**



```
database > migrations > 2025_03_05_121314_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): void {
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 };
30 }
```

### INFO

Dalam fitur migration Laravel, terdapat berbagai macam function untuk membuat kolom di table database. Silahkan cek disini

<https://laravel.com/docs/10.x/migrations#available-column-types>

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



```
php artisan migrate
```

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS D:\laragon\www\PWL_POS> php artisan migrate
INFO Preparing database.
Creating migration table ..... 12ms DONE
INFO Running migrations.

2014_10_12_000000_create_users_table ..... 16ms DONE
2014_10_12_100000_create_password_reset_tokens_table ..... 6ms DONE
2019_08_19_000000_create_failed_jobs_table ..... 42ms DONE
2019_12_14_000001_create_personal_access_tokens_table ..... 15ms DONE
2024_02_25_133526_create_m_level_table ..... 13ms DONE

PS D:\laragon\www\PWL_POS>
```

Hasil :

```
PS C:\Laragon\www\PWL2025\Week3\Jobsheet3> php artisan migrate
INFO Running migrations.

2025_03_05_121314_create_m_level_table ..... 133ms DONE
```

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

Table	Action	Rows
failed_jobs		0
migrations		5
<b>m_level</b>		0
password_reset_tokens		0
personal_access_tokens		0
users		0

hasil :

Table	Action	Rows	Type	Collation	Size	O
failed_jobs		0	InnoDB	utf8mb4_unicode_ci	16.0 Kib	
migrations		5	InnoDB	utf8mb4_unicode_ci	16.0 Kib	
<b>m_level</b>		0	InnoDB	utf8mb4_unicode_ci	16.0 Kib	
password_reset_tokens		0	InnoDB	utf8mb4_unicode_ci	16.0 Kib	
personal_access_tokens		0	InnoDB	utf8mb4_unicode_ci	16.0 Kib	
users		0	InnoDB	utf8mb4_unicode_ci	16.0 Kib	
6 tables	Sum	5	InnoDB	utf8mb4_unicode_ci	96.0 Kib	

7. Ok, table sudah dibuat di database



8. Buat table *database* dengan *migration* untuk table **m\_kategori** yang sama-sama tidak memiliki *foreign key*

```
PS C:\laragon\www\PWL2025\Week3\Jobsheet3> php artisan make:migration create_m_kategori_table --create=m_kategori
```

```
INFO Migration [C:\laragon\www\PWL2025\Week3\Jobsheet3\database\migrations\2025_03_05_130456_create_m_kategori_table.php] created successfully.
```

```
database > migrations > 2025_03_05_130456_create_m_kategori_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_kategori', function (Blueprint $table): void {
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    /**
23     * Reverse the migrations.
24     */
25    public function down(): void
26    {
27        Schema::dropIfExists('m_kategori');
28    }
29};
```

```
PS C:\laragon\www\PWL2025\Week3\Jobsheet3> php artisan migrate --path=database/migrations/2025_03_05_130456_create_m_kategori_table.php
```

```
INFO Running migrations.
```

```
2025_03_05_130456_create_m_kategori_table ..... 131ms DONE
```

Table	Action	Rows	Type	Collation	Size	Overhead
failed_jobs		0	InnoDB	utf8mb4_unicode_ci	16.0 Kib	-
migrations		6	InnoDB	utf8mb4_unicode_ci	16.0 Kib	-
m_kategori		0	InnoDB	utf8mb4_unicode_ci	16.0 Kib	-
m_level		0	InnoDB	utf8mb4_unicode_ci	16.0 Kib	-
password_reset_tokens		0	InnoDB	utf8mb4_unicode_ci	16.0 Kib	-
personal_access_tokens		0	InnoDB	utf8mb4_unicode_ci	16.0 Kib	-
users		0	InnoDB	utf8mb4_unicode_ci	16.0 Kib	-
7 tables	Sum	6	InnoDB	utf8mb4_unicode_ci	112.0 Kib	0 B

9. Laporkan hasil Praktikum-2. ini dan commit perubahan pada git.



## Praktikum 2.<sup>2</sup> - Pembuatan file migrasi dengan relasi

1. Buka terminal VSCode kalian, dan buat file migrasi untuk table m\_user

```
php artisan make:migration create_m_user_table --table=m_user
```

Hasil :

```
PS C:\laragon\www\PWL2025\Week3\Jobsheet3> php artisan make:migration create_m_user_table --create=m_user

[INFO] Migration [C:\laragon\www\PWL2025\Week3\Jobsheet3\database\migrations\2025_03_05_132406_create_m_user_table.php] created successfully.
```

2. Buka file migrasi untuk table m\_user, dan modifikasi seperti berikut

```
7 return new class extends Migration
8 {
9     /**
10      * Run the migrations.
11      */
12     public function up(): void
13     {
14         Schema::create('m_user', function (Blueprint $table) {
15             $table->id('user_id');
16             $table->unsignedBigInteger('level_id')->index(); // indexing untuk ForeignKey
17             $table->string('username', 20)->unique(); // unique untuk memastikan tidak ada username yang sama
18             $table->string('nama', 100);
19             $table->string('password');
20             $table->timestamps();
21
22             // Mendefinisikan Foreign Key pada kolom level_id mengacu pada kolom level_id di tabel m_level
23             $table->foreign('level_id')->references('level_id')->on('m_level');
24         });
25     }
26
27     /**
28      * Reverse the migrations.
29      */
30     public function down(): void
31     {
32         Schema::dropIfExists('m_user');
33     }
34 };
```

Hasil :



```
database > migrations > 2025_03_05_132406_create_m_user_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_user', function (Blueprint $table): void {
15             $table->id('user_id');
16             $table->unsignedBigInteger('level_id')->index(); //indexing untuk ForeignKey
17             $table->string('username', 20)->unique(); //unique untuk memastikan tidak ada username yang sama
18             $table->string('nama', 100);
19             $table->string('password');
20             $table->timestamps();
21
22             //Mendefinisikan Foreign Key pada kolom levelid mengacu pada kolom level_id di tabel m_level
23             $table->foreign('level_id')->references('level_id')->onDelete('cascade');
24         });
25     }
26
27     /**
28     * Reverse the migrations.
29     */
30     public function down(): void
31     {
32         Schema::dropIfExists('m_user');
33     }
34 };
```

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

```
PS C:\laragon\www\PWL2025\Week3\Jobsheet3> php artisan migrate --path=database
/migrations/2025_03_05_132406_create_m_user_table.php

[INFO] Running migrations.

2025_03_05_132406_create_m_user_table 333ms DONE
```

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

m_barang
t_penjualan
t_stok
t_penjualan_detail

Hasil :

- m\_barang

```
PS C:\laragon\www\PWL2025\Week3\Jobsheet3> php artisan make:migration create_m
_barang_table --create=m_barang

[INFO] Migration [C:\laragon\www\PWL2025\Week3\Jobsheet3\database\migrations
/2025_03_05_133522_create_m_barang_table.php] created successfully.
```



```
database > migrations > 2025_03_05_133522_create_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::create('m_barang', function (Blueprint $table): void {
15            $table->id('barang_id');
16            $table->unsignedBigInteger('kategori_id')->index(); // indexing untuk ForeignKey
17            $table->string('barang_kode', 10)->unique();
18            $table->string('barang_nama', 100);
19            $table->integer('harga_beli');
20            $table->integer('harga_jual');
21            $table->timestamps();
22
23            $table->foreign('kategori_id')->references('kategori_id')->on('m_kategori');
24        });
25    }
26
27    /**
28     * Reverse the migrations.
29     */
30    public function down(): void
31    {
32        Schema::dropIfExists('m_barang');
33    }
34});
```

```
PS C:\laragon\www\PWL2025\Week3\Jobsheet3> php artisan migrate --path=database/migrations/2025_03_05_133522_create_m_barang_table.php
[INFO] Running migrations.
2025_03_05_133522_create_m_barang_table ..... 294ms DONE
```

- t\_penjualan

```
PS C:\laragon\www\PWL2025\Week3\Jobsheet3> php artisan make:migration create_t_penjualan_table --create=t_penjualan
[INFO] Migration [C:\laragon\www\PWL2025\Week3\Jobsheet3\database\migrations\2025_03_05_134129_create_t_penjualan_table.php] created successfully.
```



```
database > migrations > 2025_03_05_134129_create_t_penjualan_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('t_penjualan', function (Blueprint $table): void {
15             $table->id('penjualan_id');
16             $table->unsignedBigInteger('user_id')->index(); // indexing untuk ForeignKey
17             $table->string('pembeli', 20);
18             $table->string('penjualan_kode', 20);
19             $table->string('penjualan_tanggal', );
20             $table->timestamps();
21
22             // Mendefinisikan Foreign Key pada kolom level_id mengacu pada kolom user_id di tabel m_user
23             $table->foreign('user_id')->references('user_id')->on('m_user');
24         });
25     }
26
27     /**
28     * Reverse the migrations.
29     */
30     public function down(): void
31     {
32         Schema::dropIfExists('t_penjualan');
33     }
34 }
```

```
PS C:\laragon\www\PWL2025\Week3\Jobsheet3> php artisan migrate --path=database/migrations/2025_03_05_134129_create_t_penjualan_table.php
[INFO] Running migrations.
2025_03_05_134129_create_t_penjualan_table ..... 459ms DONE
```

- t\_stok

```
PS C:\laragon\www\PWL2025\Week3\Jobsheet3> php artisan make:migration create_t_stok_table --create=t_stok
[INFO] Migration [C:\laragon\www\PWL2025\Week3\Jobsheet3\database\migrations\2025_03_05_134600_create_t_stok_table.php] created successfully.
```



```
database > migrations > 2025_03_05_134600_create_t_stok_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('t_stok', function (Blueprint $table): void {
15            $table->id('stok_id');
16            $table->unsignedBigInteger('barang_id')->index();
17            $table->unsignedBigInteger('user_id')->index();
18            $table->datetime('stok_tanggal');
19            $table->integer('stok_jumlah');
20            $table->timestamps();
21
22            // foreignkey
23            $table->foreign('barang_id')->references('barang_id')->on('m_barang');
24            $table->foreign('user_id')->references('user_id')->on('m_user');
25
26        });
27    }
28
29    /**
30     * Reverse the migrations.
31     */
32    public function down(): void
33    {
34        Schema::dropIfExists('t_stok');
35    }
36};
```

```
PS C:\laragon\www\PWL2025\Week3\Jobsheet3> php artisan migrate --path=database/migrations/2025_03_05_134600_create_t_stok_table.php
[INFO] Running migrations.
[INFO] Migrating: 2025_03_05_134600_create_t_stok_table ..... 372ms DONE
```

- t\_penjualan\_detail

```
PS C:\laragon\www\PWL2025\Week3\Jobsheet3> php artisan make:migration create_t_penjualan_detail_table --create=t_penjualan_detail
[INFO] Migration [C:\laragon\www\PWL2025\Week3\Jobsheet3\database\migrations/2025_03_05_134924_create_t_penjualan_detail_table.php] created successfully.
```

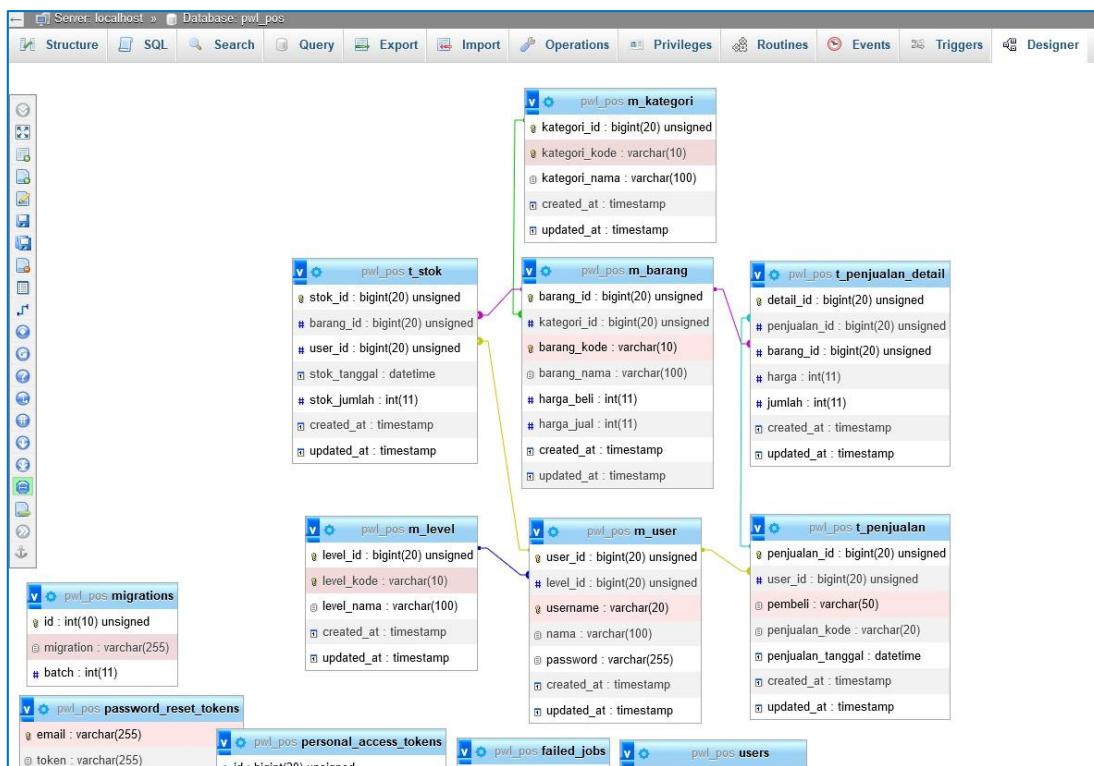


