

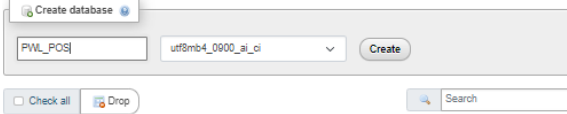
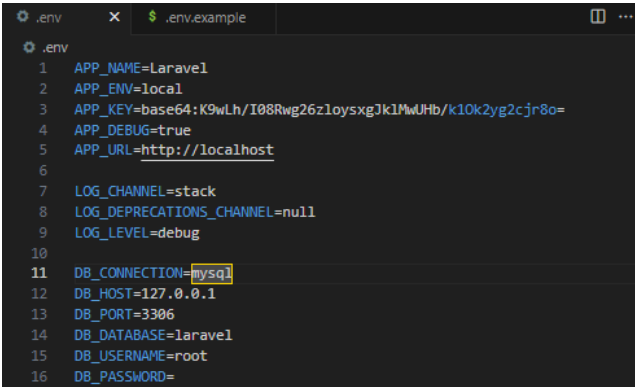


Mata Kuliah : Pemrograman Web Lanjut (PWL)
Program Studi : D4 – Sistem Informasi Bisnis
Semester : 5

Kelas : SIB
NIM : 2241760025
Nama : Hertin Nurhayati
Jobsheet Ke- : 3

Laporan Jobsheet

Praktikum Ke-1

Langkah	Jawaban/Deskripsi
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.	Membuat tabel :



- m_level
- m_kategori
- m_supplier

Praktikum Ke-2.1

Langkah	Jawaban/Deskripsi
1.	<p>Buat file migrasi untuk table m_level dengan perintah</p> <pre>C:\laragon\www\PWL_2024>php artisan make:migration create_m_level_table --create=m_level</pre> <pre>2024_09_12_155025_create_m_level_table.php U x database > migrations > 2024_09_12_155025_create_m_level_table.php 1 <?php 2 3 use Illuminate\Database\Migrations\Migration; 4 use Illuminate\Database\Schema\Blueprint; 5 use Illuminate\Support\Facades\Schema; 6 7 return new class extends Migration 8 { 9 /** 10 * Run the migrations. 11 */ 12 public function up(): void 13 { 14 Schema::create('m_level', function (Blueprint \$table) { 15 \$table->id(); 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 }</pre>
2.	<p>Kita perhatikan bagian yang di kotak merah, bagian tersebut yang akan kita modifikasi sesuai desain database yang sudah ada</p>



	
3.	<p>Simpan kode pada tahapan 4 tersebut, kemudian jalankan perintah ini pada terminal VSCode untuk melakukan migrasi</p> 
4.	<p>Kemudian kita cek di phpMyAdmin apakah table sudah ter-generate atau belum</p> 
5.	<p>Buat table database dengan migration untuk table m_kategori dan m_supplier yang sama-sama tidak memiliki foreign key</p> <p>Membuat table m_kategori</p> 



```
2024_09_12_160546_create_m_kategori_table.php U
database > migrations > 2024_09_12_160546_create_m_kategori_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_kategori', function (Blueprint $table) {
15             $table->id('kategori_id');
16             $table->id('kategori_kode',10)->unique();
17             $table->id('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 };
```

```
C:\laragon\www\PWL_2024>php artisan migrate
INFO Running migrations.
2024_09_12_160546_create_m_kategori_table ..... 109ms DONE
```

Membuat table m_supplier

```
C:\laragon\www\PWL_2024>php artisan make:migration create_m_supplier_table
--create=m_supplier
```



```
2024_09_12_161120_create_m_supplier_table.php U x
database > migrations > 2024_09_12_161120_create_m_supplier_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_supplier', function (Blueprint $table) {
15             $table->id('supplier_id');
16             $table->id('supplier_kode',10)->unique();
17             $table->id('supplier_nama',100);
18             $table->id('supplier_alamat',255);
19             $table->timestamps();
20         });
21     }
22
23     /**
24      * Reverse the migrations.
25      */
26     public function down(): void
27     {
28         Schema::dropIfExists('m_supplier');
29     }
30 };
```

```
C:\laragon\www\PWL_2024>php artisan migrate
```

```
INFO Running migrations.
```

```
2024_09_12_161120_create_m_supplier_table ..... 165ms DONE
```

Table	Action	Rows	Type	Collation	Size	Overhead
<input type="checkbox"/> failed_jobs	Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_unicode_ci	16.0 KIB	-
<input type="checkbox"/> migrations	Browse Structure Search Insert Empty Drop	7	InnoDB	utf8mb4_unicode_ci	16.0 KIB	-
<input type="checkbox"/> m_kategori	Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_unicode_ci	16.0 KIB	-
<input type="checkbox"/> m_level	Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_unicode_ci	16.0 KIB	-
<input type="checkbox"/> m_supplier	Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_unicode_ci	16.0 KIB	-
<input type="checkbox"/> password_reset_tokens	Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_unicode_ci	16.0 KIB	-
<input type="checkbox"/> personal_access_tokens	Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_unicode_ci	16.0 KIB	-
<input type="checkbox"/> users	Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_unicode_ci	16.0 KIB	-
8 tables	Sum	7	InnoDB	utf8mb4_0900_ai_ci	128.0 KIB	0 B

Praktikum Ke-2.2

Langkah	Jawaban/Deskripsi
1.	Buka terminal VSCode kalian, dan buat file migrasi untuk table m_user <pre>C:\laragon\www\PWL_2024>php artisan make:migration create_m_user_table --table=m_user</pre> <pre>INFO Migration [C:\laragon\www\PWL_2024\database\Migrations\2024_09_12_162526_create_m_user_table.php] created successfully.</pre>



