



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**.

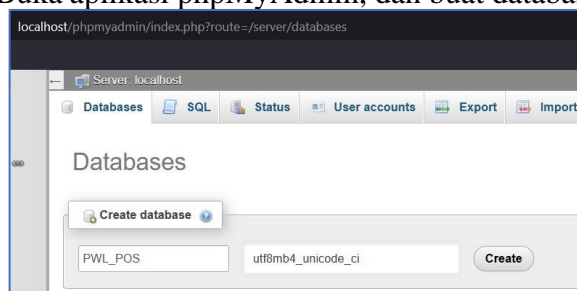
Project PWL_POS akan kita gunakan sampai pertemuan 12 nanti, sebagai project yang akan kita pelajari

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.

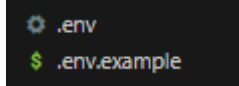
Praktikum 1 - pengaturan database:

1. Buka aplikasi phpMyAdmin, dan buat database baru dengan nama **PWL_POS**

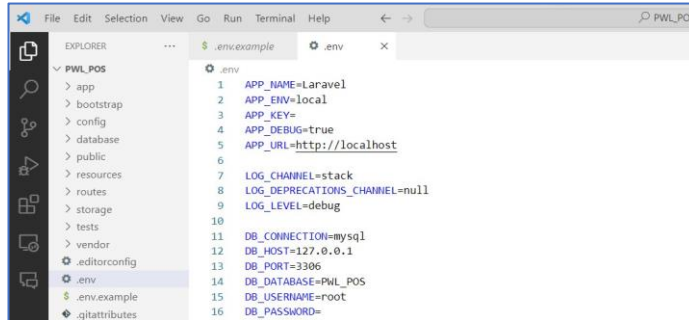




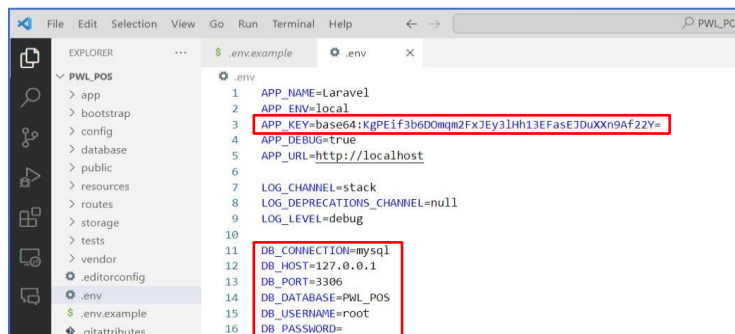
2. Buka aplikasi VSCode dan buka folder project **PWL_POS** yang sudah kita buat
3. Copy file **.env.example** menjadi **.env**



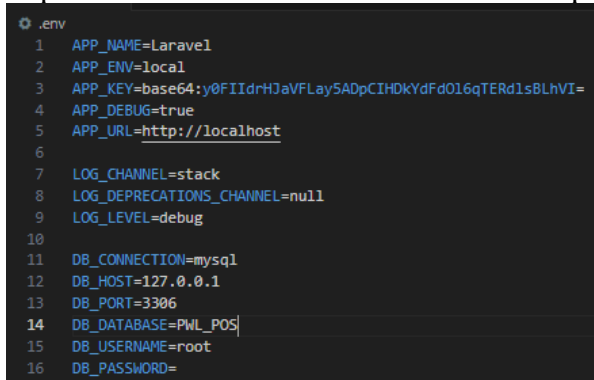
4. Buka file **.env**, dan pastikan konfigurasi **APP_KEY** bernilai. Jika belum bernilai silahkan kalian *generate* menggunakan **php artisan**.



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



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





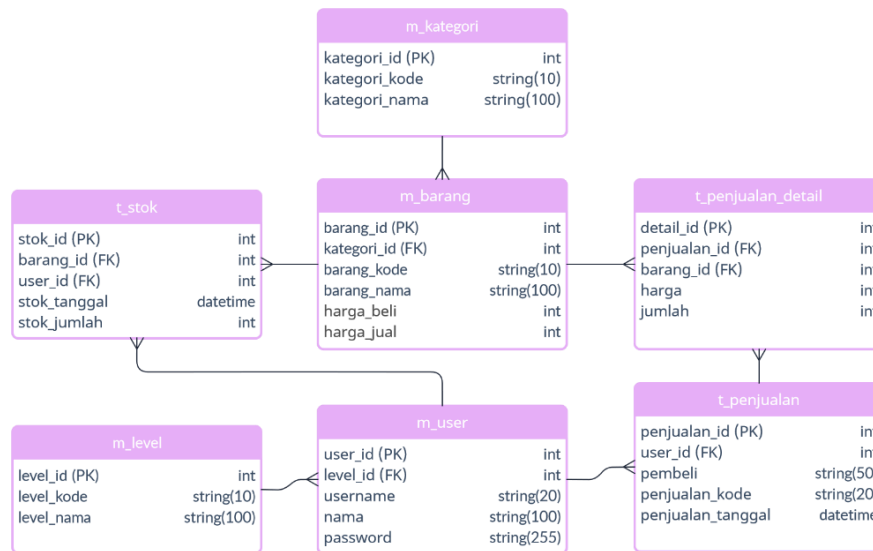
B. MIGRATION

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

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

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 dilanjutkan ke file migrasi dengan table yang memiliki relasi yang banyak.

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



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

INFO

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

Pembuatan file migrasi bisa menggunakan 2 cara, yaitu

- Menggunakan [artisan](#) untuk membuat *file migration*

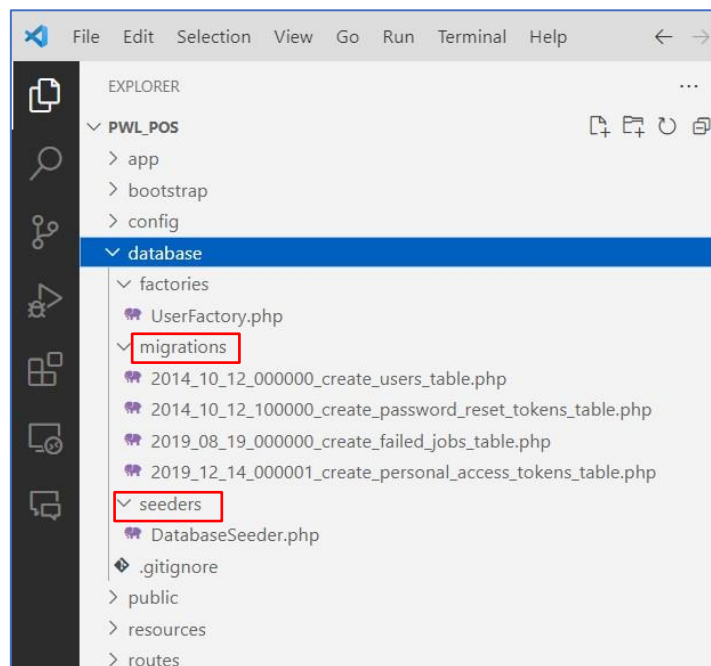
```
php artisan make:migration <nama-file-tabel> --create=<nama-tabel>
```