```
database > migrations > 2025_03_05_134924_create_t_penjualan_detail_table.php > class > up
1  <?php
2
3  use Illuminate\Database\Migrations\Migration;
4  use Illuminate\Database\Schema\Blueprint;
5  use Illuminate\Support\Facades\Schema;
6
7  return new class extends Migration
8  {
9      /**
10      * Run the migrations.
11      */
12      public function up(): void
13  {
14          Schema::create('t_penjualan_detail', function (Blueprint $table): void {
15              $table->bigIncrements('detail_id');
16              $table->unsignedBigInteger('penjualan_id')->index(); // indexing untuk ForeignKey
17              $table->unsignedBigInteger('barang_id')->index(); // indexing untuk ForeignKey
18              $table->integer('harga');
19              $table->integer('jumlah');
20              $table->timestamps(); // otomatis membuat kolom created_at dan updated_at
21
22              // Mendefinisikan Foreign Key
23              $table->foreign('penjualan_id')->references('penjualan_id')->on('t_penjualan');
24              $table->foreign('barang_id')->references('barang_id')->on('m_barang');
25          });
26      }
27  }
28
29      /**
30      * Reverse the migrations.
31      */
32      public function down(): void
33  {
34          Schema::dropIfExists('t_penjualan_detail');
35  }
36 };
```

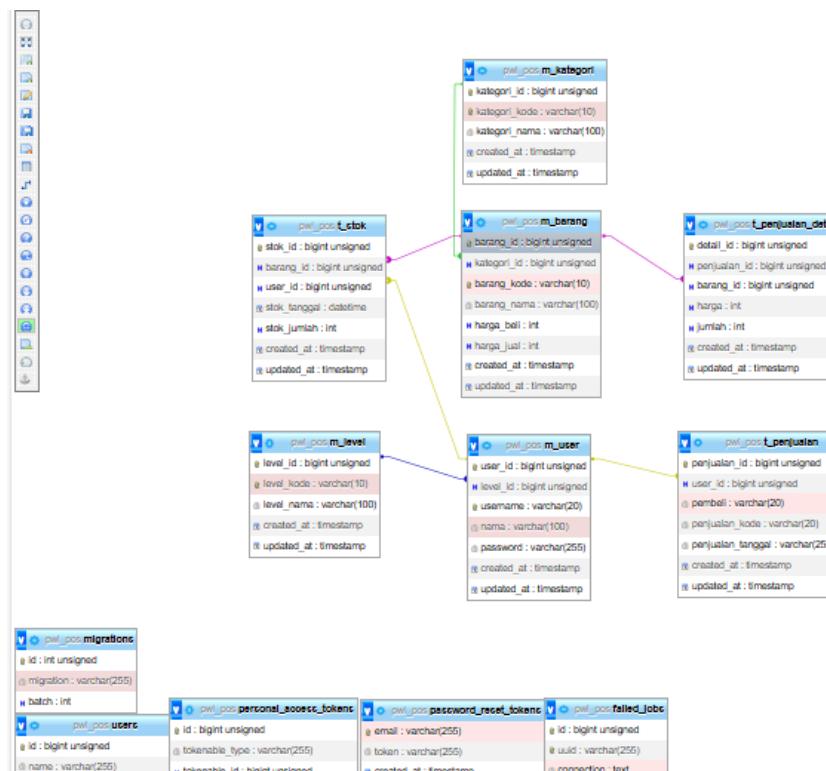
```
PS C:\laragon\www\PWL2025\Week3\Jobsheet3> php artisan migrate --path=database/migrations/2025_03_05_134924_create_t_penjualan_detail_table.php
[INFO] Running migrations.

2025_03_05_134924_create_t_penjualan_detail_table ..... 262ms DONE
```

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



### Hasil :



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



## C. SEEDER

Seeder merupakan sebuah fitur yang memungkinkan kita untuk mengisi database kita dengan data awal atau data *dummy* yang telah ditentukan. Seeder memungkinkan kita untuk membuat data awal yang sama untuk setiap penggunaan dalam pembangunan aplikasi. Umumnya, data yang sering dibuat *seeder* adalah data pengguna karena data tersebut akan digunakan saat aplikasi pertama kali di jalankan dan membutuhkan aksi *login*.

1. Perintah umum dalam **membuat file seeder** adalah seperti berikut

```
php artisan make:seeder <nama-class-seeder>
```

Perintah tersebut akan men-generate file seeder pada folder **PWL\_POS/database/seeders**

2. Dan perintah untuk **menjalankan file seeder** seperti berikut

```
php artisan db:seed --class=<nama-class-seeder>
```

Dalam proses pengembangan suatu aplikasi, seringkali kita membutuhkan data awal tiruan atau *dummy* data untuk memudahkan pengujian dan pengembangan aplikasi kita.

Sehingga fitur *seeder* bisa kita pakai dalam membuat sebuah aplikasi web.

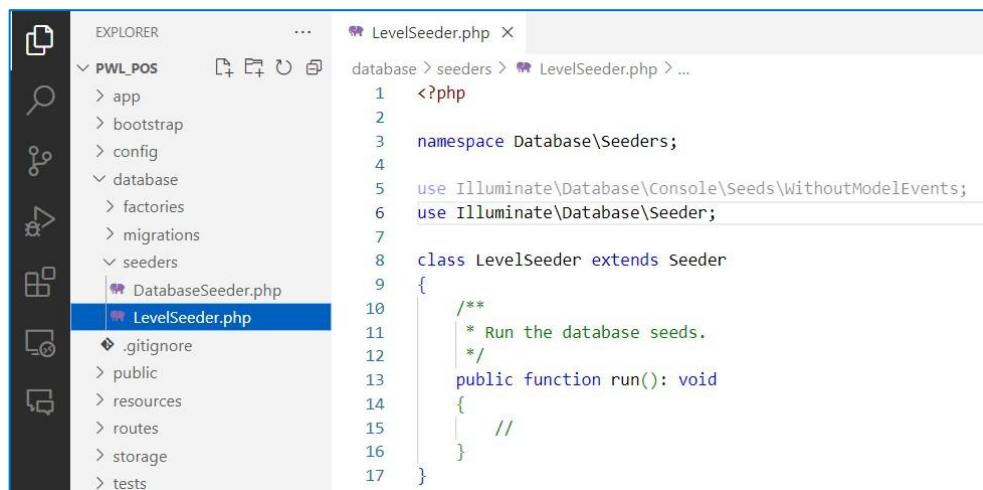
### Praktikum 3 – Membuat file seeder

---

1. Kita akan membuat file seeder untuk table **m\_level** dengan mengetikkan perintah



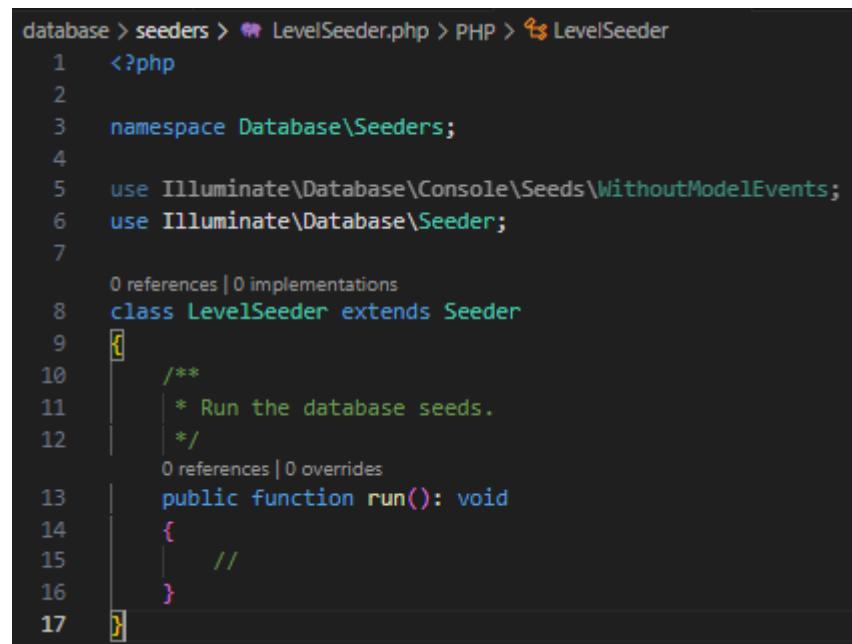
```
php artisan make:seeder LevelSeeder
```



```
database > seeders > LevelSeeder.php <?php
namespace Database\Seeders;
use Illuminate\Database\Console\Seeds\WithoutModelEvents;
use Illuminate\Database\Seeder;
class LevelSeeder extends Seeder
{
    /**
     * Run the database seeds.
     */
    public function run(): void
    {
        //
    }
}
```

### Hasil :

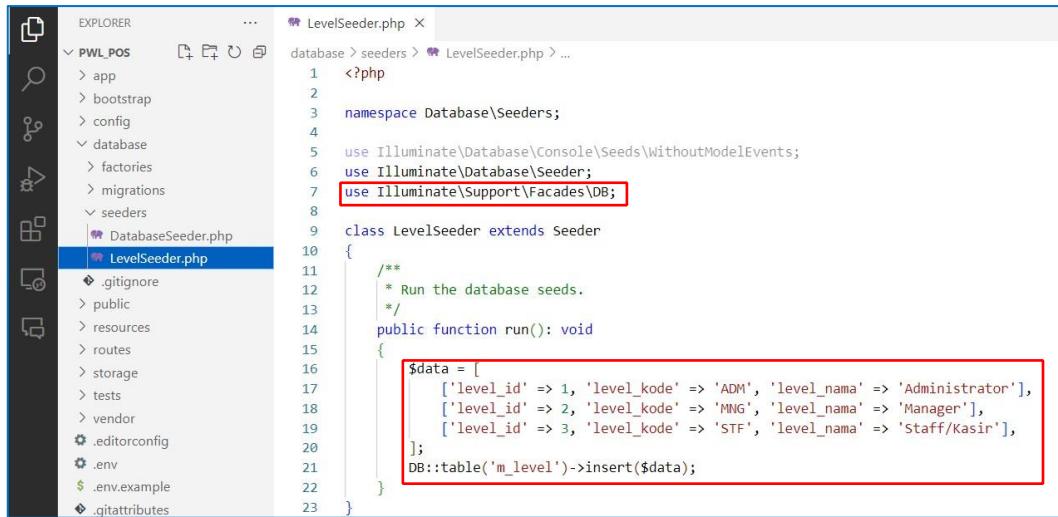
```
PS C:\laragon\www\PWL2025\Week3\Jobsheet3> php artisan make:seeder LevelSeeder
INFO Seeder [C:\laragon\www\PWL2025\Week3\Jobsheet3\database\seeders\LevelSeeder.php] created successfully.
```