2.	<p>Buka file migrasi untuk table m_user, dan modifikasi seperti berikut</p> <pre>2024_09_12_162526_create_m_user_table.php database > migrations > 2024_09_12_162526_create_m_user_table.php 1 k?php 2 3 use Illuminate\Database\Migrations\Migration; 4 use Illuminate\Database\Schema\Blueprint; 5 use Illuminate\Support\Facades\Schema; 6 7 return new class extends Migration 8 { 9 /** 10 * Run the migrations. 11 */ 12 public function up(): void 13 { 14 Schema::table('m_user', function (Blueprint \$table) { 15 // 16 }); 17 } 18 19 /** 20 * Reverse the migrations. 21 */ 22 public function down(): void 23 { 24 Schema::table('m_user', function (Blueprint \$table) { 25 // 26 }); 27 } 28 }</pre>
3.	<p>Simpan kode program Langkah 2, dan jalankan perintah php artisan migrate. Amati apa yang terjadi pada database.</p> <pre>C:\laragon\www\PWL_2024>php artisan migrate INFO Running migrations. 2024_09_12_162526_create_m_user_table 5ms DONE</pre> <p>Penjelasan :</p> <p>Pada phpmyadmin tabel m_user tidak muncul dikarenakan Schema::table untuk Tabel m_user yang tidak ada (belum dibuat).</p>
4.	<p>Membuat tabel :</p> <ul style="list-style-type: none">m_barang <pre>C:\laragon\www\PWL_2024>php artisan make:migration create_m_barang_table - -table=m_barang INFO Migration [C:\laragon\www\PWL_2024\database\Migrations\2024_09_12_164559_create_m_barang_table.php] created successfully.</pre>



```
2024_09_12_164559_create_m_barang_table.php U x
database > migrations > 2024_09_12_164559_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) {
15             $table->id('barang_id');
16             $table->unsignedBigInteger('kategori_id')->index(); // Indeks
17             $table->string('barang_kode', 10)->unique(); // Unique untuk
18             $table->string('barang_nama', 100);
19             $table->integer('harga_beli'); // Menggunakan tipe data int
20             $table->integer('harga_jual'); // Menggunakan tipe data int
21             $table->timestamps();
22         });
23     }
24
25     /**
26      * Reverse the migrations.
27      */
28     public function down(): void
29     {
30         Schema::table('m_barang', function (Blueprint $table) {
31             //
32         });
33     }
34 };
```

```
C:\laragon\www\PWL_2024>php artisan migrate
INFO Running migrations.
2024_09_12_164559_create_m_barang_table ..... 211ms DONE
```

- t_penjualan

```
C:\laragon\www\PWL_2024>php artisan make:migration create_t_penjualan_table --table=t_penjualan
INFO Migration [C:\laragon\www\PWL_2024\database\migrations\2024_09_12_171201_create_t_penjualan_table.php] created successfully.
```



```
2024_09_12_171201_create_t_penjualan_table.php U X
database > migrations > 2024_09_12_171201_create_t_penjualan_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('t_penjualan', function (Blueprint $table) {
15             $table->id('penjualan_id');
16             $table->unsignedBigInteger('user_id')->index(); // Indexing
17             $table->string('pembeli', 50); // Menghapus unique jika t
18             $table->string('penjualan_kode', 20);
19             $table->dateTime('penjualan_tanggal'); // Menggunakan tipe
20             $table->timestamps();
21         });
22     }
23
24     /**
25      * Reverse the migrations.
26      */
27     public function down(): void
28     {
29         Schema::table('t_penjualan', function (Blueprint $table) {
30             //
31         });
32     }
33 };
```

```
C:\laragon\www\PWL_2024>php artisan migrate
```

```
INFO Running migrations.
```

```
2024_09_12_171201_create_t_penjualan_table ..... 174ms DONE
```

- t_stok

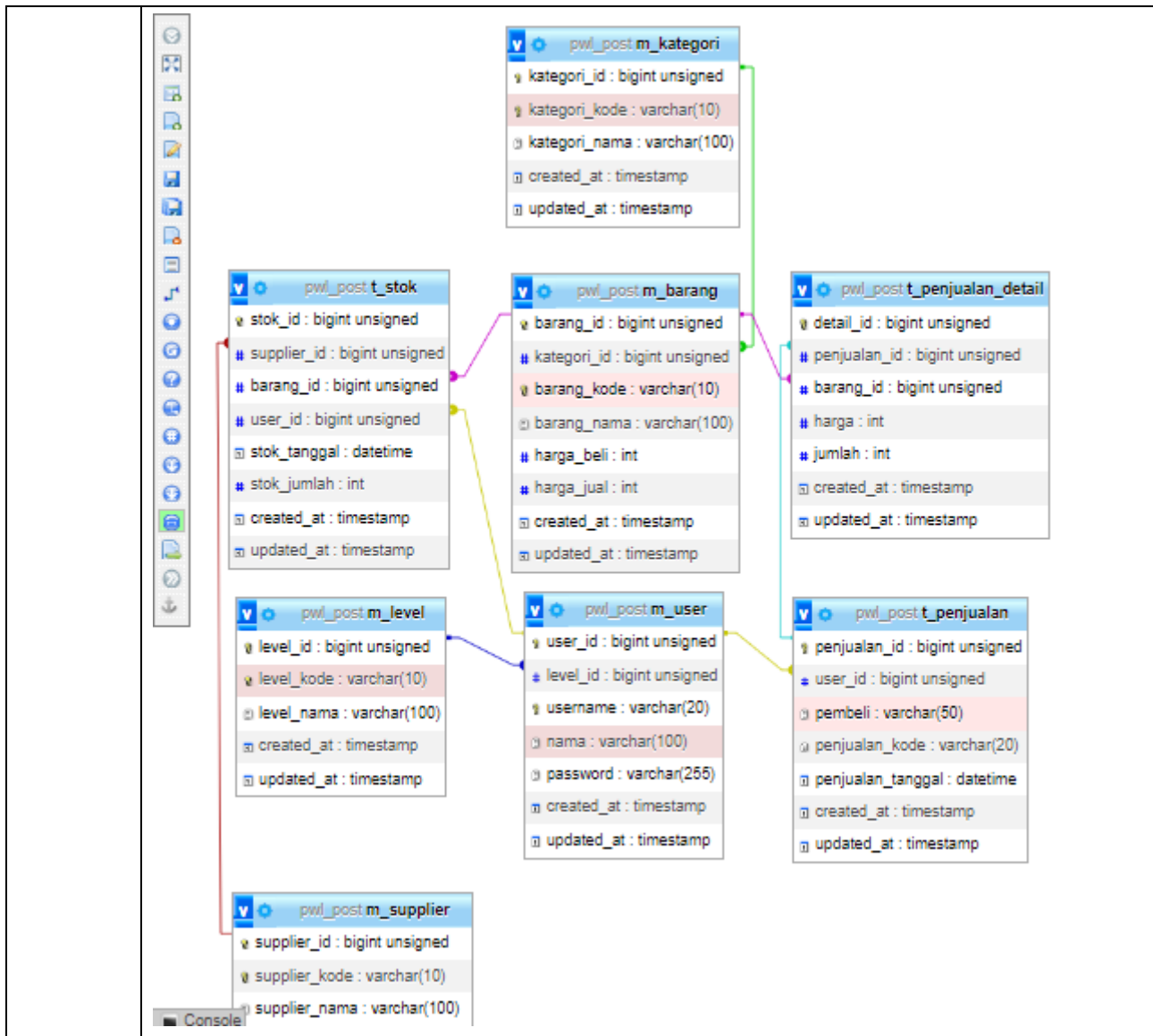
```
C:\laragon\www\PWL_2024>php artisan make:migration create_t_stok_table --table=t_s
tok
```

```
INFO Migration [C:\laragon\www\PWL_2024\database\Migrations\2024_09_12_175830_
create_t_stok_table.php] created successfully.
```

```
database > migrations > 2024_09_12_175830_create_t_stok_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('t_stok', function (Blueprint $table) {
15             $table->id('stok_id'); // Primary key
16             $table->unsignedBigInteger('supplier_id')->index(); // Foreign key
17             $table->unsignedBigInteger('barang_id')->index(); // Foreign key
18             $table->unsignedBigInteger('user_id')->index(); // Foreign key
19             $table->dateTime('stok_tanggal'); // Tanggal stok dalam format dat
20             $table->integer('stok_jumlah'); // Jumlah stok dengan tipe integ
21             $table->timestamps(); // Created_at dan Updated_at
22         });
23     }
24
25     /**
26      * Reverse the migrations.
27      */
28     public function down(): void
29     {
30         Schema::table('t_stok', function (Blueprint $table) {
31             //
32         });
33     }
34 };
```