- Menggunakan [artisan](#) untuk membuat *file model* + *file migration*

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

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

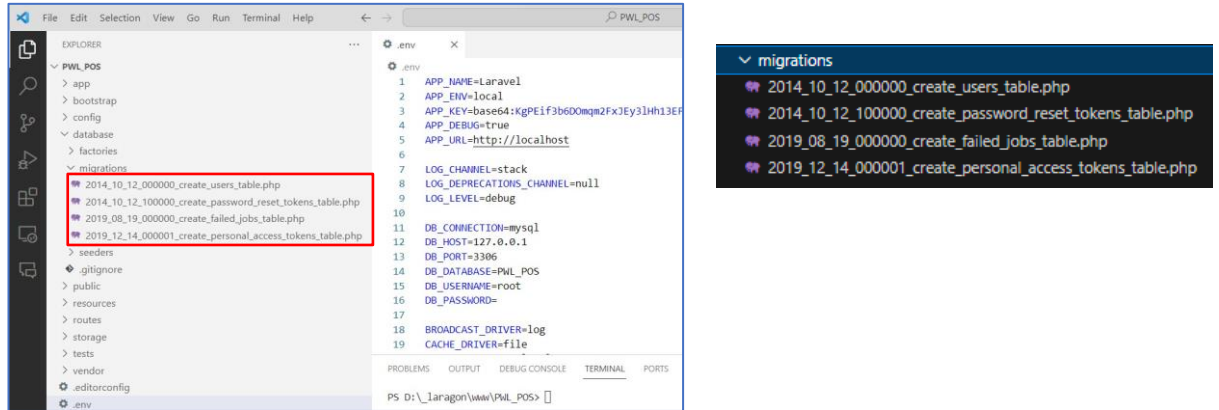
Pada Laravel, file-file *migration* ataupun *seeder* berada pada folder [PWL_POS/database](#)





Praktikum 2.1 - Pembuatan file migrasi tanpa relasi

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

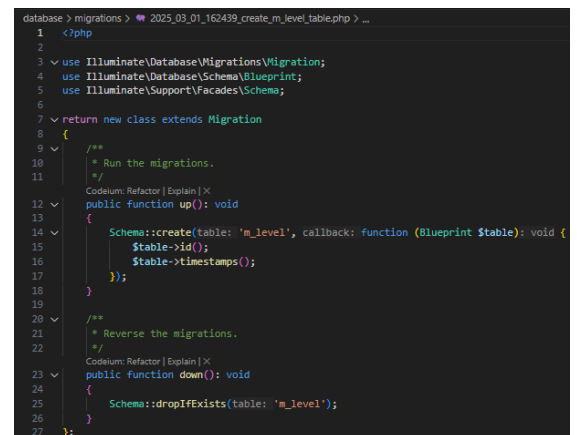
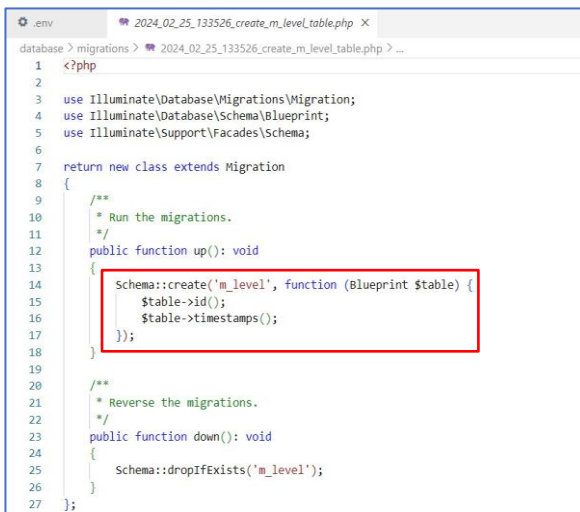


2. Kita abaikan dulu yang di kotak merah (jangan di hapus)
3. Kita buat file migrasi untuk table m_level dengan perintah

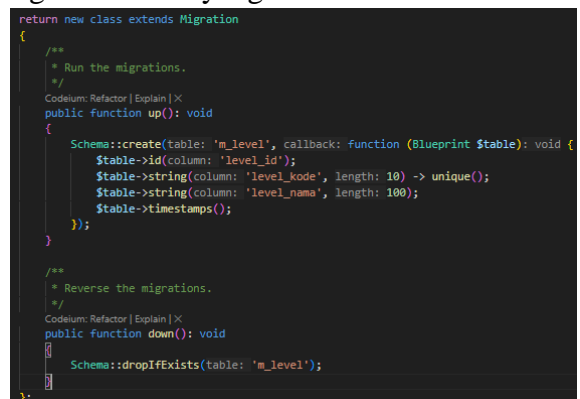
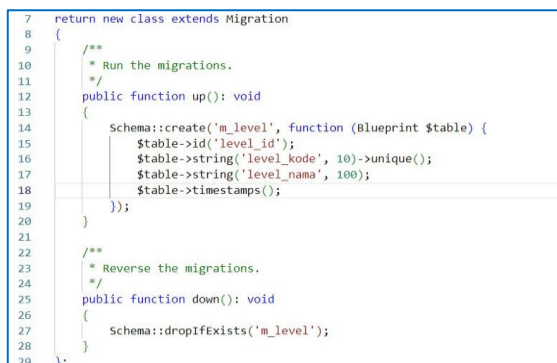
```
PS C:\laragon\www\PWL2025\PWL_Week3\Js3\PWL_POS> php artisan make:migration create_m_level_table --create=m_level
```

INFO Migration [C:\laragon\www\PWL2025\PWL_Week3\Js3\PWL_POS\database\migrations\2025_03_01_162439_create_m_level_table.php] created successfully.

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



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





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>

- 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> |
```

```
PS C:\laragon\www\PWL2025\PWL_Week3\Js3\PWL_POS> php artisan migrate
```

INFO Preparing database.

Creating migration table 520ms DONE

INFO Running migrations.

2014_10_12_000000_create_users_table 258ms DONE
2014_10_12_100000_create_password_reset_tokens_table 23ms DONE
2019_08_19_000000_create_failed_jobs_table 240ms DONE
2019_12_14_000001_create_personal_access_tokens_table 88ms DONE
2025_03_01_162439_create_m_level_table 63ms DONE

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

Table	Action	Table	Action
<input type="checkbox"/> failed_jobs	★ Browse Structure Search	<input type="checkbox"/> failed_jobs	★
<input type="checkbox"/> migrations	★ Browse Structure Search	<input type="checkbox"/> migrations	★
<input type="checkbox"/> <u>m_level</u>	★ Browse Structure Search	<input type="checkbox"/> <u>m_level</u>	★
<input type="checkbox"/> password_reset_tokens	★ Browse Structure Search	<input type="checkbox"/> password_reset_tokens	★
<input type="checkbox"/> personal_access_tokens	★ Browse Structure Search	<input type="checkbox"/> personal_access_tokens	★
<input type="checkbox"/> users	★ Browse Structure Search	<input type="checkbox"/> users	★

- Ok, table sudah dibuat di database
- Buat table *database* dengan *migration* untuk table **m_kategori** yang sama-sama tidak memiliki *foreign key*

```
PS C:\laragon\www\PWL2025\PWL_Week3\Js3\PWL_POS> php artisan make:migration create_m_kategori_table --create=m_kategori
```

INFO Migration [C:\laragon\www\PWL2025\PWL_Week3\Js3\PWL_POS\database\migrations\2025_03_01_164135_create_m_kategori_table.php] created successfully.



```
return new class extends Migration
{
    /**
     * Run the migrations.
     */
    public function up(): void {
        Schema::create('m_kategori', function (Blueprint $table) {
            $table->id('kategori_id'); // Primary Key
            $table->string('kategori_kode', 10);
            $table->string('kategori_nama', 100);
            $table->timestamps();
        });
    }
}
```

```
PS C:\laragon\www\PWL2025\PWL_Week3\Js3\PWL_POS> php artisan migrate

INFO Running migrations.