```
database > seeders > LevelSeeder.php <?php
namespace Database\Seeders;
use Illuminate\Database\Console\Seeds\WithoutModelEvents;
use Illuminate\Database\Seeder;
class LevelSeeder extends Seeder
{
    /**
     * Run the database seeds.
     */
    public function run(): void
    {
        //
    }
}
```

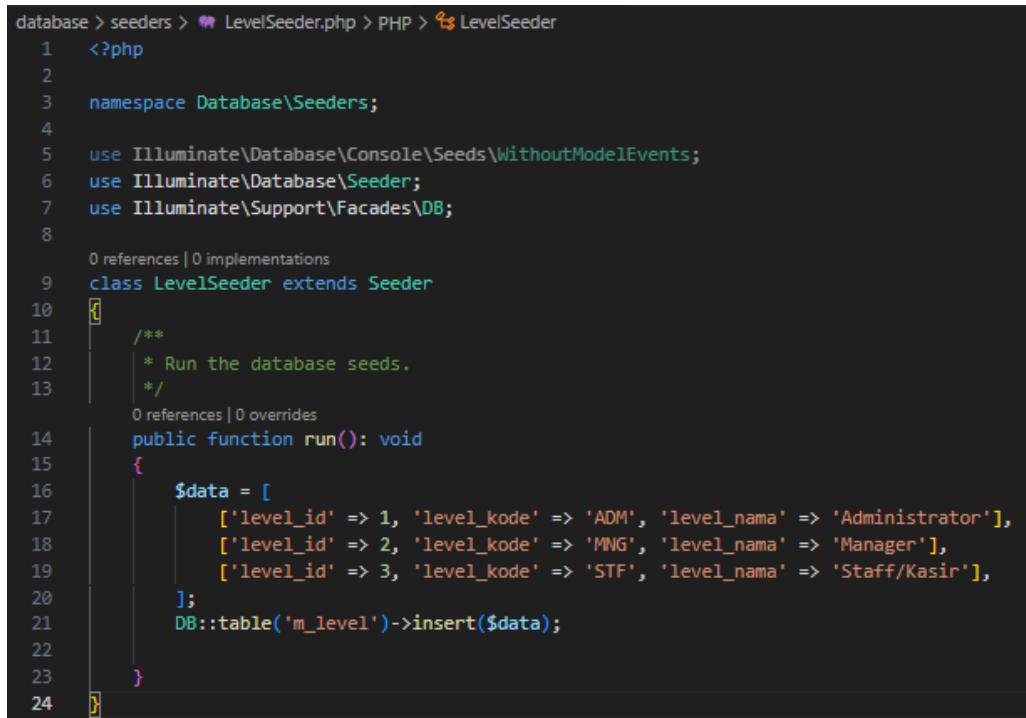


2. Selanjutnya, untuk memasukkan data awal, kita modifikasi file tersebut di dalam function `run()`



```
EXPLORER ... LevelSeeder.php X
PWL_POS database > seeders > LevelSeeder.php > ...
<?php
namespace Database\Seeders;
use Illuminate\Database\Console\Seeds\WithoutModelEvents;
use Illuminate\Database\Seeder;
use Illuminate\Support\Facades\DB;
class LevelSeeder extends Seeder
{
    /**
     * Run the database seeds.
     */
    public function run(): void
    {
        $data = [
            ['level_id' => 1, 'level_kode' => 'ADM', 'level_nama' => 'Administrator'],
            ['level_id' => 2, 'level_kode' => 'MNG', 'level_nama' => 'Manager'],
            ['level_id' => 3, 'level_kode' => 'STF', 'level_nama' => 'Staff/Kasir'],
        ];
        DB::table('m_level')->insert($data);
    }
}
```

**Hasil :**



```
database > seeders > LevelSeeder.php > PHP > LevelSeeder
<?php
namespace Database\Seeders;
use Illuminate\Database\Console\Seeds\WithoutModelEvents;
use Illuminate\Database\Seeder;
use Illuminate\Support\Facades\DB;
class LevelSeeder extends Seeder
{
    /**
     * Run the database seeds.
     */
    public function run(): void
    {
        $data = [
            ['level_id' => 1, 'level_kode' => 'ADM', 'level_nama' => 'Administrator'],
            ['level_id' => 2, 'level_kode' => 'MNG', 'level_nama' => 'Manager'],
            ['level_id' => 3, 'level_kode' => 'STF', 'level_nama' => 'Staff/Kasir'],
        ];
        DB::table('m_level')->insert($data);
    }
}
```

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



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

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS  
PS D:\_laragon\www\PWL_POS> php artisan db:seed --class=LevelSeeder  
INFO Seeding database.  
PS D:\_laragon\www\PWL_POS>
```

### Hasil :

```
PS C:\laragon\www\PWL2025\Week3\Jobsheet3> php artisan d:seed --class=LevelSeeder  
INFO Seeding database.
```

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

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

### Hasil :

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

### Hasil :

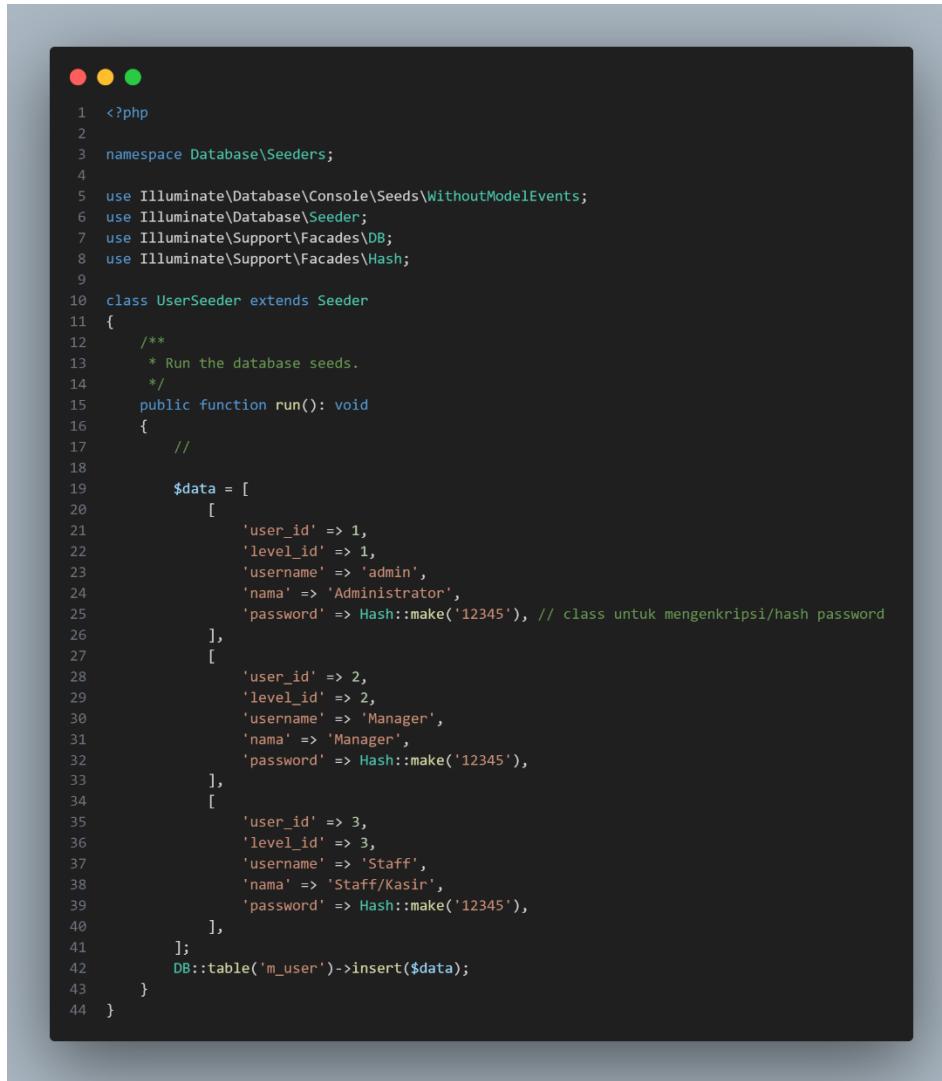
```
PS C:\laragon\www\PWL2025\Week3\Jobsheet3> php artisan make:seeder UserSeeder  
INFO Seeder [C:\laragon\www\PWL2025\Week3\Jobsheet3\database\seeders\UserSeeder.php] created successfully.
```



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 }
```

Hasil :



```
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 use Illuminate\Support\Facades\Hash;
9
10 class UserSeeder extends Seeder
11 {
12     /**
13      * Run the database seeds.
14      */
15     public function run(): void
16     {
17         //
18
19         $data = [
20             [
21                 'user_id' => 1,
22                 'level_id' => 1,
23                 'username' => 'admin',
24                 'nama' => 'Administrator',
25                 'password' => Hash::make('12345'), // class untuk mengenkripsi/hash password
26             ],
27             [
28                 'user_id' => 2,
29                 'level_id' => 2,
30                 'username' => 'Manager',
31                 'nama' => 'Manager',
32                 'password' => Hash::make('12345'),
33             ],
34             [
35                 'user_id' => 3,
36                 'level_id' => 3,
37                 'username' => 'Staff',
38                 'nama' => 'Staff/Kasir',
39                 'password' => Hash::make('12345'),
40             ],
41         ];
42         DB::table('m_user')->insert($data);
43     }
44 }
```

7. Jalankan perintah untuk mengeksekusi class `UserSeeder`

```
PS C:\laragon\www\PWL2025\Week3\Jobsheet3> php artisan db:seed --class=UserSeeder
INFO Seeding database.
```

8. Perhatikan hasil seeder pada tabel `m_user`



		user_id	level_id	username	nama	password		
<input type="checkbox"/>	Edit  Copy  Delete	1	1	admin	Administrator	\$2y\$12\$Tevu4dDO1CUAQpeM6H.Vp.LySwhY.4oAKU7FzwS6fXV...		
<input type="checkbox"/>	Edit  Copy  Delete	2	2	manager	Manager	\$2y\$12\$Ajfnz20/FdPTeUgghz31muEhlFaruLxkh5wvZ9NGRp...		
<input type="checkbox"/>	Edit  Copy  Delete	3	3	staff	Staff/Kasir	\$2y\$12\$Gi23TqGclW5pYeR0VL4o5OxPwb3Osk99VMy/BHnbJ9W...		

### Hasil :

		user_id	level_id	username	nama	password	created_at	updated_at
<input type="checkbox"/>	Edit  Copy  Delete	1	1	admin	Administrator	\$2y\$12\$yXRzPmKWJcqra5OBAmxpu6r8JmUWc5wP.c9RJ/K2C1...	NULL	NULL
<input type="checkbox"/>	Edit  Copy  Delete	2	2	Manager	Manager	\$2y\$12\$a.A.KKkeD1Sd249ot5lFH6uQTO0/DGZ4Ly4YavnD3VKU...	NULL	NULL
<input type="checkbox"/>	Edit  Copy  Delete	3	3	Staff	Staff/Kasir	\$2y\$12\$vjUxiaoYHkSy/AkojKF62.vQtToUzhFv46bDoOG/Ux2...	NULL	NULL

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

### Hasil :

- m\_kategori

```
PS C:\laragon\www\PWL2025\Week3\Jobsheet3> php artisan make:seeder Kategori
Seeder

INFO Seeder [C:\laragon\www\PWL2025\Week3\Jobsheet3\database\seeders\Ka
tegoriSeeder.php] created successfully.
```



```
database > seeders > KategoriSeeder.php > PHP Intelephense > KategoriSeeder
1  <?php
2
3  namespace Database\Seeders;
4
5  use Illuminate\Database\Seeder;
6  use Illuminate\Support\Facades\DB;
7
8      0 references | 0 implementations
9  class KategoriSeeder extends Seeder
10 {
11     /**
12      * Run the database seeds.
13      */
14     0 references | 0 overrides
15     public function run(): void
16     {
17         $data = [
18             ['kategori_id' => 1, 'kategori_kode' => 'KAT001', 'kategori_nama' => 'Makanan'],
19             ['kategori_id' => 2, 'kategori_kode' => 'KAT002', 'kategori_nama' => 'MakeUp'],
20             ['kategori_id' => 3, 'kategori_kode' => 'KAT003', 'kategori_nama' => 'Aksesoris'],
21             ['kategori_id' => 4, 'kategori_kode' => 'KAT004', 'kategori_nama' => 'Pewangi'],
22             ['kategori_id' => 5, 'kategori_kode' => 'KAT005', 'kategori_nama' => 'Alat Mandi'],
23         ];
24
25         // Insert ke tabel `m_kategori`
26         DB::table('m_kategori')->insert($data);
27     }
28 }
```