	<pre>C:\laragon\www\PMI_2024>php artisan migrate</pre> <pre>INFO Running migrations.</pre> <pre>2024_09_12_175830_create_t_stok_table 162ms DONE</pre> <ul style="list-style-type: none">• <code>t_penjualan_detail</code> <pre>C:\laragon\www\PMI_2024>php artisan make:migration create_t_penjualan_detail_table --table=t_penjualan_detail</pre> <pre>INFO Migration [C:\laragon\www\PMI_2024\database\migrations\2024_09_12_180610_create_t_penjualan_detail_table.php] created successfully.</pre> <pre>database > migrations > 2024_09_12_180610_create_t_penjualan_detail_table.php</pre> <pre>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) { 15 \$table->id('detail_id'); // Primary key 16 \$table->unsignedBigInteger('penjualan_id')->index(); // Foreign key ke tabel penjualan 17 \$table->unsignedBigInteger('barang_id')->index(); // Foreign key ke tabel barang 18 \$table->integer('harga'); // Harga dengan tipe integer 19 \$table->integer('jumlah'); // Jumlah barang yang dibeli 20 \$table->timestamps(); // Created_at dan Updated_at 21 }); 22 23 // Menambahkan foreign key constraints (Optional) 24 Schema::table('t_penjualan_detail', function (Blueprint \$table) { 25 \$table->foreign('penjualan_id')->references('penjualan_id')->on('t_penjualan')->onDelete('cascade'); 26 \$table->foreign('barang_id')->references('barang_id')->on('m_barang')->onDelete('cascade'); 27 }); 28 } 29 30 /** 31 * Reverse the migrations. 32 */ 33 public function down(): void 34 { 35 Schema::dropIfExists('t_penjualan_detail'); 36 } 37 };</pre> <pre>C:\laragon\www\PMI_2024>php artisan migrate</pre> <pre>INFO Running migrations.</pre> <pre>2024_09_12_180610_create_t_penjualan_detail_table 360ms DONE</pre>
5.	Jika semua file migrasi sudah di buat dan dijalankan maka bisa kita lihat tampilan designer pada phpMyAdmin seperti berikut



Praktikum Ke-3

Langkah	Jawaban/Deskripsi
1.	Kita akan membuat file seeder untuk table m_level dengan mengetikkan perintah <pre>C:\laragon\www\PWL_2024>php artisan make:seeder levelSeeder</pre> <p>INFO Seeder [C:\laragon\www\PWL_2024\database\seeders\levelSeeder.php] created successfully.</p>
2.	Selanjutnya, untuk memasukkan data awal, kita modifikasi file tersebut di dalam function run()



3.	Selanjutnya, kita jalankan file seeder untuk table m_level pada terminal
4.	Ketika seeder berhasil dijalankan maka akan tampil data pada table m_level
5.	Sekarang kita buat file seeder untuk table m_user yang me-refer ke table m_level
6.	Modifikasi file class UserSeeder seperti berikut



	<pre>levelSeeder.php U UserSeeder.php U X database > seeders > UserSeeder.php 1 <?php 2 3 namespace Database\Seeders; 4 5 use Illuminate\Database\Console\Seeds\WithoutModelEvents; 6 use Illuminate\Database\Seeder; 7 use Illuminate\Support\Facades\DB; 8 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 \$data = [18 [19 'user_id' => 1, 20 'level_id' => 1, 21 'username' => 'admin', 22 'nama' => 'Administrator', 23 'password' => Hash::make('12345'), 24], 25 [26 'user_id' => 2, 27 'level_id' => 2, 28 'username' => 'manager', 29 'nama' => 'Manager', 30 'password' => Hash::make('12345'), 31], 32 [33 'user_id' => 3, 34 'level_id' => 3, 35 'username' => 'staff', 36 'nama' => 'Staff/Kasir', 37 'password' => Hash::make('12345'), 38], 39]; 40 41 DB::table('m_user')->insert(\$data); 42 } 43 }</pre>
7.	<p>Jalankan perintah untuk mengeksekusi class UserSeeder</p> <pre>C:\laragon\www\PWL_2024>php artisan make:seeder UserSeeder INFO Seeder [C:\laragon\www\PWL_2024\database\seeders\UserSeeder.php] created successfully. C:\laragon\www\PWL_2024>php artisan db:seed --class=UserSeeder INFO Seeding database.</pre>
8.	<p>Perhatikan hasil seeder pada table m_user</p>



Server: localhost:3306 Database: pwl_post Table: m_user

Showing rows 0 - 2 (3 total, Query took 0.0040 seconds.)

SELECT * FROM "m_user"

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

Extra options

	user_id	level_id	username	nama	password	created_at	updated_at
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	1	1	admin	Administrator	\$2y\$12\$HPAs39FxpZGPk.cARojm00Pq7DYM6VOH7IP24B8ajZ...	NULL	NULL
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	2	2	manager	Manager	\$2y\$12\$A_yU42GPE1WTO5BbLWSeXHBelJmRMCLZYS3M9NLF...	NULL	NULL
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	3	3	staff	Staff/Kasir	\$2y\$12\$y1lcpVe.Qn//QkGTCCOUJe1ozJ3JCRj7Y57tNEMZEd...	NULL	NULL