2025_03_01_164135_create_m_kategori_table ..... 38ms DONE
```

m_kategori Browse Structure Search Insert Empty Drop 0 InnoDB utf8mb4_unicode_ci 16.0 KiB

9. Laporkan hasil Praktikum-2.1 ini dan *commit* perubahan pada *git*.

Praktikum 2.2 - 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
```

```
PS C:\laragon\www\PWL2025\PWL_Week3\Js3\PWL_POS> php artisan make:migration create_m_user_table --table=m_user

INFO Migration [C:\laragon\www\PWL2025\PWL_Week3\Js3\PWL_POS\database\migrations\2025_03_02_132815_create_m_user_table.php] created successfully.
```

2. Buka file migrasi untuk table **m_user**, dan modifikasi seperti berikut

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

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

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

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

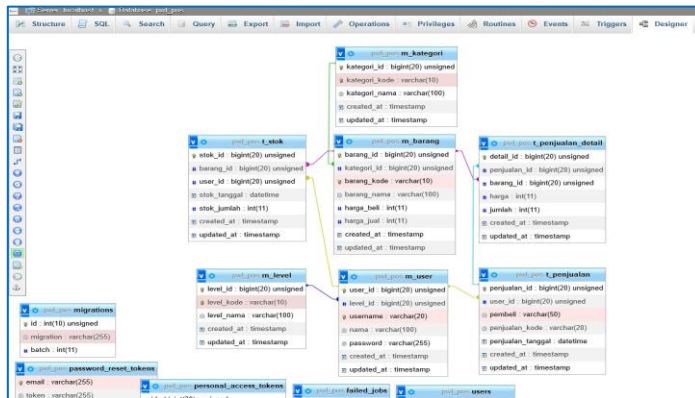
            // Mendefinisikan Foreignkey pada kolom level_id mengacu pada kolom level_id pada tabel m_level
            $table->foreign('level_id')->references('level_id')->on('m_level');
        });
    }

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

3. Simpan kode program Langkah 2, dan jalankan perintah **php artisan migrate**. Amati apa yang terjadi pada database.
4. Buat table *database* dengan *migration* untuk table-table yang memiliki *foreign key*

m_barang
t_penjualan
t_stok
t_penjualan_detail

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



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

Langkah 1: Buat file migrasi

```
INFO Migration [C:\laragon\www\PWL2025\PWL_Week3\Js3\PWL_POS\database\migrations\2025_03_02_140821_create_m_barang_table.p] created successfully.
hp] created successfully.

PS C:\laragon\www\PWL2025\PWL_Week3\Js3\PWL_POS> php artisan make:migration create_t_penjualan_table --create=t_penjualan

INFO Migration [C:\laragon\www\PWL2025\PWL_Week3\Js3\PWL_POS\database\migrations\2025_03_02_140821_create_t_penjualan_table.php] created successfully.

PS C:\laragon\www\PWL2025\PWL_Week3\Js3\PWL_POS> php artisan make:migration create_t_stok_table --create=t_stok

INFO Migration [C:\laragon\www\PWL2025\PWL_Week3\Js3\PWL_POS\database\migrations\2025_03_02_140822_create_t_stok_table.php] created successfully.

PS C:\laragon\www\PWL2025\PWL_Week3\Js3\PWL_POS> php artisan make:migration create_t_penjualan_detail_table --create=t_penjualan_detail

INFO Migration [C:\laragon\www\PWL2025\PWL_Week3\Js3\PWL_POS\database\migrations\2025_03_02_140823_create_t_penjualan_detail_table.php] created successfully.
```

Langkah 2: Modifikasi file migrasi

➤ File migrasi untuk m_barang

```
public function up(): void
{
    Schema::create(table: 'm_barang', callback: function (Blueprint $table): void {
        $table->id(column: 'barang_id');
        $table->unsignedBigInteger(column: 'kategori_id'); // FK ke m_kategori
        $table->string(column: 'barang_kode', length: 10)->unique();
        $table->string(column: 'barang_nama', length: 100);
        $table->integer(column: 'harga_beli');
        $table->integer(column: 'harga_jual');
        $table->timestamps();

        // Foreign Key
        $table->foreign(columns: 'kategori_id')->references(columns: 'kategori_id')->on(table: 'm_kategori');
    });
}
```

➤ File migrasi untuk t_penjualan

```
public function up(): void
{
    Schema::create(table: 't_penjualan', callback: function (Blueprint $table): void {
        $table->id(column: 'penjualan_id');
        $table->unsignedBigInteger(column: 'user_id'); // FK ke m_user
        $table->string(column: 'pembeli', length: 50);
        $table->string(column: 'penjualan_kode', length: 20)->unique();
        $table->dateTime(column: 'penjualan_tanggal');
        $table->timestamps();

        // Foreign Key
        $table->foreign(columns: 'user_id')->references(columns: 'user_id')->on(table: 'm_user');
    });
}
```




➤ File migrasi untuk t_stok

```
public function up(): void
{
    Schema::create(table: 't_stok', callback: function (Blueprint $table): void {
        $table->id(column: 'stok_id');
        $table->unsignedBigInteger(column: 'barang_id'); // FK ke m_barang
        $table->unsignedBigInteger(column: 'user_id'); // FK ke m_user
        $table->dateTime(column: 'stok_tanggal');
        $table->integer(column: 'stok_jumlah');
        $table->timestamps();

        // Foreign Key
        $table->foreign(columns: 'barang_id')->references(columns: 'barang_id')->on(table: 'm_barang');
        $table->foreign(columns: 'user_id')->references(columns: 'user_id')->on(table: 'm_user');
    });
}
```

➤ File migrasi untuk t_penjualan_detail

```
public function up(): void
{
    Schema::create(table: 't_penjualan_detail', callback: function (Blueprint $table): void {
        $table->id(column: 'detail_id');
        $table->unsignedBigInteger(column: 'penjualan_id'); // FK ke t_penjualan
        $table->unsignedBigInteger(column: 'barang_id'); // FK ke m_barang
        $table->integer(column: 'harga');
        $table->integer(column: 'jumlah');
        $table->timestamps();

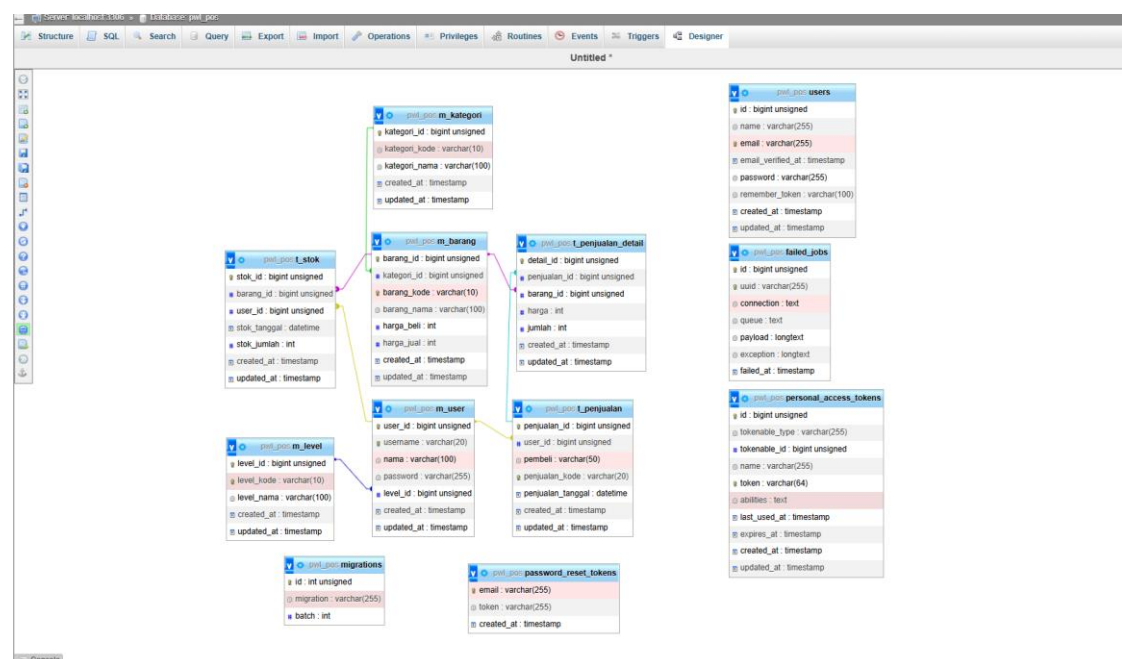
        // Foreign Key
        $table->foreign(columns: 'penjualan_id')->references(columns: 'penjualan_id')->on(table: 't_penjualan');
        $table->foreign(columns: 'barang_id')->references(columns: 'barang_id')->on(table: 'm_barang');
    });
}
```