```
PS C:\laragon\www\PWL2025\Week3\Jobsheet3> php artisan db:seed --class=KategoriSeeder
```

```
INFO Seeding database.
```

		kategori_id	kategori_kode	kategori_nama	created_at	updated_at
<input type="checkbox"/>		1	KAT001	Makanan	NULL	NULL
<input type="checkbox"/>		2	KAT002	MakeUp	NULL	NULL
<input type="checkbox"/>		3	KAT003	Aksesoris	NULL	NULL
<input type="checkbox"/>		4	KAT004	Pewangi	NULL	NULL
<input type="checkbox"/>		5	KAT005	Alat Mandi	NULL	NULL

- m\_barang

```
PS C:\laragon\www\PWL2025\Week3\Jobsheet3> php artisan make:seeder BarangSeeder
```

```
INFO Seeder [C:\laragon\www\PWL2025\Week3\Jobsheet3\database\seeders\BarangSeeder.php] created successfully.
```



```
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 BarangSeeder extends Seeder
10 {
11     /**
12      * Run the database seeds.
13      */
14     public function run(): void
15     {
16         $data = [
17             [
18                 'barang_id' => 1,
19                 'barang_kode' => 'BRG001',
20                 'barang_nama' => 'Kulkas',
21                 'harga_beli' => 5000000,
22                 'harga_jual' => 7000000,
23                 'kategori_id' => 1,
24             ],
25             [
26                 'barang_id' => 2,
27                 'barang_kode' => 'BRG002',
28                 'barang_nama' => 'Chopper',
29                 'harga_beli' => 3000000,
30                 'harga_jual' => 4000000,
31                 'kategori_id' => 1,
32             ],
33             [
34                 'barang_id' => 3,
35                 'barang_kode' => 'BRG003',
36                 'barang_nama' => 'Jaket',
37                 'harga_beli' => 50000,
38                 'harga_jual' => 100000,
39                 'kategori_id' => 2,
40             ],
41             [
42                 'barang_id' => 4,
43                 'barang_kode' => 'BRG004',
44                 'barang_nama' => 'Maskara',
45                 'harga_beli' => 100000,
46                 'harga_jual' => 200000,
47                 'kategori_id' => 2,
48             ],
49             [
50                 'barang_id' => 5,
51                 'barang_kode' => 'BRG005',
52                 'barang_nama' => 'Pensil',
53                 'harga_beli' => 2000,
54                 'harga_jual' => 5000,
55                 'kategori_id' => 3,
56             ],
57             [
58                 'barang_id' => 6,
59                 'barang_kode' => 'BRG006',
60                 'barang_nama' => 'Buku Tulis',
61                 'harga_beli' => 5000,
62                 'harga_jual' => 10000,
63                 'kategori_id' => 3,
64             ],
65             [
66                 'barang_id' => 7,
67                 'barang_kode' => 'BRG007',
68                 'barang_nama' => 'Kerupuk',
69                 'harga_beli' => 1000,
70                 'harga_jual' => 2000,
71                 'kategori_id' => 4,
72             ],
73             [
74                 'barang_id' => 8,
75                 'barang_kode' => 'BRG008',
76                 'barang_nama' => 'Nasi',
77                 'harga_beli' => 5000,
78                 'harga_jual' => 10000,
79                 'kategori_id' => 4,
80             ],
81             [
82                 'barang_id' => 9,
83                 'barang_kode' => 'BRG009',
84                 'barang_nama' => 'Jepit Rambut',
85                 'harga_beli' => 3000,
86                 'harga_jual' => 5000,
87                 'kategori_id' => 5,
88             ],
89             [
90                 'barang_id' => 10,
91                 'barang_kode' => 'BRG010',
92                 'barang_nama' => 'Sabun',
93                 'harga_beli' => 5000,
94                 'harga_jual' => 8000,
95                 'kategori_id' => 5,
96             ],
97         ];
98         DB::table('m_barang')->insert($data);
99     }
100 }
```



```
PS C:\laragon\www\PWL2025\Week3\Jobsheet3> php artisan db:seed --class=BarangSeeder
INFO  Seeding database.
```

		barang_id	kategori_id	barang_kode	barang_nama	harga_beli	harga_jual	created_at	updated_at
<input type="checkbox"/>	Edit  Copy  Delete	1	1	BRG001	Kulkas	5000000	7000000	NULL	NULL
<input type="checkbox"/>	Edit  Copy  Delete	2	1	BRG002	Chopper	3000000	4000000	NULL	NULL
<input type="checkbox"/>	Edit  Copy  Delete	3	2	BRG003	Jaket	50000	100000	NULL	NULL
<input type="checkbox"/>	Edit  Copy  Delete	4	2	BRG004	Maskara	100000	200000	NULL	NULL
<input type="checkbox"/>	Edit  Copy  Delete	5	3	BRG005	Pensil	2000	5000	NULL	NULL
<input type="checkbox"/>	Edit  Copy  Delete	6	3	BRG006	Buku Tulis	5000	10000	NULL	NULL
<input type="checkbox"/>	Edit  Copy  Delete	7	4	BRG007	Kerupuk	1000	2000	NULL	NULL
<input type="checkbox"/>	Edit  Copy  Delete	8	4	BRG008	Nasi	5000	10000	NULL	NULL
<input type="checkbox"/>	Edit  Copy  Delete	9	5	BRG009	Jepit Rambut	3000	5000	NULL	NULL
<input type="checkbox"/>	Edit  Copy  Delete	10	5	BRG010	Sabun	5000	8000	NULL	NULL

- t\_stok

```
PS C:\laragon\www\PWL2025\Week3\Jobsheet3> php artisan make:seeder StokSeeder
INFO  Seeder [C:\laragon\www\PWL2025\Week3\Jobsheet3\database\seeders\StokSeeder.php] created successfully.
```



```
● ● ●
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 use function Laravel\Prompts\table;
10
11 class StokSeeder extends Seeder
12 {
13     /**
14      * Run the database seeds.
15      */
16     public function run(): void
17     {
18         $data = [
19             [
20                 'stok_id' => 1,
21                 'stok_jumlah' => 100,
22                 'stok_tanggal' => '2024-02-27 08:00:00',
23                 'barang_id' => 1,
24                 'user_id' => 1,
25             ],
26             [
27                 'stok_id' => 2,
28                 'stok_jumlah' => 120,
29                 'stok_tanggal' => '2024-02-27 08:00:00',
30                 'barang_id' => 2,
31                 'user_id' => 1,
32             ],
33             [
34                 'stok_id' => 3,
35                 'stok_jumlah' => 10,
36                 'stok_tanggal' => '2024-02-27 08:00:00',
37                 'barang_id' => 3,
38                 'user_id' => 1,
39             ],
40             [
41                 'stok_id' => 4,
42                 'stok_jumlah' => 100,
43                 'stok_tanggal' => '2024-02-27 08:00:00',
44                 'barang_id' => 4,
45                 'user_id' => 2,
46             ],
47             [
48                 'stok_id' => 5,
49                 'stok_jumlah' => 100,
50                 'stok_tanggal' => '2024-02-27 08:00:00',
51                 'barang_id' => 5,
52                 'user_id' => 2,
53             ],
54             [
55                 'stok_id' => 6,
56                 'stok_jumlah' => 100,
57                 'stok_tanggal' => '2024-02-27 08:00:00',
58                 'barang_id' => 6,
59                 'user_id' => 2,
60             ],
61             [
62                 'stok_id' => 7,
63                 'stok_jumlah' => 100,
64                 'stok_tanggal' => '2024-02-27 08:00:00',
65                 'barang_id' => 7,
66                 'user_id' => 3,
67             ],
68             [
69                 'stok_id' => 8,
70                 'stok_jumlah' => 100,
71                 'stok_tanggal' => '2024-02-27 08:00:00',
72                 'barang_id' => 8,
73                 'user_id' => 3,
74             ],
75             [
76                 'stok_id' => 9,
77                 'stok_jumlah' => 100,
78                 'stok_tanggal' => '2024-02-27 08:00:00',
79                 'barang_id' => 9,
80                 'user_id' => 3,
81             ],
82             [
83                 'stok_id' => 10,
84                 'stok_jumlah' => 100,
85                 'stok_tanggal' => '2024-02-27 08:00:00',
86                 'barang_id' => 10,
87                 'user_id' => 1,
88             ],
89         ];
90         DB::table('t_stok')->insert($data);
91     }
92 }
```



```
PS C:\laragon\www\PWL2025\Week3\Jobsheet3> php artisan db:seed --class=StokSeeder
```

**INFO** Seeding database.

	<input type="button" value="Edit"/>	<input type="button" value="Copy"/>	<input type="button" value="Delete"/>	stok_id	barang_id	user_id	stok_tanggal	stok_jumlah	created_at	updated_at
<input type="checkbox"/>	<input type="button" value="Edit"/>	<input type="button" value="Copy"/>	<input type="button" value="Delete"/>	1	1	1	2024-02-27 08:00:00	100	NULL	NULL
<input type="checkbox"/>	<input type="button" value="Edit"/>	<input type="button" value="Copy"/>	<input type="button" value="Delete"/>	2	2	1	2024-02-27 08:00:00	120	NULL	NULL
<input type="checkbox"/>	<input type="button" value="Edit"/>	<input type="button" value="Copy"/>	<input type="button" value="Delete"/>	3	3	1	2024-02-27 08:00:00	10	NULL	NULL
<input type="checkbox"/>	<input type="button" value="Edit"/>	<input type="button" value="Copy"/>	<input type="button" value="Delete"/>	4	4	2	2024-02-27 08:00:00	100	NULL	NULL
<input type="checkbox"/>	<input type="button" value="Edit"/>	<input type="button" value="Copy"/>	<input type="button" value="Delete"/>	5	5	2	2024-02-27 08:00:00	100	NULL	NULL
<input type="checkbox"/>	<input type="button" value="Edit"/>	<input type="button" value="Copy"/>	<input type="button" value="Delete"/>	6	6	2	2024-02-27 08:00:00	100	NULL	NULL
<input type="checkbox"/>	<input type="button" value="Edit"/>	<input type="button" value="Copy"/>	<input type="button" value="Delete"/>	7	7	3	2024-02-27 08:00:00	100	NULL	NULL
<input type="checkbox"/>	<input type="button" value="Edit"/>	<input type="button" value="Copy"/>	<input type="button" value="Delete"/>	8	8	3	2024-02-27 08:00:00	100	NULL	NULL
<input type="checkbox"/>	<input type="button" value="Edit"/>	<input type="button" value="Copy"/>	<input type="button" value="Delete"/>	9	9	3	2024-02-27 08:00:00	100	NULL	NULL
<input type="checkbox"/>	<input type="button" value="Edit"/>	<input type="button" value="Copy"/>	<input type="button" value="Delete"/>	10	10	1	2024-02-27 08:00:00	100	NULL	NULL

- **t\_penjualan**

```
PS C:\laragon\www\PWL2025\Week3\Jobsheet3> php artisan make:seeder PenjualanSeeder
```

**INFO** Seeder [C:\laragon\www\PWL2025\Week3\Jobsheet3\database\seeders\PenjualanSeeder.php] created successfully.