Check all With selected: Edit Copy Delete Export

9. Ok, data seeder berhasil di masukkan ke database.

10. Sekarang coba kalian masukkan data seeder untuk table yang lain, dengan ketentuan seperti berikut

- m_kategori

```
C:\laragon\www\pwl_2024>php artisan make:seeder KategoriSeeder

INFO Seeder [C:\laragon\www\pwl_2024\database\seeders\KategoriSeeder.php] created successfully.
```

```
KategoriSeeder.php U x
base > seeders > KategoriSeeder.php
<?php

namespace Database\Seeders;

use Illuminate\Database\Seeder;
use Illuminate\Support\Facades\DB;

class KategoriSeeder extends Seeder
{
    /**
     * Run the database seeds.
     */
    public function run(): void
    {
        $data = [
            ['kategori_id' => 1, 'kategori_kode' => 'ELEC', 'kategori_nama' => 'Elektronik'],
            ['kategori_id' => 2, 'kategori_kode' => 'FURN', 'kategori_nama' => 'Furniture'],
            ['kategori_id' => 3, 'kategori_kode' => 'FOOD', 'kategori_nama' => 'Makanan'],
            ['kategori_id' => 4, 'kategori_kode' => 'CLOT', 'kategori_nama' => 'Pakaian'],
            ['kategori_id' => 5, 'kategori_kode' => 'BOOK', 'kategori_nama' => 'Buku'],
        ];

        DB::table('m_kategori')->insert($data);
    }
}
```

```
C:\laragon\www\pwl_2024>php artisan db:seed --class=KategoriSeeder

INFO Seeding database.
```

- m_supplier

```
C:\laragon\www\pwl_2024>php artisan make:seeder SupplierSeeder

INFO Seeder [C:\laragon\www\pwl_2024\database\seeders\SupplierSeeder.php] created successfully.
```



```
SupplierSeeder.php U X
database > seeders > SupplierSeeder.php
3 namespace Database\Seeders;
4
5 use Illuminate\Database\Seeder;
6 use Illuminate\Support\Facades\DB;
7
8 class SupplierSeeder extends Seeder
9 {
10     /**
11      * Run the database seeds.
12      */
13     public function run(): void
14     {
15         $data = [
16             [
17                 'supplier_id' => 1, // Menentukan supplier_id
18                 'supplier_kode' => 'SUP001',
19                 'supplier_nama' => 'Supplier A',
20                 'supplier_alamat' => 'Jl. Merdeka No. 1',
21             ],
22             [
23                 'supplier_id' => 2, // Menentukan supplier_id
24                 'supplier_kode' => 'SUP002',
25                 'supplier_nama' => 'Supplier B',
26                 'supplier_alamat' => 'Jl. Proklamasi No. 2',
27             ],
28             [
29                 'supplier_id' => 3, // Menentukan supplier_id
30                 'supplier_kode' => 'SUP003',
31                 'supplier_nama' => 'Supplier C',
32                 'supplier_alamat' => 'Jl. Pahlawan No. 3',
33             ]
34         ];
35
36         DB::table('m_supplier')->insert($data);
37     }
38 }
```

```
C:\laragon\www\PWL_POS>php artisan db:seed --class=SupplierSeeder
```

```
INFO Seeding database.
```

- m_barang

```
C:\laragon\www\PWL_POS>php artisan make:seeder BarangSeeder
```

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



```
BarangSeeder.php U X
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     /**
11      * Run the database seeds.
12      */
13     public function run(): void
14     {
15         $data = [];
16         $kategori = [1, 2, 3, 4, 5]; // Kategori IDs
17
18         for ($i = 1; $i <= 15; $i++) {
19             $data[] = [
20                 'barang_id' => $i,
21                 'kategori_id' => $kategori[array_rand($kategori)],
22                 'barang_kode' => 'BRG' . $i,
23                 'barang_nama' => 'Barang ' . $i,
24                 'harga_beli' => rand(10000, 50000),
25                 'harga_jual' => rand(60000, 100000),
26             ];
27         }
28
29         DB::table('m_barang')->insert($data);
30     }
31 }
```

```
C:\laragon\www\PWL_POS>php artisan db:seed --class=BarangSeeder
INFO Seeding database.
```

- t_stok

```
C:\laragon\www\PWL_POS_2024>php artisan make:seeder StokSeeder
INFO Seeder [C:\laragon\www\PWL_POS_2024\database\seeders\StokSeeder.php] created successfully.
```



```
StokSeeder.php u x
database > seeders > StokSeeder.php
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     /**
12      * Run the database seeds.
13      */
14     public function run(): void
15     {
16         $data = [];
17         $now = Carbon::now();
18
19         for ($i = 1; $i <= 15; $i++) {
20             $data[] = [
21                 'stok_id' => $i,
22                 'supplier_id' => ($i % 3) + 1,
23                 'barang_id' => $i,
24                 'user_id' => 1,
25                 'stok_tanggal' => $now->subDays(rand(1, 30))->toDateTimeString(),
26                 'stok_jumlah' => rand(10, 100),
27             ];
28         }
29
30         DB::table('t_stok')->insert($data);
31     }
32 }
```

```
C:\laragon\www\PWL_POS_2024>php artisan db:seed --class=StokSeeder

INFO Seeding database.
```

- t_penjualan

```
C:\laragon\www\PWL_POS_2024>php artisan make:seeder PenjualanSeeder

INFO Seeder [C:\laragon\www\PWL_POS_2024\database\seeders\PenjualanSeeder.php] created successfully.
```




```
PenjualanSeeder.php u X
database > seeders > PenjualanSeeder.php
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     /**
12      * Run the database seeds.
13      */
14     public function run(): void
15     {
16         $data = [];
17         $now = Carbon::now();
18
19         for ($i = 1; $i <= 10; $i++) {
20             $data[] = [
21                 'penjualan_id' => $i,
22                 'user_id' => 1,
23                 'pembeli' => 'Pembeli ' . $i,
24                 'penjualan_kode' => 'PNJ' . $i,
25                 'penjualan_tanggal' => $now->subDays(rand(1, 30))->toDateTimeString()
26             ];
27         }
28
29         DB::table('t_penjualan')->insert($data);
30     }
31 }
```

```
C:\laragon\www\PWL_POS_2024>php artisan db:seed --class=PenjualanSeeder
INFO Seeding database.
```

- t_penjualan_detail

```
C:\laragon\www\PWL_POS_2024>php artisan make:seeder Penjualan_DetailSeeder
INFO Seeder [C:\laragon\www\PWL_POS_2024\database\seeders\Penjualan_DetailSeeder.php] created successfully.
```