Langkah 3: Jalankan migrasi

```
PS C:\laragon\www\PWL2025\PWL_Week3\Js3\PWL_POS> php artisan migrate

INFO Running migrations.

2014_10_12_000000_create_users_table ..... 186ms DONE
2014_10_12_100000_create_password_reset_tokens_table ..... 31ms DONE
2019_08_19_000000_create_failed_jobs_table ..... 221ms DONE
2019_12_14_000001_create_personal_access_tokens_table ..... 104ms DONE
2025_03_01_162439_create_m_level_table ..... 91ms DONE
2025_03_01_164135_create_m_kategori_table ..... 29ms DONE
2025_03_02_132815_create_m_user_table ..... 302ms DONE
2025_03_02_140821_create_m_barang_table ..... 157ms DONE
2025_03_02_140821_create_t_penjualan_table ..... 181ms DONE
2025_03_02_140822_create_t_stok_table ..... 196ms DONE
2025_03_02_140823_create_t_penjualan_detail_table ..... 183ms DONE
```





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/seeder**s

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

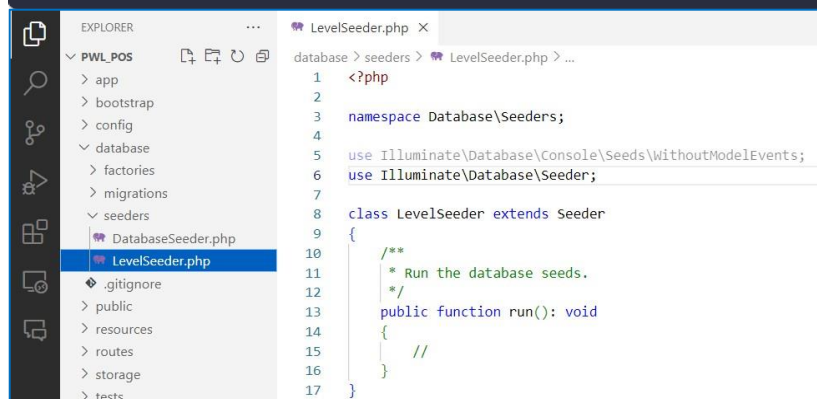
```
php artisan db:seed --class=<nama-class-seeder>
```

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.

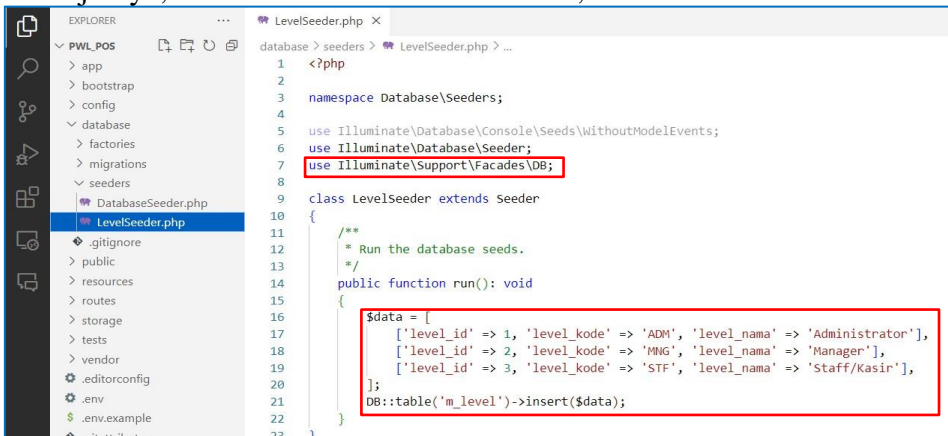
Praktikum 3 – Membuat file *seeder*

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

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



2. Selanjutnya, untuk memasukkan data awal, kita modifikasi file tersebut di dalam





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> █
```

```
PS C:\laragon\www\PWL2025\PWL_Week3\Js3\PWL_POS> php artisan db:seed --class=LevelSeeder
INFO Seeding database.
```

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

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

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

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

6. Modifikasi file `class UserSeeder` seperti berikut

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



7. Jalankan perintah untuk mengeksekusi class **UserSeeder**

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

```
PS C:\laragon\www\PWL2025\PWL_Week3\Js3\PWL_POS> php artisan db:seed --class=UserSeeder  
INFO Seeding database.
```

8. Perhatikan hasil seeder pada table **m_user**

	user_id	level_id	username	nama	password
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	1	1	admin	Administrator	\$2y\$12\$Tevu4dD01CUAQpeM8H Vp LySwY4oAKU7FzwS80V...
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	2	2	manager	Manager	\$2y\$12\$Ajms20/FdPteUgghz31muEhIFanLxkt5wvZ9NGRpu...
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	3	3	staff	Staff Kasir	\$2y\$12\$GIZ3TqGclW5pYeR0VL4o5OxPwb30sk9VMYBhnbJ9W...

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

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

1. Buat Seeder dengan Artisan Command

```
PS C:\laragon\www\PWL2025\PWL_Week3\Js3\PWL_POS> php artisan make:seeder BarangSeeder  
INFO Seeder [C:\laragon\www\PWL2025\PWL_Week3\Js3\PWL_POS\database\seeders\BarangSeeder.php] created successfully.  
PS C:\laragon\www\PWL2025\PWL_Week3\Js3\PWL_POS> php artisan make:seeder StokSeeder  
INFO Seeder [C:\laragon\www\PWL2025\PWL_Week3\Js3\PWL_POS\database\seeders\StokSeeder.php] created successfully.  
PS C:\laragon\www\PWL2025\PWL_Week3\Js3\PWL_POS> php artisan make:seeder PenjualanSeeder  
INFO Seeder [C:\laragon\www\PWL2025\PWL_Week3\Js3\PWL_POS\database\seeders\PenjualanSeeder.php] created successfully.  
PS C:\laragon\www\PWL2025\PWL_Week3\Js3\PWL_POS> php artisan make:seeder PenjualanDetailSeeder  
INFO Seeder [C:\laragon\www\PWL2025\PWL_Week3\Js3\PWL_POS\database\seeders\PenjualanDetailSeeder.php] created successfully.
```

2. Modifikasi Seeder

➤ Seeder KategoriSeeder.php

```
database > seeders > KategoriSeeder.php > ...  
1 <?php  
2  
3 namespace Database\Seeders;  
4  
5 use Illuminate\Database\Seeder;  
6 use Illuminate\Support\Facades\DB;  
7  
8  
9 0 references | 0 implementations | Codeium: Refactor | Explain  
10 class KategoriSeeder extends Seeder  
11 {  
12 0 references | 0 overrides | Codeium: Refactor | Explain | Generate Function Comment | X  
13 public function run(): void  
14 {  
15     $data = [  
16         ['kategori_id' => 1, 'kategori_kode' => 'KTG001', 'kategori_nama' => 'Makanan'],  
17         ['kategori_id' => 2, 'kategori_kode' => 'KTG002', 'kategori_nama' => 'Minuman'],  
18         ['kategori_id' => 3, 'kategori_kode' => 'KTG003', 'kategori_nama' => 'Elektronik'],  
19         ['kategori_id' => 4, 'kategori_kode' => 'KTG004', 'kategori_nama' => 'Peralatan Rumah'],  
20         ['kategori_id' => 5, 'kategori_kode' => 'KTG005', 'kategori_nama' => 'Pakaian'],  
21     ];  
22     DB::table('m_kategori')->insert(values: $data);  
23 }
```



➤ Seeder BarangSeeder.php