```
● ● ●
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 PenjualanSeeder extends Seeder
10 {
11     /**
12      * Run the database seeds.
13      */
14     public function run(): void
15     {
16         $data = [
17             [
18                 'penjualan_id' => 1,
19                 'pembeli' => 'John Doe',
20                 'penjualan_kode' => 'PJM001',
21                 'penjualan_tanggal' => '2024-02-27 08:00:00',
22                 'user_id' => 1,
23             ],
24             [
25                 'penjualan_id' => 2,
26                 'pembeli' => 'Jane Smith',
27                 'penjualan_kode' => 'PJM002',
28                 'penjualan_tanggal' => '2024-02-27 08:30:00',
29                 'user_id' => 1,
30             ],
31             [
32                 'penjualan_id' => 3,
33                 'pembeli' => 'Michael Johnson',
34                 'penjualan_kode' => 'PJM003',
35                 'penjualan_tanggal' => '2024-02-27 09:00:00',
36                 'user_id' => 1,
37             ],
38             [
39                 'penjualan_id' => 4,
40                 'pembeli' => 'Emily Brown',
41                 'penjualan_kode' => 'PJM004',
42                 'penjualan_tanggal' => '2024-02-27 09:30:00',
43                 'user_id' => 1,
44             ],
45             [
46                 'penjualan_id' => 5,
47                 'pembeli' => 'David Wilson',
48                 'penjualan_kode' => 'PJM005',
49                 'penjualan_tanggal' => '2024-02-27 08:00:00',
50                 'user_id' => 2,
51             ],
52             [
53                 'penjualan_id' => 6,
54                 'pembeli' => 'Jessica Lee',
55                 'penjualan_kode' => 'PJM006',
56                 'penjualan_tanggal' => '2024-02-27 08:30:00',
57                 'user_id' => 2,
58             ],
59             [
60                 'penjualan_id' => 7,
61                 'pembeli' => 'Christopher Taylor',
62                 'penjualan_kode' => 'PJM007',
63                 'penjualan_tanggal' => '2024-02-27 08:00:00',
64                 'user_id' => 2,
65             ],
66             [
67                 'penjualan_id' => 8,
68                 'pembeli' => 'Sarah Martinez',
69                 'penjualan_kode' => 'PJM008',
70                 'penjualan_tanggal' => '2024-02-27 08:30:00',
71                 'user_id' => 3,
72             ],
73             [
74                 'penjualan_id' => 9,
75                 'pembeli' => 'Daniel Garcia',
76                 'penjualan_kode' => 'PJM009',
77                 'penjualan_tanggal' => '2024-02-27 08:00:00',
78                 'user_id' => 3,
79             ],
80             [
81                 'penjualan_id' => 10,
82                 'pembeli' => 'Olivia Rodriguez',
83                 'penjualan_kode' => 'PJM010',
84                 'penjualan_tanggal' => '2024-02-27 08:30:00',
85                 'user_id' => 3,
86             ],
87         ];
88         DB::table('t_penjualan')->insert($data);
89     }
90 }
```



```
PS C:\laragon\www\PWL2025\Week3\Jobsheet3> php artisan db:seed --class=PenjualanSeeder
```

**INFO** Seeding database.

	penjualan_id	user_id	pembeli	penjualan_kode	penjualan_tanggal	created_at	updated_at
<input type="checkbox"/>	<a href="#">Edit</a> <a href="#">Copy</a> <a href="#">Delete</a>	1	1 John Doe	PJN001	2024-02-27 08:00:00	NULL	NULL
<input type="checkbox"/>	<a href="#">Edit</a> <a href="#">Copy</a> <a href="#">Delete</a>	2	1 Jane Smith	PJN002	2024-02-27 08:30:00	NULL	NULL
<input type="checkbox"/>	<a href="#">Edit</a> <a href="#">Copy</a> <a href="#">Delete</a>	3	1 Michael Johnson	PJN003	2024-02-27 09:00:00	NULL	NULL
<input type="checkbox"/>	<a href="#">Edit</a> <a href="#">Copy</a> <a href="#">Delete</a>	4	1 Emily Brown	PJN004	2024-02-27 09:30:00	NULL	NULL
<input type="checkbox"/>	<a href="#">Edit</a> <a href="#">Copy</a> <a href="#">Delete</a>	5	2 David Wilson	PJN005	2024-02-27 08:00:00	NULL	NULL
<input type="checkbox"/>	<a href="#">Edit</a> <a href="#">Copy</a> <a href="#">Delete</a>	6	2 Jessica Lee	PJN006	2024-02-27 08:30:00	NULL	NULL
<input type="checkbox"/>	<a href="#">Edit</a> <a href="#">Copy</a> <a href="#">Delete</a>	7	2 Christopher Taylor	PJN007	2024-02-27 08:00:00	NULL	NULL
<input type="checkbox"/>	<a href="#">Edit</a> <a href="#">Copy</a> <a href="#">Delete</a>	8	3 Sarah Martinez	PJN008	2024-02-27 08:30:00	NULL	NULL
<input type="checkbox"/>	<a href="#">Edit</a> <a href="#">Copy</a> <a href="#">Delete</a>	9	3 Daniel Garcia	PJN009	2024-02-27 08:00:00	NULL	NULL
<input type="checkbox"/>	<a href="#">Edit</a> <a href="#">Copy</a> <a href="#">Delete</a>	10	3 Olivia Rodriguez	PJN010	2024-02-27 08:30:00	NULL	NULL

- t\_penjualan\_detail

```
PS C:\laragon\www\PWL2025\Week3\Jobsheet3> php artisan make:seeder Penjualan_DetailSeeder
```

**INFO** Seeder [C:\laragon\www\PWL2025\Week3\Jobsheet3\database\seeders\Penjualan\_DetailSeeder.php] created successfully.



```
database > seeders > Penjualan_DetailSeeder.php > PHP > Penjualan_DetailSeeder
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  /**
10  * Run the database seeds.
11  */
12
13
14  public function run(): void
15  {
16      $data = [];
17
18      // Menentukan jumlah transaksi penjualan yang diinginkan
19      $jumlah_transaksi = 10;
20
21      // Menambahkan setiap transaksi penjualan dengan 3 barang
22      for ($i = 1; $i <= $jumlah_transaksi; $i++) {
23          $barang_ids = range(start: 1, end: 3); // Menyertakan 3 barang dalam setiap transaksi
24          foreach ($barang_ids as $barang_id) {
25              $data[] = [
26                  'detail_id' => count(value: $data) + 1,
27                  'harga' => rand(min: 100000, max: 2000000), // Harga acak untuk setiap barang
28                  'jumlah' => rand(min: 1, max: 5), // Jumlah acak untuk setiap barang
29                  'penjualan_id' => $i,
30                  'barang_id' => $barang_id,
31              ];
32          }
33      }
34
35      DB::table('t_penjualan_detail')->insert($data);
36  }
37 }
```

PS C:\laragon\www\PWL2025\Week3\Jobsheet3> **php artisan db:seed --class=Penjualan\_DetailSeeder**

**INFO** Seeding database.



	<a href="#">Edit</a> <a href="#">Copy</a> <a href="#">Delete</a>	detail_id	penjualan_id	barang_id	harga	jumlah	created_at	updated_at
<input type="checkbox"/>	<a href="#">Edit</a> <a href="#">Copy</a> <a href="#">Delete</a>	1	1	1	11158437	3	NULL	NULL
<input type="checkbox"/>	<a href="#">Edit</a> <a href="#">Copy</a> <a href="#">Delete</a>	2	1	2	19556194	3	NULL	NULL
<input type="checkbox"/>	<a href="#">Edit</a> <a href="#">Copy</a> <a href="#">Delete</a>	3	1	3	8367267	2	NULL	NULL
<input type="checkbox"/>	<a href="#">Edit</a> <a href="#">Copy</a> <a href="#">Delete</a>	4	2	1	14203510	1	NULL	NULL
<input type="checkbox"/>	<a href="#">Edit</a> <a href="#">Copy</a> <a href="#">Delete</a>	5	2	2	12257001	3	NULL	NULL
<input type="checkbox"/>	<a href="#">Edit</a> <a href="#">Copy</a> <a href="#">Delete</a>	6	2	3	598264	4	NULL	NULL
<input type="checkbox"/>	<a href="#">Edit</a> <a href="#">Copy</a> <a href="#">Delete</a>	7	3	1	11414242	1	NULL	NULL
<input type="checkbox"/>	<a href="#">Edit</a> <a href="#">Copy</a> <a href="#">Delete</a>	8	3	2	12356532	2	NULL	NULL
<input type="checkbox"/>	<a href="#">Edit</a> <a href="#">Copy</a> <a href="#">Delete</a>	9	3	3	2258637	1	NULL	NULL
<input type="checkbox"/>	<a href="#">Edit</a> <a href="#">Copy</a> <a href="#">Delete</a>	10	4	1	1877600	1	NULL	NULL
<input type="checkbox"/>	<a href="#">Edit</a> <a href="#">Copy</a> <a href="#">Delete</a>	11	4	2	3864203	4	NULL	NULL
<input type="checkbox"/>	<a href="#">Edit</a> <a href="#">Copy</a> <a href="#">Delete</a>	12	4	3	5520441	3	NULL	NULL
<input type="checkbox"/>	<a href="#">Edit</a> <a href="#">Copy</a> <a href="#">Delete</a>	13	5	1	14475589	5	NULL	NULL
<input type="checkbox"/>	<a href="#">Edit</a> <a href="#">Copy</a> <a href="#">Delete</a>	14	5	2	4062435	5	NULL	NULL
<input type="checkbox"/>	<a href="#">Edit</a> <a href="#">Copy</a> <a href="#">Delete</a>	15	5	3	3724735	2	NULL	NULL
<input type="checkbox"/>	<a href="#">Edit</a> <a href="#">Copy</a> <a href="#">Delete</a>	16	6	1	12861689	2	NULL	NULL
<input type="checkbox"/>	<a href="#">Edit</a> <a href="#">Copy</a> <a href="#">Delete</a>	17	6	2	4336709	1	NULL	NULL
<input type="checkbox"/>	<a href="#">Edit</a> <a href="#">Copy</a> <a href="#">Delete</a>	18	6	3	7955849	3	NULL	NULL
<input type="checkbox"/>	<a href="#">Edit</a> <a href="#">Copy</a> <a href="#">Delete</a>	19	7	1	12292236	4	NULL	NULL
<input type="checkbox"/>	<a href="#">Edit</a> <a href="#">Copy</a> <a href="#">Delete</a>	20	7	2	11841758	4	NULL	NULL
<input type="checkbox"/>	<a href="#">Edit</a> <a href="#">Copy</a> <a href="#">Delete</a>	21	7	3	9889979	3	NULL	NULL
	<a href="#">Console</a>							
<input type="checkbox"/>	<a href="#">Edit</a> <a href="#">Copy</a> <a href="#">Delete</a>	22	8	1	18378663	3	NULL	NULL
<input type="checkbox"/>	<a href="#">Edit</a> <a href="#">Copy</a> <a href="#">Delete</a>	23	8	2	10963015	3	NULL	NULL
<input type="checkbox"/>	<a href="#">Edit</a> <a href="#">Copy</a> <a href="#">Delete</a>	24	8	3	1891065	3	NULL	NULL
<input type="checkbox"/>	<a href="#">Edit</a> <a href="#">Copy</a> <a href="#">Delete</a>	25	9	1	16651036	1	NULL	NULL

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

## D. DB FACADE

DB Façade merupakan fitur dari Laravel yang digunakan untuk melakukan *query* secara langsung dengan mengetikkan perinta SQL secara utuh (*raw query*). Disebut *raw query* (query mentah) karena penulisan query pada DB Façade langsung ditulis sebagaimana yang biasa dituliskan pada database, seperti “`select * from m_user`” atau “`insert into m_user...`” atau “`update m_user set ... Where ...`”

*Raw query* adalah cara paling dasar dan tradisional yang ada di Laravel. Raw query terasa familiar karena biasa kita pakai ketika melakukan query langsung ke database.

### INFO

Dokumentasi penggunaan DB Façade bisa dicek di laman ini

<https://laravel.com/docs/10.x/database#running-queries>



Terdapat banyak method yang bisa digunakan pada DB Façade ini. Akan tetapi yang kita pelajari cukup 4 (empat) method yang umum dipakai, yaitu

a. **DB::select()**

Method ini digunakan untuk mengambil data dari database. Method ini

```
DB::select('select * from m_user'); //Query semua data pada tabel m_user
```

```
DB::select('select * from m_user where level_id = ?', [1]); //Query tabel m_user dengan level_id = 1
```

```
DB::select('select * from m_user where level_id = ? and username = ?', [1, 'admin']);
```

**mengembalikan (return)** data hasil *query*. Contoh

b. **DB::insert()**

Method ini digunakan untuk memasukkan data pada table database. Method ini **tidak memiliki nilai pengembalian (no return)**. Contoh

```
DB::insert('insert into m_level(level_kode, level_nama) values(?,?)', ['cus', 'Pelanggan']);
```

c.

**DB::update()**

Method ini digunakan saat menjalankan *raw query* untuk meng-update data pada database. Method ini **memiliki nilai pengembalian (return)** berupa jumlah baris data yang ter-update. Contoh

```
DB::update('update m_level set level_nama = ? where level_kode = ?', ['Customer', 'cus']);
```

d. **DB::delete()**

Method ini digunakan saat menjalankan *raw query* untuk menghapus data dari table. Method ini **memiliki nilai pengembalian (return)** berupa jumlah baris data yang telah dihapus. Contoh