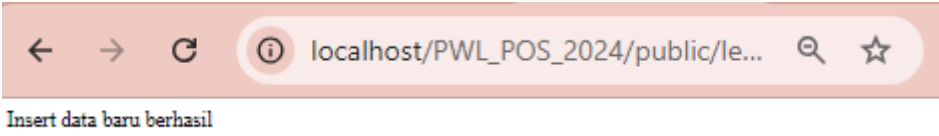


	<pre>Penjualan_DetailSeeder.php U X database > seeders > Penjualan_DetailSeeder.php 1 <?php 2 3 namespace Database\Seeders; 4 5 use Illuminate\Database\Seeder; 6 use Illuminate\Support\Facades\DB; 7 8 class Penjualan_DetailSeeder extends Seeder 9 { 10 /** 11 * Run the database seeds. 12 */ 13 public function run(): void 14 { 15 \$data = []; 16 17 for (\$i = 1; \$i <= 10; \$i++) { 18 for (\$j = 1; \$j <= 3; \$j++) { 19 \$data[] = [20 'detail_id' => (\$i - 1) * 3 + \$j, 21 'penjualan_id' => \$i, 22 'barang_id' => rand(1, 15), 23 'harga' => rand(60000, 100000), 24 'jumlah' => rand(1, 10), 25]; 26 } 27 } 28 29 DB::table('t_penjualan_detail')->insert(\$data); 30 } 31 }</pre> <pre>C:\laragon\www\PWL_POS_2024>php artisan db:seed --class=Penjualan_DetailSeeder INFO Seeding database.</pre>	
--	-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	--

Praktikum Ke-4

Langkah	Jawaban/Deskripsi
1.	Kita buat controller dahulu untuk mengelola data pada table m_level <pre>C:\laragon\www\PWL_POS_2024>php artisan make:controller levelController</pre> <pre>INFO Controller [C:\laragon\www\PWL_POS_2024\app\Http\Controllers\levelController.php] created successfully.</pre>
2.	Kita modifikasi dulu untuk routing-nya, ada di PWL_POS/routes/web.php



	<pre>routes > web.php 1 <?php 2 3 use App\Http\Controllers\LevelController; 4 use Illuminate\Support\Facades\Route; 5 6 /* 7 ----- 8 Web Routes 9 ----- 10 11 Here is where you can register web routes for your application. These 12 routes are loaded by the RouteServiceProvider and all of them will 13 be assigned to the "web" middleware group. Make something great! 14 15 */ 16 17 Route::get('/', function () { 18 return view('welcome'); 19 }); 20 21 Route::get('/level', [LevelController::class, 'index']);</pre>
3.	<p>Selanjutnya, kita modifikasi file LevelController untuk menambahkan 1 data ke table m_level</p> <pre>levelController.php U X 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; // Pastikan untuk mengimpor DB 7 8 class LevelController extends Controller 9 { 10 public function index() 11 { 12 13 DB::insert("insert into m_level (level_kode, level_nama, created_at) values (?, ?, ?)", ['CUS', 'Pelanggan', now()]); 14 15 return "Insert data baru berhasil"; 16 } 17 }</pre>
4.	<p>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</p> 



Server: localhost:3306 > Database: pwl_pos > Table: m_level

Showing rows 0 - 3 (4 total, Query took 0.0038 seconds.)

SELECT * FROM `m_level`

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

Show all Number of rows: 25 Filter rows: Search this table Sort by key: Non

Extra options

	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	2024-09-13 02:45:20	NULL

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

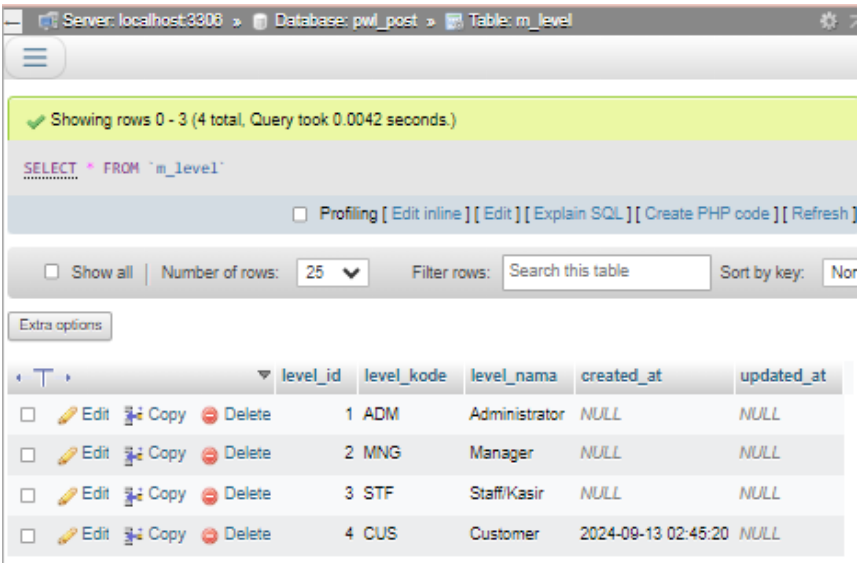
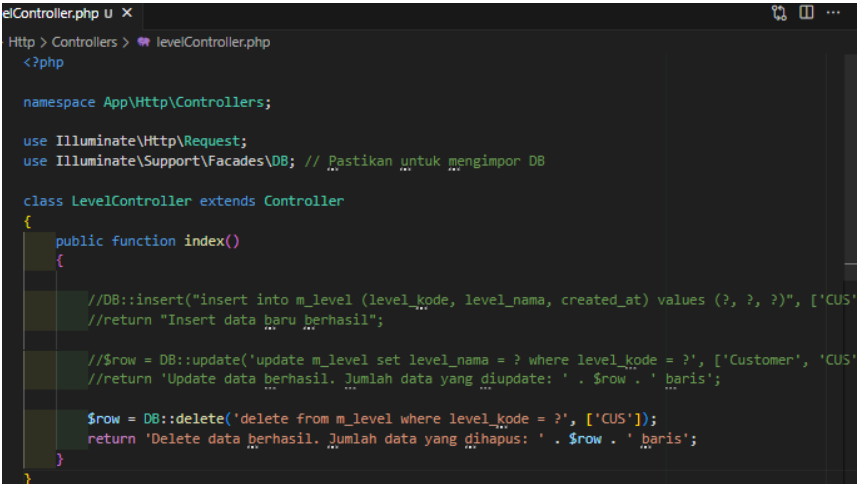
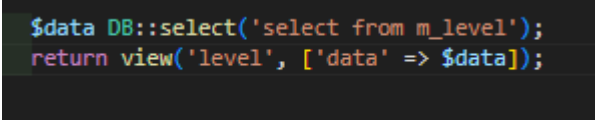
```
levelController.php U x
pp > Http > Controllers > levelController.php
1 <?php
2
3 namespace App\Http\Controllers;
4
5 use Illuminate\Http\Request;
6 use Illuminate\Support\Facades\DB; // Pastikan untuk mengimpor DB
7
8 class LevelController extends Controller
9 {
10     public function index()
11     {
12
13         //DB::insert("insert into m_level (level_kode, level_nama, created_at) values (?, ?, ?)", ['CUS
14         //return "Insert data baru berhasil";
15
16         $row = DB::update('update m_level set level_nama = ? where level_kode = ?', ['Customer', 'CUS'])
17         return 'Update data berhasil. Jumlah data yang diupdate: ' . $row . ' baris';
18     }
19 }
```

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

localhost/PWL_POS_2024/public/le...