```
database > seeders > BarangSeeder.php > ...
1 <?php
2
3 namespace Database\Seeders;
4
5 use Illuminate\Database\Seeder;
6 use Illuminate\Support\Facades\DB;
7
8 class BarangSeeder extends Seeder
9 {
10     public function run(): void
11     {
12         $data = [
13             ['barang_id' => 1, 'kategori_id' => 1, 'barang_kode' => 'BRG001', 'barang_nama' => 'Biskuit', 'harga_beli' => 3000, 'harga_jual' => 5000],
14             ['barang_id' => 2, 'kategori_id' => 1, 'barang_kode' => 'BRG002', 'barang_nama' => 'Roti', 'harga_beli' => 5000, 'harga_jual' => 7000],
15             ['barang_id' => 3, 'kategori_id' => 2, 'barang_kode' => 'BRG003', 'barang_nama' => 'Teh Botol', 'harga_beli' => 2000, 'harga_jual' => 3000],
16             ['barang_id' => 4, 'kategori_id' => 2, 'barang_kode' => 'BRG004', 'barang_nama' => 'Kopi Instan', 'harga_beli' => 3000, 'harga_jual' => 4000],
17             ['barang_id' => 5, 'kategori_id' => 3, 'barang_kode' => 'BRG005', 'barang_nama' => 'Mouse Wireless', 'harga_beli' => 100000, 'harga_jual' => 150000],
18             ['barang_id' => 6, 'kategori_id' => 3, 'barang_kode' => 'BRG006', 'barang_nama' => 'Keyboard Mechanical', 'harga_beli' => 250000, 'harga_jual' => 350000],
19             ['barang_id' => 7, 'kategori_id' => 4, 'barang_kode' => 'BRG007', 'barang_nama' => 'Sapu', 'harga_beli' => 15000, 'harga_jual' => 20000],
20             ['barang_id' => 8, 'kategori_id' => 4, 'barang_kode' => 'BRG008', 'barang_nama' => 'Pel', 'harga_beli' => 20000, 'harga_jual' => 25000],
21             ['barang_id' => 9, 'kategori_id' => 5, 'barang_kode' => 'BRG009', 'barang_nama' => 'Kaos Polos', 'harga_beli' => 40000, 'harga_jual' => 50000],
22             ['barang_id' => 10, 'kategori_id' => 5, 'barang_kode' => 'BRG010', 'barang_nama' => 'Jaket Hoodie', 'harga_beli' => 100000, 'harga_jual' => 120000],
23         ];
24
25         DB::table(table: 'm_barang')->insert(values: $data);
26     }
27 }
```

➤ Seeder StokSeeder.php

```
database > seeders > StokSeeder.php > PHP Intelephense > StokSeeder
1 <?php
2
3 namespace Database\Seeders;
4
5 use Illuminate\Database\Seeder;
6 use Illuminate\Support\Facades\DB;
7 use Carbon\Carbon;
8
9 class StokSeeder extends Seeder
10 {
11     public function run(): void
12     {
13         $data = [];
14         for ($i = 1; $i <= 10; $i++) {
15             $data[] = [
16                 'stok_id' => $i,
17                 'barang_id' => $i,
18                 'user_id' => 3, // User acak
19                 'stok_tanggal' => Carbon::now()->subDays(value: rand(min: 1, max: 30))->toDateTimeString(),
20                 'stok_jumlah' => rand(min: 10, max: 100),
21             ];
22         }
23
24         DB::table(table: 't_stok')->insert(values: $data);
25     }
26 }
```

➤ Seeder PenjualanSeeder.php

```
database > seeders > PenjualanSeeder.php > PHP Intelephense > PenjualanSeeder > run
1 <?php
2
3 namespace Database\Seeders;
4
5 use Illuminate\Database\Seeder;
6 use Illuminate\Support\Facades\DB;
7 use Carbon\Carbon;
8
9 class PenjualanSeeder extends Seeder
10 {
11     public function run(): void
12     {
13         $data = [];
14         for ($i = 1; $i <= 10; $i++) {
15             $data[] = [
16                 'penjualan_id' => $i,
17                 'user_id' => rand(min: 1, max: 3), // User acak
18                 'pembeli' => "Pelanggan " . $i,
19                 'penjualan_kode' => "PID" . str_pad(string: $i, length: 4, pad_string: '0', pad_type: STR_PAD_LEFT),
20                 'penjualan_tanggal' => Carbon::now()->subDays(value: rand(min: 0, max: 10))->toDateTimeString(),
21             ];
22         }
23
24         DB::table(table: 't_penjualan')->insert(values: $data);
25     }
26 }
```

➤ Seeder PenjualanDetailSeeder.php

```
database > seeders > PenjualanDetailSeeder.php > PHP > PenjualanDetailSeeder > run
1 <?php
2
3 namespace Database\Seeders;
4
5 use Illuminate\Database\Seeder;
6 use Illuminate\Support\Facades\DB;
7
8 class PenjualanDetailSeeder extends Seeder
9 {
10     public function run(): void
11     {
12         $data = [];
13         for ($i = 1; $i <= 10; $i++) { // 10 transaksi
14             for ($j = 1; $j <= 3; $j++) { // 3 barang per transaksi
15                 $barang_id = rand(min: 1, max: 10);
16                 $harga = DB::table(table: 'm_barang')->where(column: 'barang_id', operator: '=', value: ($barang_id))->value(column: 'harga_jual');
17                 $jumlah = rand(min: 1, max: 5);
18
19                 $data[] = [
20                     'detail_id' => count(value: $data) + 1,
21                     'penjualan_id' => $i,
22                     'barang_id' => $barang_id,
23                     'harga' => $harga,
24                     'jumlah' => $jumlah,
25                 ];
26             }
27         }
28
29         DB::table(table: 't_penjualan_detail')->insert(values: $data);
30     }
31 }
```




3. Jalankan Seeder

```
PS C:\laragon\www\PWL2025\PWL_Week3\Js3\PWL_POS> php artisan db:seed --class=KategoriSeeder

INFO Seeding database.

PS C:\laragon\www\PWL2025\PWL_Week3\Js3\PWL_POS> php artisan db:seed --class=BarangSeeder

INFO Seeding database.

PS C:\laragon\www\PWL2025\PWL_Week3\Js3\PWL_POS> php artisan db:seed --class=StokSeeder

INFO Seeding database.

PS C:\laragon\www\PWL2025\PWL_Week3\Js3\PWL_POS> php artisan db:seed --class=PenjualanSeeder

INFO Seeding database.