```
DB::delete('delete from m_level where level_kode = ?', ['cus']);
```



Ok, sekarang mari kita coba praktikkan menggunakan DB Façade pada project kita

#### Praktikum 4 – Implementasi DB Facade

---

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

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

Hasil :

```
PS C:\laragon\www\PWL2025\Week3\Jobsheet3> php artisan make:controller LevelController
[INFO] Controller [C:\laragon\www\PWL2025\Week3\Jobsheet3\app\Http\Controllers\LevelController.php] created successfully.
```

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

```
routes > web.php > ...
1  <?php
2
3  use App\Http\Controllers\LevelController;
4  use Illuminate\Support\Facades\Route;
5
6
7  Route::get('/', function () {
8      return view('welcome');
9  });
10 Route::get('/level', [LevelController::class, 'index']);
```

Hasil :

```
routes > web.php > ...
1  <?php
2
3  use App\Http\Controllers\LevelController;
4  use Illuminate\Support\Facades\Route;
5
6
7  Route::get('/', function (): Factory|View {
8      return view('welcome');
9  });
10 Route::get('/level', [LevelController::class, 'index']);
```

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



### m\_level

```
LevelController.php X web.php
app > Http > Controllers > LevelController.php > ...
1 <?php
2
3 namespace App\Http\Controllers;
4
5 use Illuminate\Http\Request;
6 use Illuminate\Support\Facades\DB;
7
8 class LevelController extends Controller
9 {
10     public function index()
11     {
12         DB::insert('insert into m_level(level_kode, level_nama, created_at) values(?, ?, ?)', ['cus', 'Pelanggan', now()]);
13
14         return 'Insert data baru berhasil';
15     }
16 }
```

### Hasil :

```
app > Http > Controllers > LevelController.php > PHP > LevelController > index()
1 <?php
2
3 namespace App\Http\Controllers;
4
5 use Illuminate\Http\Request;
6 use Illuminate\Support\Facades\DB;
7
8 2 references | 0 implementations
9 class LevelController extends Controller
{
10     1 reference | 0 overrides
11     public function index(): string
12     {
13         DB::insert('insert into m_level(level_kode, level_nama, created_at) values(?, ?, ?)', ['CUS2', 'Pelanggan', now()]);
14         return 'Insert data baru berhasil';
15     }
16 }
```

4. Kita coba jalankan di browser dengan url [localhost/PWL\\_POS/public/level](http://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
1	ADM	Administrator	NULL	NULL
2	MNG	Manager	NULL	NULL
3	STF	Staff/Kasir	NULL	NULL
4	CUS	Pelanggan	2024-02-26 08:20:00	NULL

### Hasil :

The screenshot shows a browser window with the URL [pwl2025.test/Week3/Jobsheet3/public/level](http://pwl2025.test/Week3/Jobsheet3/public/level). The page displays the message "Insert data baru berhasil" (Data inserted successfully).



	<a href="#">Edit</a>	<a href="#">Copy</a>	<a href="#">Delete</a>	level_id	level_kode	level_nama	created_at	updated_at
<input type="checkbox"/>	<a href="#">Edit</a>	<a href="#">Copy</a>	<a href="#">Delete</a>	1	ADM	Administrator	NULL	NULL
<input type="checkbox"/>	<a href="#">Edit</a>	<a href="#">Copy</a>	<a href="#">Delete</a>	2	MNG	Manager	NULL	NULL
<input type="checkbox"/>	<a href="#">Edit</a>	<a href="#">Copy</a>	<a href="#">Delete</a>	3	STF	Staff/Kasir	NULL	NULL
<input type="checkbox"/>	<a href="#">Edit</a>	<a href="#">Copy</a>	<a href="#">Delete</a>	4	CUS2	Pelanggan	2025-03-05 15:25:22	NULL

5. Selanjutnya, kita modifikasi lagi file [LevelController](#) untuk meng-update data di table [m\\_level](#) seperti berikut

```
LevelController.php x web.php
app > Http > Controllers > LevelController.php > ...
1  <?php
2
3  namespace App\Http\Controllers;
4
5  use Illuminate\Http\Request;
6  use Illuminate\Support\Facades\DB;
7
8  class LevelController extends Controller
9  {
10     public function index()
11     {
12         //DB::insert('insert into m_level(level_kode, level_nama, created_at) values(?, ?, ?)', ['cus', 'Pelanggan', now()]);
13         // return 'Insert data baru berhasil';
14
15         $row = DB::update('update m_level set level_nama = ? where level_kode = ?', ['Customer', 'CUS']);
16         return 'Update data berhasil. Jumlah data yang diupdate: ' . $row.' baris';
17     }
18 }
```

**Hasil :**

```
app > Http > Controllers > LevelController.php > PHP Intelephense > LevelController > Index
1  <?php
2
3  namespace App\Http\Controllers;
4
5  use Illuminate\Http\Request;
6  use Illuminate\Support\Facades\DB;
7
8  class LevelController extends Controller
9  {
10     public function index(): string
11     {
12         //DB::insert('insert into m_level(level_kode, level_nama, created_at) values(?, ?, ?)', ['cus2', 'Pelanggan', now()]);
13         //return 'Insert data baru berhasil';
14
15         $row = DB::update('update m_level set level_nama = ? where level_kode = ?', ['Customer', 'CUS2']);
16         return 'Update data berhasil. Jumlah data yang diupdate: ' . $row.' baris';
17     }
18 }
```

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](#)

**Hasil :**



← ⏪ Not secure | [pwl2025.test/Week3/Jobsheet3/public/level](http://pwl2025.test/Week3/Jobsheet3/public/level)

Update data berhasil. Jumlah data yang diupdate: 1 baris

level_id	level_kode	level_nama	created_at	updated_at
1	ADM	Administrator	NULL	NULL
2	MNG	Manager	NULL	NULL
3	STF	Staff/Kasir	NULL	NULL
4	CUS2	Customer	2025-03-05 15:25:22	NULL

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

```
LevelController.php X web.php
app > Http > Controllers > LevelController.php > LevelController > index
1 <?php
2
3 namespace App\Http\Controllers;
4
5 use Illuminate\Http\Request;
6 use Illuminate\Support\Facades\DB;
7
8 class LevelController extends Controller
9 {
10     public function index()
11     {
12         // DB::insert('insert into m_level(level_kode, level_nama, created_at) values(?, ?, ?)', ['cus', 'Pelanggan', now()]);
13         // return 'Insert data baru berhasil';
14
15         // $row = DB::update('update m_level set level_nama = ? where level_kode = ?', ['Customer', 'CUS']);
16         // return 'Update data berhasil. Jumlah data yang diupdate: ' . $row.' baris';
17
18         $row = DB::delete('delete from m_level where level_kode = ?', ['cus']);
19         return 'Delete data berhasil. Jumlah data yang dihapus: ' . $row.' baris';
20     }
21 }
```



### Hasil :

```
app > Http > Controllers > LevelController.php > PHP Intelephense > LevelController > index > $row
 1  <?php
 2
 3  namespace App\Http\Controllers;
 4
 5  use Illuminate\Http\Request;
 6  use Illuminate\Support\Facades\DB;
 7
 8  2 references | 0 implementations
 9  class LevelController extends Controller
10 {
11     1 reference | 0 overrides
12     public function index(): string
13     {
14         //DB::insert('insert into m_level(level_kode, level_nama, created_at) values(?, ?, ?)', ['CUS2', 'Pelanggan', now()]);
15         //return 'Insert data baru berhasil';
16
17         //$row = DB::update('update m_level set level_nama = ? where level_kode = ?', ['Customer', 'CUS2']);
18         //return 'Update data berhasil. Jumlah data yang diupdate: ' . $row.' baris';
19
20         $row = DB::delete('delete from m_level where level_kode = ?', ['CUS']);
21         return 'Delete data berhasil. Jumlah data yang dihapus: ' . $row. ' baris';
22     }
}
```

← ⏪ Not secure | [pwl2025.test/Week3/Jobsheet3/public/level](http://pwl2025.test/Week3/Jobsheet3/public/level)

Delete data berhasil. Jumlah data yang dihapus: 0 baris

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

```
3  namespace App\Http\Controllers;
4
5  use Illuminate\Http\Request;
6  use Illuminate\Support\Facades\DB;
7
8  class LevelController extends Controller
9  {
10     public function index()
11     {
12         // DB::insert('insert into m_level(level_kode, level_nama, created_at) values(?, ?, ?)', ['CUS', 'Pelanggan', now()]);
13         // return 'Insert data baru berhasil';
14
15         //$row = DB::update('update m_level set level_nama = ? where level_kode = ?', ['Customer', 'CUS']);
16         // return 'Update data berhasil. Jumlah data yang diupdate: ' . $row.' baris';
17
18         //$row = DB::delete('delete from m_level where level_kode = ?', ['CUS']);
19         // return 'Delete data berhasil. Jumlah data yang dihapus: ' . $row. ' baris';
20
21         $data = DB::select('select * from m_level');
22         return view('level', ['data' => $data]);
23     }
24 }
```



```
app > Http > Controllers > LevelController.php > PHP > LevelController
  1  <?php
  2
  3  namespace App\Http\Controllers;
  4
  5  use Illuminate\Http\Request;
  6  use Illuminate\Support\Facades\DB;
  7
  8  2 references | 0 implementations
  9  class LevelController extends Controller
 [ 10  1 reference | 0 overrides
 11  public function index(): Factory|View
 12  {
 13      //DB::insert('insert into m_level(level_kode, level_nama, created_at) values(?, ?, ?)', ['CUS2', 'Pelanggan', now()]);
 14      //return 'Insert data baru berhasil';
 15
 16      //$row = DB::update('update m_level set level_nama = ? where level_kode = ?', ['Customer', 'CUS2']);
 17      //return 'Update data berhasil. Jumlah data yang diupdate: ' . $row.' baris';
 18
 19      //$row = DB::delete('delete from m_level where level_kode = ?', ['CUS']);
 20      //return 'Delete data berhasil. Jumlah data yang dihapus: '.$row.' baris';
 21
 22      $data = DB::select('select * from m_level');
 23      return view('level', data: ['data' => $data]);
 24  }
```

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

```
LevelController.php  level.blade.php  web.php
resources > views > level.blade.php > ...
  1  <!DOCTYPE html>
  2  <html>
  3      <head>
  4          <title>Data Level Pengguna</title>
  5      </head>
  6      <body>
  7          <h1>Data Level Pengguna</h1>
  8          <table border="1" cellpadding="2" cellspacing="0">
  9              <tr>
 10                  <th>ID</th>
 11                  <th>Kode Level</th>
 12                  <th>Nama Level</th>
 13              </tr>
 14              @foreach ($data as $d)
 15              <tr>
 16                  <td>{{ $d->level_id }}</td>
 17                  <td>{{ $d->level_kode }}</td>
 18                  <td>{{ $d->level_nama }}</td>
 19              </tr>
 20              @endforeach
 21          </table>
 22      </body>
 23  </html>
```

**Hasil :**



```
resources > views > 📄 level.blade.php > ⚙️ html
 1  <!DOCTYPE html>
 2  <html lang="en">
 3  <head>
 4  |   <title>Data Level Pengguna</title>
 5  </head>
 6  <body>
 7  |   <h1>Data Level Pengguna</h1>
 8  |   <table border="1" cellpadding="2" cellspacing="0">
 9  |       <tr>
10  |           <th>ID</th>
11  |           <th>Kode Level</th>
12  |           <th>Nama Level</th>
13  |       </tr>
14  |       @foreach ($data as $d)
15  |           <tr>
16  |               <td>{{$d->level_id}}</td>
17  |               <td>{{$d->level_kode}}</td>
18  |               <td>{{$d->level_nama}}</td>
19  |           </tr>
20  |       @endforeach
21  |   </table>
22  </body>
23  </html>
```

10. Silahkan dicoba pada browser dan amati apa yang terjadi