Update data berhasil. Jumlah data yang diupdate: 1 baris

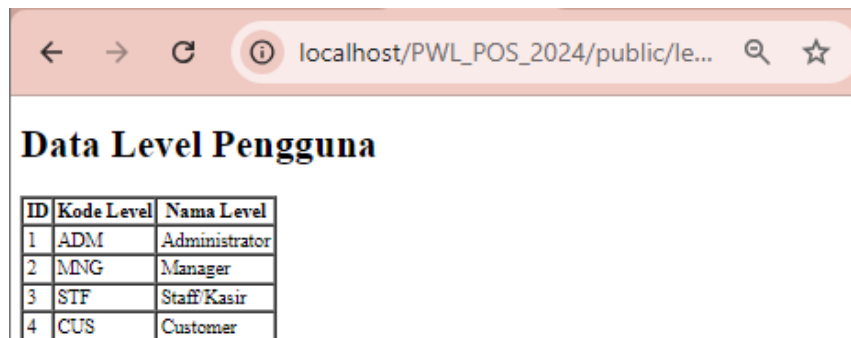


	
7.	<p>Kita coba modifikasi lagi file LevelController untuk melakukan proses hapus data</p>  <pre>elController.php u X Http > Controllers > levelController.php <?php namespace App\Http\Controllers; use Illuminate\Http\Request; use Illuminate\Support\Facades\DB; // Pastikan untuk mengimpor DB class LevelController extends Controller { public function index() { //DB::insert("insert into m_level (level_kode, level_nama, created_at) values (?, ?, ?)", ['CUS' //return "Insert data baru berhasil"; //\$row = DB::update('update m_level set level_nama = ? where level_kode = ?', ['Customer', 'CUS' //return 'Update data berhasil. Jumlah data yang diupdate: ' . \$row . ' baris'; \$row = DB::delete('delete from m_level where level_kode = ?', ['CUS']); return 'Delete data berhasil. Jumlah data yang dihapus: ' . \$row . ' baris'; } }</pre>
8.	<p>Method terakhir yang kita coba adalah untuk menampilkan data yang ada di table m_level. Kita modifikasi file LevelController seperti berikut</p>  <pre>\$data DB::select('select from m_level'); return view('level', ['data' => \$data]);</pre>
9.	<p>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</p>



```
level.blade.php U X
resources > views > level.blade.php
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>Data Level Pengguna</title>
7 </head>
8 <body>
9   <h1>Data Level Pengguna</h1>
10  <table border="1" cellpadding="2" cellspacing="0">
11    <thead>
12      <tr>
13        <th>ID</th>
14        <th>Kode Level</th>
15        <th>Nama Level</th>
16      </tr>
17    </thead>
18    <tbody>
19      @foreach ($data as $d)
20        <tr>
21          <td>{{ $d->level_id }}</td>
22          <td>{{ $d->level_kode }}</td>
23          <td>{{ $d->level_nama }}</td>
24        </tr>
25      @endforeach
26    </tbody>
27  </table>
28 </body>
29 </html>
```

10. Silahkan dicoba pada browser dan amati apa yang terjadi



Setelah mengikuti langkah-langkah di atas, akan terlihat halaman web yang menampilkan tabel data dari tabel m_level. Tabel harus menampilkan kolom ID, Kode Level, dan Nama Level sesuai dengan data yang ada di database

Praktikum Ke-5

Langkah	Jawaban/Deskripsi
---------	-------------------



1.	<p>Kita buat controller dahulu untuk mengelola data pada table m_kategori</p> <pre>C:\laragon\www\PWL_POS_2024>php artisan make:controller KategoriController</pre> <p>INFO Controller [C:\laragon\www\PWL_POS_2024\app\Http\Controllers\KategoriController.php] created successfully.</p>
2.	<p>Kita modifikasi dulu untuk routing-nya, ada di PWL_POS/routes/web.php</p> <pre>web.php M X routes > web.php 14 15 */ 16 17 Route::get('/', function () { 18 return view('welcome'); 19 }); 20 21 Route::get('/level', [LevelController::class, 'index']); 22 Route::get('/kategori', [KategoriController::class, 'index']);</pre>
3.	<p>Selanjutnya, kita modifikasi file KategoriController untuk menambahkan 1 data ke table m_kategori</p> <pre>app > Http > Controllers > KategoriController.php 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' => 'SPC', 14 'kategori_nama' => 'Spice/Bumbu', 15 'created_at' => now() 16]; 17 DB::table('m_kategori')->insert(\$data); 18 return 'Insert data baru berhasil'; 19 } 20 }</pre>
4.	<p>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</p>



5.	<p>Selanjutnya, kita modifikasi lagi file KategoriController untuk meng-update data di table m_kategori seperti berikut</p> <pre>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' => 'SPC', 14 'kategori_nama' => 'Spice/Bumbu', 15 'created_at' => now() 16]; 17 DB::table('m_kategori')->insert(\$data); 18 return 'Insert data baru berhasil'; 19 } 20 21 \$row = DB::table('m_kategori')->where('kategori_kode', 'SPC')->update(['kategori_nama' => 'Seasoning']); 22 return 'Update data berhasil. Jumlah data yang diupdate: '.\$row.' baris'; 23 } 24 }</pre>
6.	<p>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</p>
7.	<p>Kita coba modifikasi lagi file KategoriController untuk melakukan proses hapus data</p>



```
app > Http > Controllers > KategoriController.php
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         /*$data = [
13             'kategori_kode' => 'SPC',
14             'kategori_nama' => 'Spice/Bumbu',
15             'created_at' => now()
16         ];
17         DB::table('m_kategori')->insert($data);
18         return 'Insert data baru berhasil';*/
19
20         // $row = DB::table('m_kategori')->where('kategori_kode', 'SPC')->update(['kategori_nama' => 'Spice/Bumbu']);
21         // return 'Update data berhasil. Jumlah data yang diupdate: '.$row.' baris';
22
23         $row = DB::table('m_kategori')->where('kategori_kode', 'SPC')->delete();
24         return 'Delete data berhasil. Jumlah data yang dihapus: '.$row.' baris';
25     }
26 }
```

←

→

↺

localhost/PWL_POS_2024/public/ka...

🔍

☆

Delete data berhasil. Jumlah data yang dihapus: 1 baris

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

⌵

Baris sudah terhapus.