PS C:\laragon\www\PWL2025\PWL_Week3\Js3\PWL_POS> php artisan db:seed --class=PenjualanDetailSeeder

INFO Seeding database.
```

4. Periksa Data di Database

Showing rows 0 - 4 (5 total, Query took 0.0013 seconds)

SELECT * FROM m_kategori;

Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

Show all | Number of rows: 25 | Filter rows: Search this table | Sort by key: None

	kategori_id	kategori_kode	kategori_nama	created_at	updated_at
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	1	KTG001	Makanan	NULL	NULL
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	2	KTG002	Minuman	NULL	NULL
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	3	KTG003	Elektronik	NULL	NULL
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	4	KTG004	Peralatan Rumah	NULL	NULL
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	5	KTG005	Pakaian	NULL	NULL

Showing rows 0 - 9 (10 total, Query took 0.0004 seconds)

SELECT * FROM t_penjualan;

Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

Show all | Number of rows: 25 | Filter rows: Search this table | Sort by key: None

	penjualan_id	user_id	pembeli	penjualan_kode	penjualan_tanggal	created_at	updated_at
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	1	3	Pelanggan 1	PNJ0001	2025-02-28 16:47:08	NULL	NULL
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	2	3	Pelanggan 2	PNJ0002	2025-03-01 16:47:08	NULL	NULL
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	3	3	Pelanggan 3	PNJ0003	2025-02-26 16:47:08	NULL	NULL
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	4	3	Pelanggan 4	PNJ0004	2025-02-21 16:47:08	NULL	NULL
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	5	3	Pelanggan 5	PNJ0005	2025-02-27 16:47:08	NULL	NULL
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	6	3	Pelanggan 6	PNJ0006	2025-02-20 16:47:08	NULL	NULL
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	7	3	Pelanggan 7	PNJ0007	2025-02-24 16:47:08	NULL	NULL
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	8	3	Pelanggan 8	PNJ0008	2025-02-22 16:47:08	NULL	NULL
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	9	3	Pelanggan 9	PNJ0009	2025-03-01 16:47:08	NULL	NULL
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	10	3	Pelanggan 10	PNJ0010	2025-02-24 16:47:08	NULL	NULL

Showing rows 0 - 9 (10 total, Query took 0.0004 seconds)

SELECT * FROM t_stok;

Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

Show all | Number of rows: 25 | Filter rows: Search this table | Sort by key: None

	stok_id	barang_id	user_id	stok_tanggal	stok_jumlah	created_at	updated_at
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	1	1	3	2025-02-14 16:47:08	34	NULL	NULL
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	2	2	3	2025-02-15 16:47:08	66	NULL	NULL
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	3	3	3	2025-02-17 16:47:08	54	NULL	NULL
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	4	4	3	2025-02-18 16:47:08	43	NULL	NULL
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	5	5	3	2025-02-21 16:47:08	68	NULL	NULL
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	6	6	3	2025-02-08 16:47:08	100	NULL	NULL
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	7	7	3	2025-02-25 16:47:08	55	NULL	NULL
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	8	8	3	2025-02-04 16:47:08	53	NULL	NULL
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	9	9	3	2025-02-16 16:47:08	92	NULL	NULL
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	10	10	3	2025-02-06 16:47:08	64	NULL	NULL

Showing rows 0 - 9 (10 total, Query took 0.0004 seconds)

SELECT * FROM m_barang;

Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

Show all | Number of rows: 25 | Filter rows: Search this table | Sort by key: None

	barang_id	kategori_id	barang_kode	barang_nama	harga_beli	harga_jual	created_at	updated_at
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	1	1	BRG0001	Biskuit	3000	5000	NULL	NULL
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	2	1	BRG0002	Roti	5000	7000	NULL	NULL
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	3	2	BRG0003	Teh Botol	2000	3000	NULL	NULL
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	4	2	BRG0004	Kopi Instan	3000	4000	NULL	NULL
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	5	3	BRG0005	Mouse Wireless	100000	150000	NULL	NULL
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	6	3	BRG0006	Keyboard Mechanical	250000	350000	NULL	NULL
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	7	4	BRG0007	Sapu	15000	20000	NULL	NULL
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	8	4	BRG0008	Pel	20000	25000	NULL	NULL
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	9	5	BRG0009	Kaos Polo	40000	50000	NULL	NULL
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	10	5	BRG0010	Jaket Hoodie	100000	120000	NULL	NULL

Showing rows 0 - 24 (30 total, Query took 0.0004 seconds)

SELECT * FROM t_penjualan_detail;

Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

Show all | Number of rows: 25 | Filter rows: Search this table | Sort by key: None

	detail_id	penjualan_id	barang_id	harga	jumlah	created_at	updated_at
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	1	1	1	5000	2	NULL	NULL
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	2	1	1	5000	5	NULL	NULL
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	3	1	9	50000	1	NULL	NULL
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	4	2	1	5000	4	NULL	NULL
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	5	2	8	25000	3	NULL	NULL
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	6	2	10	120000	2	NULL	NULL
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	7	3	2	7000	2	NULL	NULL
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	8	3	6	350000	5	NULL	NULL
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	9	3	5	150000	3	NULL	NULL
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	10	4	4	4000	5	NULL	NULL
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	11	4	6	350000	2	NULL	NULL
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	12	4	2	7000	1	NULL	NULL
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	13	5	9	50000	1	NULL	NULL
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	14	5	6	350000	1	NULL	NULL
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	15	5	7	20000	3	NULL	NULL
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	16	6	10	120000	4	NULL	NULL
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	17	6	10	120000	4	NULL	NULL
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	18	6	5	150000	5	NULL	NULL
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	19	7	9	50000	3	NULL	NULL
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	20	7	1	5000	2	NULL	NULL
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	21	7	6	350000	5	NULL	NULL
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	22	8	6	350000	4	NULL	NULL
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	23	8	9	50000	2	NULL	NULL
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	24	8	8	25000	4	NULL	NULL
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	25	9	10	120000	1	NULL	NULL



D. DB FACADE

DB Façade merupakan fitur dari Laravel yang digunakan untuk melakukan *query* secara langsung dengan mengetikkan perintah 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 **mengembalikan** (*return*) data hasil *query*. Contoh

```
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']);
```

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
```

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

```
LevelController.php  web.php  X
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
11 Route::get('/level', [LevelController::class, 'index']);
```

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

```
LevelController.php  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 }
```



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

➤ Sebelum

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

➤ Sesudah



	level_id	level_kode	level_nama	created_at	updated_at
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	1	ADM	Administrator	NULL	NULL
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	2	MNG	Manager	NULL	NULL
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	3	STF	Staff	NULL	NULL
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	4	CUS	Pelanggan	2025-03-03 02:40:04	NULL

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

	level_id	level_kode	level_nama	created_at	updated_at
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	1	ADM	Administrator	NULL	NULL
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	2	MNG	Manager	NULL	NULL
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	3	STF	Staff/Kasir	NULL	NULL
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	4	CUS	Pelanggan	2024-02-26 08:20:00	NULL

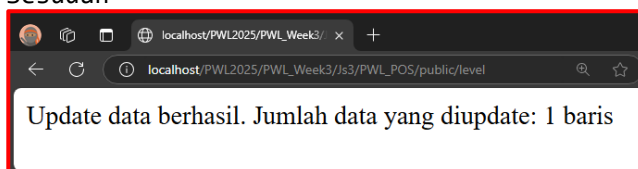
6. Kita coba jalankan di browser dengan url localhost/PWL_POS/public/level lagi dan amati apa yang terjadi pada table `m_level` di database, *screenshot* perubahan yang ada pada table `m_level`

```
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 }
```

➤ Sebelum

	level_id	level_kode	level_nama	created_at	updated_at
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	1	ADM	Administrator	NULL	NULL
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	2	MNG	Manager	NULL	NULL
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	3	STF	Staff	NULL	NULL
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	4	CUS	Pelanggan	2025-03-03 02:40:04	NULL

➤ Sesudah



	level_id	level_kode	level_nama	created_at	updated_at
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	1	ADM	Administrator	NULL	NULL
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	2	MNG	Manager	NULL	NULL
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	3	STF	Staff	NULL	NULL
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	4	CUS	Customer	2025-03-03 02:40:04	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 }
```

➤ Sebelum

	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	NULL	NULL
<input type="checkbox"/>	4	CUS	Customer	2025-03-03 02:40:04	NULL

➤ Sesudah

Hapus data berhasil. Jumlah data yang dihapus: 1 baris

	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	NULL	NULL

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

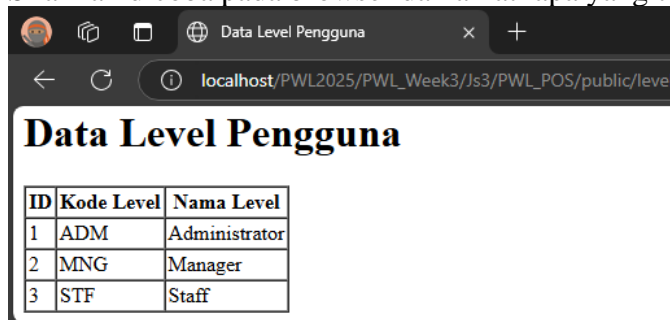
```
3 namespace App\Http\Controllers;
4
5 use Illuminate\Http\Request;
6 use Illuminate\Support\Facades\DB;
7
8 class LevelController extends Controller
9 {
10     public function index()
11     {
12         // DB::insert('insert into m_level(level_kode, level_nama, created_at) values(?, ?, ?)', ['CUS', '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 }
```

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/`