```
app > Http > Controllers > 📄 LevelController.php > PHP > 🛡️ LevelController
 1  <?php
 2
 3  namespace App\Http\Controllers;
 4
 5  use Illuminate\Http\Request;
 6  use Illuminate\Support\Facades\DB;
 7
 8  2 references | 0 implementations
 9  class LevelController extends Controller
10  {
11      1 reference | 0 overrides
12      public function index(): Factory|View
13      {
14          //DB::insert('insert into m_level(level_kode, level_nama, created_at) values(?, ?, ?)', ['CUS2', 'Pelanggan', now()]);
15          //return 'Insert data baru berhasil';
16
17          //$row = DB::update('update m_level set level_nama = ? where level_kode = ?', ['Customer', 'CUS2']);
18          //return 'Update data berhasil. Jumlah data yang diupdate: ' . $row.' baris';
19
20          //$row = DB::delete('delete from m_level where level_kode = ?', ['CUS']);
21          //return 'Delete data berhasil. Jumlah data yang dihapus: '.$row.' baris';
22
23          $data = DB::select('select * from m_level');
24          return view('view: level', data: ['data' => $data]);
25      }
26  }
```



```
resources > views > 📄 level.blade.php > ⚙️ html
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4  |   <title>Data Level Pengguna</title>
5  </head>
6  <body>
7  |   <h1>Data Level Pengguna</h1>
8  |   <table border="1" cellpadding="2" cellspacing="0">
9  |       <tr>
10 |           <th>ID</th>
11 |           <th>Kode Level</th>
12 |           <th>Nama Level</th>
13 |       </tr>
14 |       @foreach ($data as $d)
15 |           <tr>
16 |               <td>{{$d->level_id}}</td>
17 |               <td>{{$d->level_kode}}</td>
18 |               <td>{{$d->level_nama}}</td>
19 |           </tr>
20 |       @endforeach
21 |   </table>
22 | </body>
23 | </html>
```

```
routes > 📄 web.php > ...
1  <?php
2  |
3  |use App\Http\Controllers\LevelController;
4  |use Illuminate\Support\Facades\Route;
5  |
6  |
7  Route::get('/', function (): Factory|View {
8  |    return view('welcome');
9  });
10 |
11 Route::get('/level', [LevelController::class, 'index']);
```



Not secure

pwl2025.test/Week3/Jobsheet3/public/level

## Data Level Pengguna

ID	Kode Level	Nama Level
1	ADM	Administrator
2	MNG	Manager
3	STF	Staff/Kasir
4	CUS2	Customer

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

### E. QUERY BUILDER

*Query builder* adalah fitur yang disediakan Laravel untuk melakukan proses CRUD (*create, retrieve/read, update, delete*) pada database. Berbeda dengan *raw query* pada DB Facede yang mengharuskan kita menulis perintah SQL, pada *query builder* perintah SQL ini diakses menggunakan method. Jadi, kita tidak menulis perintah SQL secara langsung, melainkan cukup memanggil method-method yang ada di *query builder*.

*Query builder* membuat kode kita menjadi rapi dan lebih mudah dibaca. Selain itu *query builder* tidak terikat ke satu jenis database, jadi *query builder* bisa digunakan untuk mengakses berbagai jenis database seperti MySQL, MariaDB, PostgreSQL, SQL Server, dll. Jika suatu saat ingin beralih dari database MySQL ke PostgreSQL, tidak akan banyak kendala. Namun kelemahan dari *query builder* adalah kita harus mengetahui method-method apa saja yang ada di *query builder*.

#### INFO

Dokumentasi penggunaan Query Builder pada Laravel bisa dicek di laman ini

<https://laravel.com/docs/10.x/queries>

Ciri khas *query builder* Laravel adalah kita tentukan dahulu target table yang akan kita akses untuk operasi CRUD.

```
DB::table('<nama-tabel>'); // query builder untuk melakukan operasi CRUD pada tabel yang dituju
```



Perintah pertama yang dilakukan pada query builder adalah menentukan nama table yang akan dilakukan operasi CRUD. Kemudian baru disusul method yang ingin digunakan sesuai dengan peruntukannya. Contoh

- a. Perintah untuk *insert* data dengan method `insert()`

```
DB::table('m_kategori')->insert(['kategori_kode' => 'SMP', 'kategori_nama' => 'Smartphone']);
```

Query yang dihasilkan dari kode di atas adalah

```
insert into m_kategori(kategori_kode, kategori_nama) values('SMP', 'Smartphone');
```

- b. Perintah untuk *update* data dengan method `where()` dan `update()`

```
DB::table('m_kategori')->where('kategori_id', 1)->update(['kategori_nama' => 'Makanan Ringan']);
```

Query yang dihasilkan dari kode di atas adalah

```
update m_kategori set kategori_nama = 'Makanan Ringan' where kategori_id = 1;
```

- c. Perintah untuk *delete* data dengan method `where()` dan `delete()`

```
DB::table('m_kategori')->where('kategori_id', 9) ->delete();
```

Query yang dihasilkan dari kode di atas adalah

```
delete from m_kategori where kategori_id = 9;
```

- d. Perintah untuk ambil data

<i>Method Query Builder</i>	<i>Query yang dihasilkan</i>
<code>DB::table('m_kategori')-&gt;get();</code>	<code>select * from m_kategori</code>
<code>DB::table('m_kategori')-&gt;where('kategori_id', 1)-&gt;get();</code>	<code>select * from m_kategori where kategori_id = 1;</code>
<code>DB::table('m_kategori')-&gt;select('kategori_kode')-&gt;where('kategori_id', 1)-&gt;get();</code>	<code>select kategori_kode from m_kategori where kategori_id = 1;</code>

## Praktikum 5 – Implementasi *Query Builder*

---

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



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

## Hasil

```
PS C:\laragon\www\PWL2025\Week3\Jobsheet3> php artisan make:controller KategoriController
```

```
[INFO] Controller [C:\laragon\www\PWL2025\Week3\Jobsheet3\app\Http\Controllers\KategoriController.php] created successfully.
```

2. Kita modifikasi dulu untuk routing-nya, ada di [PWL\\_POS/routes/web.php](#)

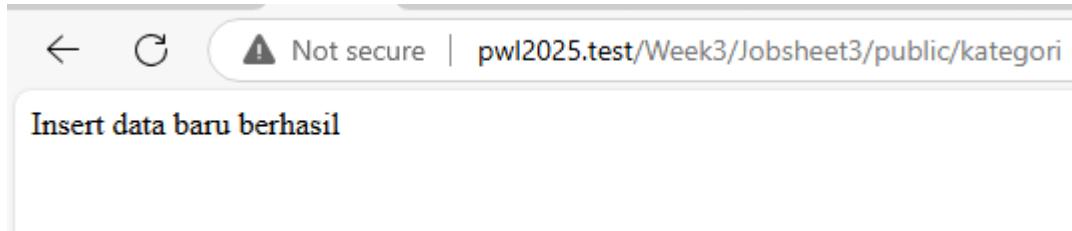
```
routes > web.php > ...
1 <?php
2
3 use App\Http\Controllers\KategoriController;
4 use App\Http\Controllers\LevelController;
5 use Illuminate\Support\Facades\Route;
6
7
8 Route::get('/', function () {
9     return view('welcome');
10 });
11
12 Route::get('/level', [LevelController::class, 'index']);
13 Route::get('/kategori', [KategoriController::class, 'index']);
```

3. Selanjutnya, kita modifikasi file [KategoriController](#) untuk menambahkan 1 data ke table [m\\_kategori](#)

```
app > Http > Controllers > KategoriController.php > KategoriController > index
1 <?php
2
3 namespace App\Http\Controllers;
4
5 use Illuminate\Http\Request;
6 use Illuminate\Support\Facades\DB;
7
8 class KategoriController extends Controller
9 {
10     public function index()
11     {
12         $data = [
13             'kategori_kode' => 'SNK',
14             'kategori_nama' => 'Snack/Makanan Ringan',
15             'created_at' => now()
16         ];
17         DB::table('m_kategori')->insert($data);
18         return 'Insert data baru berhasil';
19     }
20 }
```

4. Kita coba jalankan di browser dengan url [localhost/PWL\\_POS/public/kategori](http://localhost/PWL_POS/public/kategori) dan amati apa yang terjadi pada table [m\\_kategori](#) di database, *screenshot* perubahan yang ada pada table [m\\_kategori](#)

**Hasil :**



← →	kategori_id	kategori_kode	kategori_nama	created_at	updated_at
<input type="checkbox"/> <a href="#">Edit</a> <a href="#">Copy</a> <a href="#">Delete</a>	1	KAT001	Makanan	NULL	NULL
<input type="checkbox"/> <a href="#">Edit</a> <a href="#">Copy</a> <a href="#">Delete</a>	2	KAT002	MakeUp	NULL	NULL
<input type="checkbox"/> <a href="#">Edit</a> <a href="#">Copy</a> <a href="#">Delete</a>	3	KAT003	Aksesoris	NULL	NULL
<input type="checkbox"/> <a href="#">Edit</a> <a href="#">Copy</a> <a href="#">Delete</a>	4	KAT004	Pewangi	NULL	NULL
<input type="checkbox"/> <a href="#">Edit</a> <a href="#">Copy</a> <a href="#">Delete</a>	5	KAT005	Alat Mandi	NULL	NULL
<input type="checkbox"/> <a href="#">Edit</a> <a href="#">Copy</a> <a href="#">Delete</a>	6	SNK	Snack/Makanan Ringan	2025-03-05 15:57:41	NULL

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

```
app > Http > Controllers >  KategoriController.php >  KategoriController >  index
1  <?php
2
3  namespace App\Http\Controllers;
4
5  use Illuminate\Http\Request;
6  use Illuminate\Support\Facades\DB;
7
8  class KategoriController extends Controller
9  {
10     public function index()
11     {
12         /* $data = [
13             'kategori.kode' => 'SNK',
14             'kategori.nama' => 'Snack/Makanan Ringan',
15             'created_at' => now()
16         ];
17         DB::table('m_kategori')->insert($data);
18         return 'Insert data baru berhasil';
19
20         $row = DB::table('m_kategori')->where('kategori.kode', 'SNK')->update(['kategori.nama' => 'Camilan']);
21         return 'Update data berhasil. Jumlah data yang diupdate: ' . $row.' baris';
22     }
23 }
```

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`

**Hasil :**



Not secure | pwl2025.test/Week3/Jobsheet3/public/kategori

Update data berhasil. Jumlah data yang diupdate: 1 baris

	kategori_id	kategori_kode	kategori_nama	created_at	updated_at
<input type="checkbox"/>	1	KAT001	Makanan	NULL	NULL
<input type="checkbox"/>	2	KAT002	MakeUp	NULL	NULL
<input type="checkbox"/>	3	KAT003	Aksesoris	NULL	NULL
<input type="checkbox"/>	4	KAT004	Pewangi	NULL	NULL
<input type="checkbox"/>	5	KAT005	Alat Mandi	NULL	NULL
<input type="checkbox"/>	6	SNK	Camilan	2025-03-05 15:57:41	NULL

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

```
10 public function index()
11 {
12     /* $data = [
13         'kategori.kode' => 'SNK',
14         'kategori.nama' => 'Snack/Makanan Ringan',
15         'created_at' => now()
16     ];
17     DB::table('m_kategori')->insert($data);
18     return 'Insert data baru berhasil';
19
20     // $row = DB::table('m_kategori')->where('kategori.kode', 'SNK')->update(['kategori.nama' => 'Camilan']);
21     // return 'Update data berhasil. Jumlah data yang diupdate: ' . $row.' baris';
22
23     $row = DB::table('m_kategori')->where('kategori.kode', 'SNK')->delete();
24     return 'Delete data berhasil. Jumlah data yang dihapus: ' . $row.' baris';
25 }
```

Hasil :

Not secure | pwl2025.test/Week3/Jobsheet3/public/kateg...

Delete data berhasil. Jumlah data yang diupdate: 1 baris



	kategori_id	kategori_kode	kategori_nama	created_at	updated_at
<input type="checkbox"/>	1	KAT001	Makanan	NULL	NULL
<input type="checkbox"/>	2	KAT002	MakeUp	NULL	NULL
<input type="checkbox"/>	3	KAT003	Aksesoris	NULL	NULL
<input type="checkbox"/>	4	KAT004	Pewangi	NULL	NULL
<input type="checkbox"/>	5	KAT005	Alat Mandi	NULL	NULL

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