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

← → ↺ ⓘ localhost/PWL_P

Data Kategori

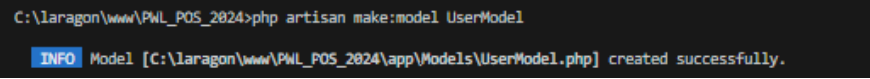

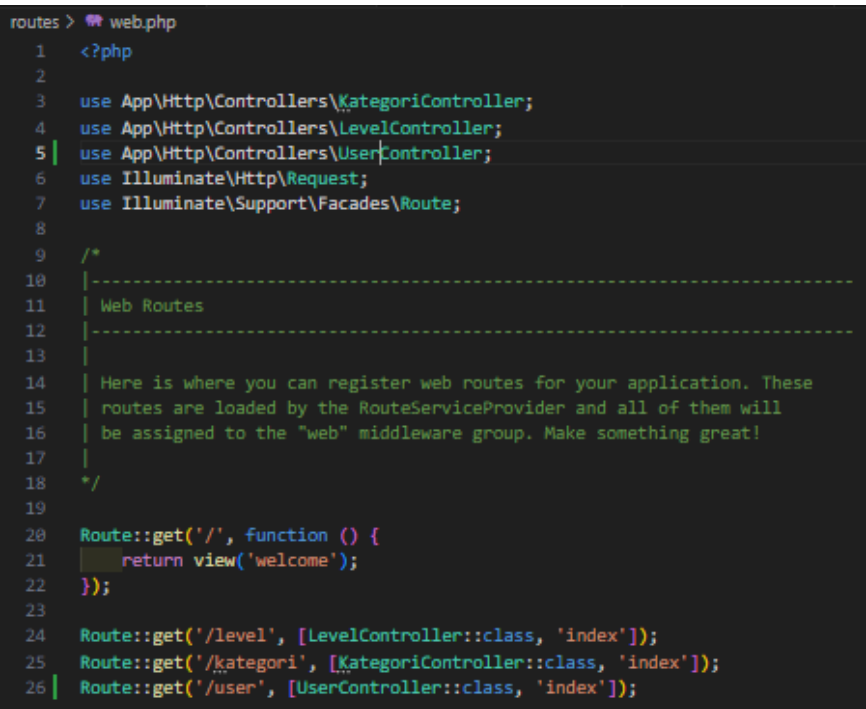
ID	Kode Kategori	Nama Kategori	Waktu Dibuat
1	ELEC	Elektronik	
2	FURN	Furniture	
3	FOOD	Makanan	
4	CLOT	Pakaian	
5	BOOK	Buku	



	<pre>app > Http > Controllers > KategoriController.php 5 use Illuminate\Http\Request; 6 use Illuminate\Support\Facades\DB; 7 8 class KategoriController extends Controller 9 { 10 public function index(){ 11 /*\$data = [12 'kategori_kode' => 'SPC', 13 'kategori_nama' => 'Spice/Bumbu', 14 'created_at' => now() 15]; 16 DB::table('m_kategori')->insert(\$data); 17 return 'Insert data baru berhasil';*/ 18 19 // \$row = DB::table('m_kategori')->where('kategori_kode', 'SPC')->update(['kategori_nama' 20 // return 'Update data berhasil. Jumlah data yang diupdate: '.\$row.' baris'; 21 22 // \$row = DB::table('m_kategori')->where('kategori_kode', 'SPC')->delete(); 23 // return 'Delete data berhasil. Jumlah data yang dihapus: '.\$row.' baris'; 24 25 \$data = DB::table('m_kategori')->get(); 26 return view('kategori', ['data'=>\$data]); 27 } 28 } 29 }</pre>
9.	<p>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</p> <pre>resources > views > kategori.blade.php 1 <!DOCTYPE html> 2 <html lang="en"> 3 <head> 4 <meta charset="UTF-8"> 5 <meta name="viewport" content="width=device-width, initial-scale=1.0"> 6 <title>Data Kategori Barang</title> 7 </head> 8 <body> 9 <h1>Data Kategori Barang</h1> 10 <table border="1" cellpadding="2" cellspacing="0"> 11 <thead> 12 <tr> 13 <th>ID</th> 14 <th>Kode Kategori</th> 15 <th>Nama Kategori</th> 16 </tr> 17 </thead> 18 <tbody> 19 @foreach (\$data as \$d) 20 <tr> 21 <td>{{ \$d->kategori_id }}</td> 22 <td>{{ \$d->kategori_kode }}</td> 23 <td>{{ \$d->kategori_nama }}</td> 24 </tr> 25 @endforeach 26 </tbody> 27 </table> 28 </body> 29 </html></pre>
10.	<p>Silahkan dicoba pada browser dan amati apa yang terjadi.</p> <p>Jawaban :</p> <p>Pada browser muncul table dan isi sesuai dengan yang ada pada database table m_kategori</p>



Soal Praktikum 6

Langkah	Jawaban/Deskripsi
1.	Kita buat file model untuk tabel m_user dengan mengetikkan perintah <pre>C:\laragon\www\PWL_P05_2024>php artisan make:model UserModel</pre> 
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 
4.	Kita modifikasi route web.php untuk mencoba routing ke controller UserController 
5.	Sekarang, kita buat file controller UserController dan memodifikasinya seperti berikut



	<pre>C:\laragon\www\pwl_P05_2024>php artisan make:controller UserController [INFO] Controller [C:\laragon\www\pwl_P05_2024\app\Http\Controllers\UserController.php] created successfully. app > Http > Controllers > UserUserController.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 \$users = UserModel::all(); // Ambil semua data dari tabel 'users' 14 15 // Kirim data ke view 'user' 16 return view('user', ['data' => \$users]); 17 } 18 </pre>
6.	<p>Kemudian kita buat view user.blade.php</p> <pre>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 <thead> 10 <tr> 11 <th>ID</th> 12 <th>Username</th> 13 <th>Nama</th> 14 <th>ID Level Pengguna</th> 15 </tr> 16 </thead> 17 <tbody> 18 @foreach (\$data as \$d) 19 <tr> 20 <td>{{ \$d->user_id }}</td> 21 <td>{{ \$d->username }}</td> 22 <td>{{ \$d->nama }}</td> 23 <td>{{ \$d->level_id }}</td> 24 </tr> 25 @endforeach 26 </tbody> 27 </table> 28 </body> 29 </html></pre>
7.	<p>Jalankan di browser, catat dan laporkan apa yang terjadi</p>