```
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>
```

10. Silahkan dicoba pada browser dan amati apa yang terjadi



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

- 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');
```

- 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

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

Praktikum 5 – Implementasi *Query Builder*

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

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

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

```
LevelController.php  KategoriController.php  level.blade.php  web.php X
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`

```
LevelController.php  KategoriController.php X  level.blade.php  web.php
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 }
```

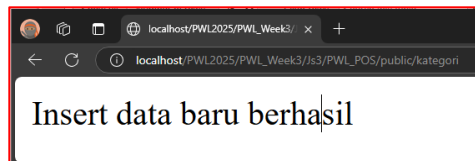


4. Kita coba jalankan di browser dengan url localhost/PWL_POS/public/kategori dan amati apa yang terjadi pada table `m_kategori` di database, *screenshot* perubahan yang ada pada table `m_kategori`

➤ Sebelum

		kategori_id	kategori_kode	kategori_nama	created_at	updated_at
<input type="checkbox"/>	Edit					
<input type="checkbox"/>	Copy					
<input type="checkbox"/>	Delete					
		1	KTG001	Makanan	NULL	NULL
<input type="checkbox"/>	Edit					
<input type="checkbox"/>	Copy					
<input type="checkbox"/>	Delete					
		2	KTG002	Minuman	NULL	NULL
<input type="checkbox"/>	Edit					
<input type="checkbox"/>	Copy					
<input type="checkbox"/>	Delete					
		3	KTG003	Elektronik	NULL	NULL
<input type="checkbox"/>	Edit					
<input type="checkbox"/>	Copy					
<input type="checkbox"/>	Delete					
		4	KTG004	Peralatan Rumah	NULL	NULL
<input type="checkbox"/>	Edit					
<input type="checkbox"/>	Copy					
<input type="checkbox"/>	Delete					
		5	KTG005	Pakaian	NULL	NULL

➤ Sesudah



		kategori_id	kategori_kode	kategori_nama	created_at	updated_at
<input type="checkbox"/>	Edit					
<input type="checkbox"/>	Copy					
<input type="checkbox"/>	Delete					
		1	KTG001	Makanan	NULL	NULL
<input type="checkbox"/>	Edit					
<input type="checkbox"/>	Copy					
<input type="checkbox"/>	Delete					
		2	KTG002	Minuman	NULL	NULL
<input type="checkbox"/>	Edit					
<input type="checkbox"/>	Copy					
<input type="checkbox"/>	Delete					
		3	KTG003	Elektronik	NULL	NULL
<input type="checkbox"/>	Edit					
<input type="checkbox"/>	Copy					
<input type="checkbox"/>	Delete					
		4	KTG004	Peralatan Rumah	NULL	NULL
<input type="checkbox"/>	Edit					
<input type="checkbox"/>	Copy					
<input type="checkbox"/>	Delete					
		5	KTG005	Pakaian	NULL	NULL
<input type="checkbox"/>	Edit					
<input type="checkbox"/>	Copy					
<input type="checkbox"/>	Delete					
		6	SNK	Snack/Makanan Ringan	2025-03-03 03:46:19	NULL

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

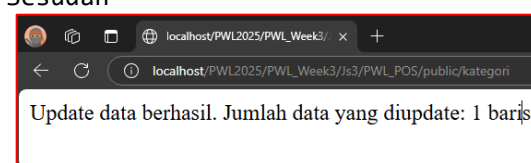
```
app > Http > Controllers > KategoriController.php > KategoriController > index
1  <?php
2
3  namespace App\Http\Controllers;
4
5  use Illuminate\Http\Request;
6  use Illuminate\Support\Facades\DB;
7
8  class KategoriController extends Controller
9  {
10     public function index()
11     {
12   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

➤ Sebelum

		kategori_id	kategori_kode	kategori_nama	created_at	updated_at
<input type="checkbox"/>	Edit					
<input type="checkbox"/>	Copy					
<input type="checkbox"/>	Delete					
		1	KTG001	Makanan	NULL	NULL
<input type="checkbox"/>	Edit					
<input type="checkbox"/>	Copy					
<input type="checkbox"/>	Delete					
		2	KTG002	Minuman	NULL	NULL
<input type="checkbox"/>	Edit					
<input type="checkbox"/>	Copy					
<input type="checkbox"/>	Delete					
		3	KTG003	Elektronik	NULL	NULL
<input type="checkbox"/>	Edit					
<input type="checkbox"/>	Copy					
<input type="checkbox"/>	Delete					
		4	KTG004	Peralatan Rumah	NULL	NULL
<input type="checkbox"/>	Edit					
<input type="checkbox"/>	Copy					
<input type="checkbox"/>	Delete					
		5	KTG005	Pakaian	NULL	NULL
<input type="checkbox"/>	Edit					
<input type="checkbox"/>	Copy					
<input type="checkbox"/>	Delete					
		6	SNK	Snack/Makanan Ringan	2025-03-03 03:46:19	NULL

➤ Sesudah



		kategori_id	kategori_kode	kategori_nama	created_at	updated_at
<input type="checkbox"/>	Edit					
<input type="checkbox"/>	Copy					
<input type="checkbox"/>	Delete					
		1	KTG001	Makanan	NULL	NULL
<input type="checkbox"/>	Edit					
<input type="checkbox"/>	Copy					
<input type="checkbox"/>	Delete					
		2	KTG002	Minuman	NULL	NULL
<input type="checkbox"/>	Edit					
<input type="checkbox"/>	Copy					
<input type="checkbox"/>	Delete					
		3	KTG003	Elektronik	NULL	NULL
<input type="checkbox"/>	Edit					
<input type="checkbox"/>	Copy					
<input type="checkbox"/>	Delete					
		4	KTG004	Peralatan Rumah	NULL	NULL
<input type="checkbox"/>	Edit					
<input type="checkbox"/>	Copy					
<input type="checkbox"/>	Delete					
		5	KTG005	Pakaian	NULL	NULL
<input type="checkbox"/>	Edit					
<input type="checkbox"/>	Copy					
<input type="checkbox"/>	Delete					
		6	SNK	Camilan	2025-03-03 03:46:19	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 }
```

➤ **Sebelum**

	kategori_id	kategori_kode	kategori_nama	created_at	updated_at
  	1	KTG001	Makanan	NULL	NULL
  	2	KTG002	Minuman	NULL	NULL
  	3	KTG003	Elektronik	NULL	NULL
  	4	KTG004	Peralatan Rumah	NULL	NULL
  	5	KTG005	Pakaian	NULL	NULL
  	6	SNK	Snack/Makanan Ringan	2025-03-03 03:46:19	NULL

➤ **Sesudah**

localhost/PWL2025/PWL_Week3/ x +

← → local:localhost/PWL2025/PWL_Week3/js3/PWL_POS/public/kategori

Delete data berhasil. Jumlah data yang dihapus: 1 baris

	kategori_id	kategori_kode	kategori_nama	created_at	updated_at
  	1	KTG001	Makanan	NULL	NULL
  	2	KTG002	Minuman	NULL	NULL
  	3	KTG003	Elektronik	NULL	NULL
  	4	KTG004	Peralatan Rumah	NULL	NULL
  	5	KTG005	Pakaian	NULL	NULL

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

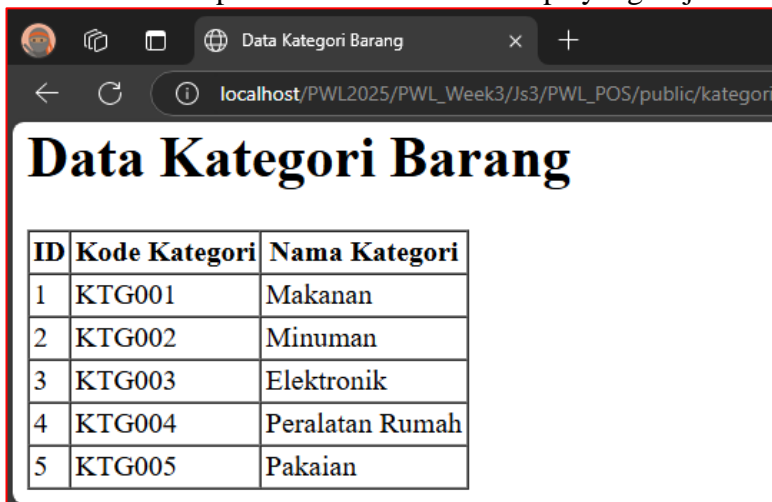
```
10 public function index()
11 {
12     /* $data = [
13         'kategori_kode' => 'SNK',
14         'kategori_nama' => 'Snack/Makanan Ringan',
15         'created_at' => now()
16     ];
17     DB::table('m_kategori')->insert($data);
18     return 'Insert data baru berhasil'; */
19
20     // $row = 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.