```
10    public function index()
11    {
12        /* $data = [
13            'kategori.kode' => 'SNK',
14            'kategori.nama' => 'Snack/Makanan Ringan',
15            'created_at' => now()
16        ];
17        DB::table('m_kategori')->insert($data);
18        return 'Insert data baru berhasil';
19
20        // $row = DB::table('m_kategori')->where('kategori.kode', 'SNK')->update(['kategori.nama' => 'Camilan']);
21        // return 'Update data berhasil. Jumlah data yang diupdate: ' . $row.' baris';
22
23        // $row = DB::table('m_kategori')->where('kategori.kode', 'SNK')->delete();
24        // return 'Delete data berhasil. Jumlah data yang dihapus: ' . $row.' baris';
25
26        $data = DB::table('m_kategori')->get();
27        return view('kategori', ['data' => $data]);
28    }
```

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`

```
resources > views >  kategori.blade.php >  html >  body >  table >  tr >  td
1  <!DOCTYPE html>
2  <html>
3      <head>
4          <title>Data Kategori Barang</title>
5      </head>
6      <body>
7          <h1>Data Kategori Barang</h1>
8          <table border="1" cellpadding="2" cellspacing="0">
9              <tr>
10                 <th>ID</th>
11                 <th>Kode Kategori</th>
12                 <th>Nama Kategori</th>
13             </tr>
14             @foreach ($data as $d)
15                 <tr>
16                     <td>{{ $d->kategori_id }}</td>
17                     <td>{{ $d->kategori_kode }}</td>
18                     <td>{{ $d->kategori_nama }}</td>
19                 </tr>
20             @endforeach
21         </table>
22     </body>
23 </html>
```

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



**Hasil :**



## Data kategori Barang

ID	Kode Kategori	Nama Kategori
1	KAT001	Makanan
2	KAT002	MakeUp
3	KAT003	Aksesoris
4	KAT004	Pewangi
5	KAT005	Alat Mandi

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

### F. ELOQUENT ORM

Eloquent ORM adalah fitur bawaan dari laravel. Eloquent ORM adalah cara pengaksesan database dimana setiap baris tabel dianggap sebagai sebuah object. Kata ORM sendiri merupakan singkatan dari *Object-relational mapping*, yakni suatu teknik programming untuk mengkonversi data ke dalam bentuk object.

#### INFO

Eloquent ORM memerlukan Model untuk proses konversi data pada tabel menjadi object. Object inilah yang nantinya akan kita akses dari dalam controller. Oleh karena itu **membuat Model pada Laravel berarti menggunakan Eloquent ORM**. Silahkan cek disini

<https://laravel.com/docs/10.x/eloquent>

Perintah untuk membuat model adalah sebagai berikut

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

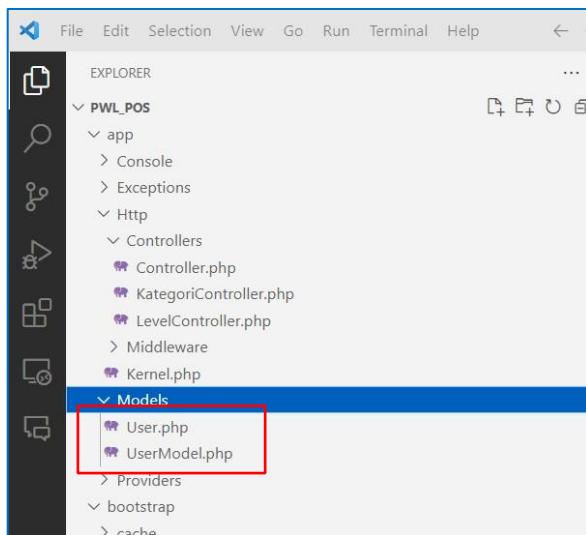
Untuk bisa melakukan operasi **CRUD** (*create, read/retrieve, update, delete*), kita harus membuat sebuah model sesuai dengan target tabel yang ingin digunakan. Jadi, **dalam 1 model, merepresentasikan 1 tabel database.**



## 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

```
app > Models > UserModel.php > UserModel
1  <?php
2
3  namespace App\Models;
4
5  use Illuminate\Database\Eloquent\Factories\HasFactory;
6  use Illuminate\Database\Eloquent\Model;
7
8  class UserModel extends Model
9  {
10     use HasFactory;
11
12     protected $table = 'm_user';           // Mendefinisikan nama tabel yang digunakan oleh model ini
13     protected $primaryKey = 'user_id';    // Mendefinisikan primary key dari tabel yang digunakan
14 }
15
```

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



```
routes > 🌐 web.php > ...
1   <?php
2
3   use App\Http\Controllers\KategoriController;
4   use App\Http\Controllers\LevelController;
5   use App\Http\Controllers\UserController;
6   use Illuminate\Support\Facades\Route;
7
8
9   Route::get('/', function () {
10     return view('welcome');
11 });
12
13 Route::get('/level', [LevelController::class, 'index']);
14 Route::get('/kategori', [KategoriController::class, 'index']);
15 Route::get('/user', [UserController::class, 'index']);
16
```

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

```
app > Http > Controllers > 🌐 UserController.php > ...
1   <?php
2
3   namespace App\Http\Controllers;
4
5   use App\Models\UserModel;
6   use Illuminate\Http\Request;
7
8   class UserController extends Controller
9   {
10     public function index()
11     {
12       // coba akses model UserModel
13       $user = UserModel::all(); // ambil semua data dari tabel m_user
14       return view('user', ['data' => $user]);
15     }
16 }
```

6. Kemudian kita buat view **user.blade.php**

```
resources > views > 🌐 user.blade.php > ...
1   <!DOCTYPE html>
2   <html>
3     <head>
4       <title>Data User</title>
5     </head>
6     <body>
7       <h1>Data User</h1>
8       <table border="1" cellpadding="2" cellspacing="0">
9         <tr>
10           <th>ID</th>
11           <th>Username</th>
12           <th>Nama</th>
13           <th>ID Level Pengguna</th>
14         </tr>
15         @foreach ($data as $d)
16         <tr>
17           <td>{{ $d->user_id }}</td>
18           <td>{{ $d->username }}</td>
19           <td>{{ $d->nama }}</td>
20           <td>{{ $d->level_id }}</td>
21         </tr>
22       @endforeach
23     </table>
24   </body>
25 </html>
```

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



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

8. Setelah itu, kita modifikasi lagi file **UserController**

```
app > Http > Controllers > UserController.php > ...
1   <?php
2
3   namespace App\Http\Controllers;
4
5   use App\Models\UserModel;
6   use Illuminate\Http\Request;
7   use Illuminate\Support\Facades\Hash;
8
9   class UserController extends Controller
10  {
11      public function index()
12      {
13          // tambah data user dengan Eloquent Model
14          $data = [
15              'username' => 'customer-1',
16              'nama' => 'Pelanggan',
17              'password' => Hash::make('12345'),
18              'level_id' => 4
19          ];
20          UserModel::insert($data); // tambahkan data ke tabel m_user
21
22          // coba akses model UserModel
23          $user = UserModel::all(); // ambil semua data dari tabel m_user
24          return view('user', ['data' => $user]);
25      }
26  }
```

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
4	customer-1	Manager 1	4

10. Kita modifikasi lagi file [UserController](#) menjadi seperti berikut

```
9  class UserController extends Controller
10 {
11     public function index()
12     {
13         // tambah data user dengan Eloquent Model
14         $data = [
15             'nama' => 'Pelanggan Pertama',
16         ];
17         UserModel::where('username', 'customer-1')->update($data); // update data user
18
19         // coba akses model UserModel
20         $user = UserModel::all(); // ambil semua data dari tabel m_user
21         return view('user', ['data' => $user]);
22     }
23 }
```

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

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



## G. Penutup

Jawablah pertanyaan berikut sesuai pemahaman materi di atas

1. Pada **Praktikum 1 - Tahap 5**, apakah fungsi dari `APP_KEY` pada *file setting .env* Laravel?

**Jawab :** APP\_KEY digunakan untuk mengenkripsi data sensitif dalam aplikasi Laravel, seperti sesi dan token

2. Pada **Praktikum 1**, bagaimana kita men-*generate* nilai untuk `APP_KEY`?

**Jawab :** 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?

**Jawab:**

- `Create_users_table` , digunakan untuk membuat tabel users, yang menyimpan informasi pengguna aplikasi, seperti nama, email, password, dan timestamp
- `Create_password_reset_tokens_table`
- `Create_failed_jobs_table`, digunakan untuk membuat tabel failed\_jobs, yang menyimpan informasi tentang job yang gagal diproses dalam sistem queue Laravel
- `Create_personal_access_tokens_table`, digunakan untuk membuat tabel personal\_access\_tokens, yang menyimpan token akses pribadi untuk autentikasi API.

4. Secara *default*, file migrasi terdapat kode `$table->timestamps();`, apa tujuan/*output* dari fungsi tersebut?

**Jawab :** untuk mencatat waktu pembuatan dan pembaruan data

5. Pada File Migrasi, terdapat fungsi `$table->id();` Tipe data apa yang dihasilkan dari fungsi tersebut?

**Jawab :** Menghasilkan kolom id bertipe BIGINT unsigned dengan auto increment

6. Apa bedanya hasil migrasi pada table `m_level`, antara menggunakan `$table->id();` dengan menggunakan `$table->id('level_id');` ?

**Jawab :**

- `$table->id();` → Membuat kolom id sebagai primary key default.
- `$table->id('level_id');` → Mengubah nama primary key menjadi level\_id.

7. Pada migration, Fungsi `->unique()` digunakan untuk apa?

**Jawab :** Digunakan untuk memastikan nilai dalam kolom tidak boleh duplikat.



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')` ?

**Jawab:**

- Pada tabel `m_level`, `$table->id('level_id')` digunakan untuk mendefinisikan primary key.
- Pada tabel `m_user`, `$table->unsignedBigInteger('level_id')` digunakan untuk mendefinisikan foreign key yang mengacu pada `level_id` di tabel `m_level`.

9. Pada **Praktikum 3 - Tahap 6**, apa tujuan dari Class `Hash`? dan apa maksud dari kode program `Hash::make('1234');`?

**Jawab :** Class Hash di Laravel digunakan untuk hashing data sensitif, seperti password, agar lebih aman. Fungsi `Hash::make('1234')` mengenkripsi ``1234`` dengan bcrypt untuk mencegah pencurian data.

10. Pada **Praktikum 4 - Tahap 3/5/7**, pada *query builder* terdapat tanda tanya (?), apa kegunaan dari tanda tanya (?) tersebut?

**Jawab:** Pada query builder di Laravel, tanda tanya (?) berfungsi sebagai placeholder untuk menghindari risiko SQL injection. Ketika tanda tanya digunakan dalam query, itu akan digantikan dengan nilai yang aman oleh query builder secara otomatis saat eksekusi. Ini membantu untuk binding parameter dengan aman, memastikan bahwa nilai yang dimasukkan ke dalam query akan disanitasi dan tidak dapat dimanipulasi untuk merusak query atau melakukan serangan berbahaya.

11. Pada **Praktikum 6 - Tahap 3**, apa tujuan penulisan kode `protected $table = 'm_user';` dan `protected $primaryKey = 'user_id';` ?

**Jawab :** Penulisan `protected $table = 'm_user';` dalam model Laravel digunakan untuk menetapkan nama tabel database yang berbeda dari default yang digunakan oleh Laravel, yaitu nama tabel yang setara dengan nama model dalam bentuk plural. Dengan menggunakan penulisan ini, kita dapat menyesuaikan nama tabel di database yang mungkin tidak mengikuti konvensi tersebut. Sedangkan `protected $primaryKey = 'user_id';` digunakan untuk menentukan kolom primary key yang berbeda dari default (`id`). Ini memungkinkan model untuk berfungsi dengan tabel dan kolom primary key yang tidak mengikuti konvensi default Laravel



12. Menurut kalian, lebih mudah menggunakan mana dalam melakukan operasi CRUD ke database (*DB Façade / Query Builder / Eloquent ORM*) ? jelaskan

**Jawab :** Menurut saya, Query Builder lebih mudah digunakan untuk operasi CRUD ke database karena Query Builder memungkinkan buat saya menulis query yang lebih ringkas dan jelas tanpa harus menulis query SQL panjang seperti yang diperlukan dengan DB Façade. Selain itu, struktur kode yang lebih rapi dan sintaksis yang mudah dipahami menjadikan Query Builder sehingga lebih nyaman bagi saya dalam membuat dan memahami query yang ingin dijalankan.

\*\*\* *Sekian, dan selamat belajar* \*\*\*