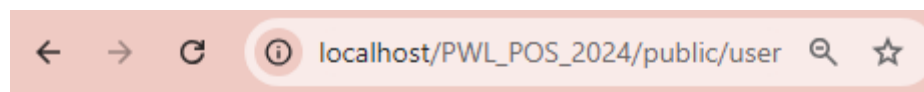
	<div>← → ↺ ⓘ localhost/PWL_POS_2024/public/user</div> <h3>Data User</h3> <table><tr><th>ID</th><th>Username</th><th>Nama</th><th>ID Level Pengguna</th></tr><tr><td>1</td><td>admin</td><td>Administrator</td><td>1</td></tr><tr><td>2</td><td>manager</td><td>Manager</td><td>2</td></tr><tr><td>3</td><td>staff</td><td>Staff Kasir</td><td>3</td></tr></table> <p>Penjelasan :</p> <p>browser akan menampilkan halaman dengan daftar user yang diambil dari tabel m_user di database.</p>	ID	Username	Nama	ID Level Pengguna	1	admin	Administrator	1	2	manager	Manager	2	3	staff	Staff Kasir	3
ID	Username	Nama	ID Level Pengguna														
1	admin	Administrator	1														
2	manager	Manager	2														
3	staff	Staff Kasir	3														
8.	<p>Setelah itu, kita modifikasi lagi file UserController</p> <pre>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 21 // Tambahkan data ke tabel m_user 22 UserModel::insert(\$data); 23 24 \$user = UserModel::all(); // ambil semua data dari tabel m_user 25 26 // Kirimkan data ke view 'user' 27 return view('user', ['data' => \$user]); 28 } 29 }</pre>																
9.	<p>Jalankan di browser, amati dan laporkan apa yang terjadi</p> <p>Penjelasan :</p>																



Data pengguna baru (username 'customer-1', nama 'Pelanggan', password hash, level_id 4) ditambahkan ke tabel m_user. Semua data dari tabel m_user diambil menggunakan UserModel::all(). Kemudian data tersebut ditampilkan di view user.blade.php dalam bentuk tabel, termasuk data pengguna baru yang telah ditambahkan.

10. Kita modifikasi lagi file UserController menjadi 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  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         $data = [
15             'nama' => 'Pelanggan Pertama',
16         ];
17
18         // Update data user dengan username 'customer-1'
19         UserModel::where('username', 'customer-1')->update($data);
20
21         // Ambil semua data dari tabel m_user
22         $user = UserModel::all();
23
24         // Kirim data ke view 'user'
25         return view('user', ['data' => $user]);
26     }
27 }
```



Data User

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

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

Penjelasan :



	<ul style="list-style-type: none">• Data user dengan username = 'customer-1' akan di-update, mengubah kolom nama menjadi 'Pelanggan Pertama'.• Semua data dari tabel m_user akan diambil dan ditampilkan di view user.blade.php.• Di browser, akan terlihat tabel dengan data user, termasuk perubahan pada customer-1.
--	---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Soal

Soal	Jawaban
1.	Fungsi APP_KEY di file .env Laravel: APP_KEY digunakan untuk mengenkripsi data, seperti session cookies dan data sensitif lainnya, guna memastikan keamanan aplikasi Laravel. Ini adalah kunci aplikasi yang digunakan untuk proses hashing.
2.	Untuk men-generate nilai APP_KEY, kita dapat menjalankan perintah di terminal php artisan key:generate. Perintah ini akan menghasilkan kunci yang secara otomatis diisi ke dalam file .env.
3.	Jumlah file migrasi default di Laravel: Secara default, Laravel memiliki tiga file migrasi: <ul style="list-style-type: none">• create_users_table untuk membuat tabel pengguna (users).• create_password_resets_table untuk menyimpan token reset password.• create_failed_jobs_table untuk menyimpan informasi tentang pekerjaan yang gagal pada antrian
4.	Tujuan \$table->timestamps(); Fungsi ini otomatis menambahkan dua kolom ke tabel: created_at dan updated_at. Keduanya menyimpan informasi waktu kapan record dibuat dan diperbarui.



5.	Tipe data dari fungsi \$table->id();: Fungsi ini menghasilkan kolom bertipe data BIGINT dengan sifat auto-increment, yang digunakan sebagai primary key.
6.	Perbedaan antara \$table->id(); dan \$table->id('level_id'); <ul style="list-style-type: none">• \$table->id(); secara default membuat kolom id sebagai primary key.• \$table->id('level_id'); membuat kolom level_id sebagai primary key. Ini memberi nama khusus untuk kolom primary key.
7.	Fungsi ->unique(): Fungsi ini memastikan bahwa nilai dalam kolom tertentu harus unik, tidak boleh ada nilai yang sama di kolom tersebut.
8.	Karena: <ul style="list-style-type: none">• Di tabel m_user, level_id menggunakan unsignedBigInteger karena kolom ini adalah foreign key yang mengacu ke level_id di tabel lain.• Di tabel m_level, level_id menggunakan \$table->id('level_id') karena ini adalah primary key di tabel tersebut, jadi harus auto-increment
9.	Tujuan Class Hash dan maksud dari Hash::make('1234'); <ul style="list-style-type: none">• Class Hash digunakan untuk mengenkripsi atau melakukan hashing data, khususnya password.• Hash::make('1234') menghasilkan hash dari string '1234', yang digunakan untuk menyimpan password dalam bentuk terenkripsi.
10.	Kegunaan tanda tanya (?) pada query builder: <p>Tanda tanya ? digunakan sebagai placeholder untuk binding parameter. Nilai yang akan diinputkan ke query dimasukkan setelahnya, untuk mencegah SQL injection.</p>
11.	Tujuan dari protected \$table = 'm_user'; dan protected \$primaryKey = 'user_id';:



	<ul style="list-style-type: none">• <code>protected \$table = 'm_user'</code>; digunakan untuk menetapkan tabel yang terkait dengan model.• <code>protected \$primaryKey = 'user_id'</code>; digunakan untuk menetapkan primary key yang berbeda dari default (yang biasanya adalah id).
12.	<p>Kemudahan menggunakan DB Façade, Query Builder, atau Eloquent ORM dalam operasi CRUD:</p> <ul style="list-style-type: none">• Eloquent ORM paling mudah digunakan karena memiliki sintaks yang lebih sederhana dan menggunakan konsep OOP (pemrograman berorientasi objek). Eloquent juga secara otomatis menangani relasi antar tabel dan aturan-aturan di database.• Query Builder lebih fleksibel untuk membuat query yang lebih rumit, tetapi masih lebih mudah dibandingkan menulis SQL secara langsung.• DB Façade memungkinkan kita menulis query SQL asli, tapi kita perlu lebih memahami SQL dengan baik. <p>Kesimpulan: Eloquent ORM adalah pilihan termudah untuk operasi CRUD karena memberikan kemudahan dalam penulisan kode yang lebih simpel dan mudah dipahami.</p>