ID	Kode Kategori	Nama Kategori
1	KTG001	Makanan
2	KTG002	Minuman
3	KTG003	Elektronik
4	KTG004	Peralatan Rumah
5	KTG005	Pakaian

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

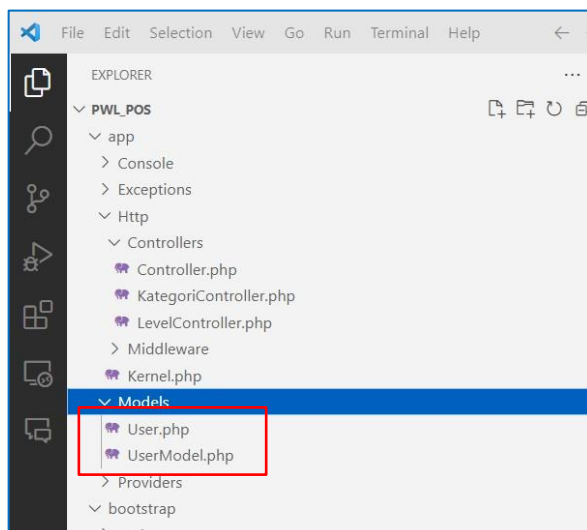
```
php artisan make:model <nama-model-CamelCase>
```

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
```



```
PS C:\laragon\www\PWL2025\PWL_Week3\Js3\PWL_POS> php artisan make:model UserModel  
INFO Model [C:\laragon\www\PWL2025\PWL_Week3\Js3\PWL_POS\app\Models\UserModel.php] created successfully.
```



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 }
```

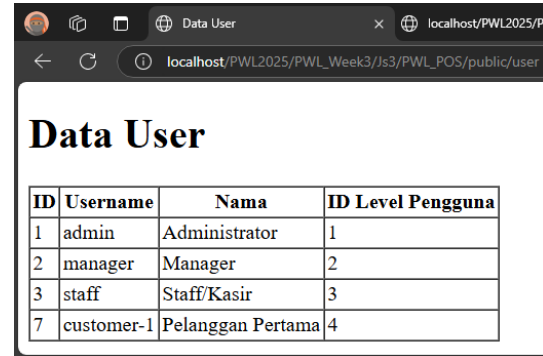
9. Jalankan di browser, amati dan laporkan apa yang terjadi

ID	Username	Nama	ID Level Pengguna
1	admin	Administrator	1
2	manager	Manager	2
3	staff	Staff/Kasir	3
7	customer-1	Pelanggan	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 }
```



ID	Username	Nama	ID Level Pengguna
1	admin	Administrator	1
2	manager	Manager	2
3	staff	Staff/Kasir	3
7	customer-1	Pelanggan Pertama	4

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

12. Jika sudah, laporkan 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?
2. Pada **Praktikum 1**, bagaimana kita *men-generate* nilai untuk **APP_KEY**?
3. Pada **Praktikum 2.1 - Tahap 1**, secara *default* Laravel memiliki berapa file migrasi? dan untuk apa saja file migrasi tersebut?
4. Secara *default*, file migrasi terdapat kode `$table->timestamps()`, apa tujuan/*output* dari fungsi tersebut?
5. Pada File Migrasi, terdapat fungsi `$table->id()`; Tipe data apa yang dihasilkan dari fungsi tersebut?
6. Apa bedanya hasil migrasi pada table `m_level`, antara menggunakan `$table->id()`; dengan menggunakan `$table->id('level_id')`; ?
7. Pada migration, Fungsi `->unique()` digunakan untuk apa?
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')` ?
9. Pada **Praktikum 3 - Tahap 6**, apa tujuan dari Class **Hash**? dan apa maksud dari kode program `Hash::make('1234')`;
10. Pada **Praktikum 4 - Tahap 3/5/7**, pada *query builder* terdapat tanda tanya (?), apa kegunaan dari tanda tanya (?) tersebut?
11. Pada **Praktikum 6 - Tahap 3**, apa tujuan penulisan kode `protected $table = 'm_user'`; dan `protected $primaryKey = 'user_id'` ; ?
12. Menurut kalian, lebih mudah menggunakan mana dalam melakukan operasi CRUD ke database (*DB Façade / Query Builder / Eloquent ORM*) ? jelaskan

Jawaban :

1. **APP_KEY** digunakan untuk mengenkripsi data dalam Laravel, termasuk session dan password hashing.
2. Untuk *men-generate* **APP_KEY**, gunakan perintah:
`php artisan key:generate`
3. Secara default, Laravel memiliki **3 file migrasi**, yaitu:



- `create_users_table.php` → Untuk tabel user
 - `create_password_resets_table.php` → Untuk reset password
 - `create_failed_jobs_table.php` → Untuk menyimpan job yang gagal
4. `$table->timestamps()`; akan membuat **kolom `created_at` dan `updated_at`** secara otomatis.
 5. `$table->id()`; menghasilkan tipe data **BIGINT dengan auto-increment**.
 6. `$table->id()`; membuat **nama default `id`**, sedangkan `$table->id('level_id')`; mengganti nama kolom menjadi **`level_id`**.
 7. `->unique()` digunakan untuk **mencegah duplikasi nilai pada kolom tertentu**.
 8. `level_id` di **`m_user` menggunakan `unsignedBigInteger`** karena harus cocok dengan tipe data `id('level_id')` di **`m_level`** yang berupa **BIGINT unsigned**.
 9. **Class Hash** digunakan untuk mengenkripsi password. `Hash::make('1234')`; mengubah "1234" menjadi **hash bcrypt** yang aman.
 10. Tanda `?` digunakan untuk **binding parameter** dalam query, agar mencegah **SQL Injection**.
 11. `protected $table = 'm_user'`; menentukan **nama tabel** yang digunakan model, sedangkan `protected $primaryKey = 'user_id'`; menentukan **primary key**.
 12. **Eloquent ORM lebih mudah** karena menggunakan sintaks **lebih simpel dan OOP-friendly**, dibanding **Query Builder** (lebih fleksibel) atau **DB Facade** (lebih manual).

*** Sekian, dan selamat belajar ***