



## **PRAKTIKUM**

Pemrograman Web Lanjut - Minggu 4

### **DISUSUN OLEH:**

NAMA : Ferdi Riansyah Ramadhani Kusuma  
NIM : 2241720264  
JURUSAN : TEKNOLOGI INFORMASI  
KELAS/ ABSEN : INFORMATIKA - 2H

PROGAM STUDI D-IV TEKNIK INFORMATIKA

JURUSAN TEKNOLOGI INFORMASI

POLITEKNIK NEGERI MALANG

**2024**

## A. PENGATURAN DATABASE

### Praktikum 1 - pengaturan database

#### Steps

Buka aplikasi phpMyAdmin, dan buat database baru dengan nama PWL\_POS

#### Attachments



Buka file .env, dan pastikan konfigurasi APP\_KEY bernilai. Jika belum bernilai silahkan kalian generate menggunakan php artisan.

```
1 APP_NAME=Laravel
2 APP_ENV=local
3 APP_KEY=base64:NfQSE7s1QUuups0Rk0d8P4I2dL2dYV13EmgP
4 APP_DEBUG=true
5 APP_URL=http://localhost
6
7 LOG_CHANNEL=stack
8 LOG_DEPRECATIONS_CHANNEL=null
9 LOG_LEVEL=debug
10
11 DB_CONNECTION=mysql
12 DB_HOST=127.0.0.1
13 DB_PORT=3306
14 DB_DATABASE=pwl_pos
15 DB_USERNAME=root
16 DB_PASSWORD=
17
18 BROADCAST_DRIVER=log
19 CACHE_DRIVER=file
20 FILESYSTEM_DISK=local
```

Edit file .env dan sesuaikan dengan database yang telah dibuat

```
1 APP_NAME=Laravel
2 APP_ENV=local
3 APP_KEY=base64:NfQSE7s1QUuups0Rk0d8P4I2dL2dYV13EmgP
4 APP_DEBUG=true
5 APP_URL=http://localhost
6
7 LOG_CHANNEL=stack
8 LOG_DEPRECATIONS_CHANNEL=null
9 LOG_LEVEL=debug
10
11 DB_CONNECTION=mysql
12 DB_HOST=127.0.0.1
13 DB_PORT=3306
14 DB_DATABASE=pwl_pos
15 DB_USERNAME=root
16 DB_PASSWORD=
17
18 BROADCAST_DRIVER=log
19 CACHE_DRIVER=file
20 FILESYSTEM_DISK=local
```

## B. MIGRATION

### Praktikum 2.1 - pembuatan file migrasi tanpa relasi

#### Steps

Kita buat file migrasi untuk table m\_level dengan perintah

#### Attachments

```
PS C:\laragon\www\POS> php artisan make:migration create_m_level_table --create=m_level
[INFO] Migration [C:\laragon\www\POS\database\migrations\2024_03_06_072927_create_m_level_table.php] created
```

```
migrations
├── 2014_10_12_000000_create_users_table.php
├── 2014_10_12_100000_create_password_reset_...
├── 2019_08_19_000000_create_failed_jobs_table...
├── 2019_12_14_000001_create_personal_access_...
└── 2024_03_06_072927_create_m_level_tab... U
```

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

```
public function up(): void
{
    Schema::create('m_level', function (Blueprint $table) {
        $table->id('level_id');
        $table->string('level_kode', 10)->unique();
        $table->string('level_nama', 100);
        $table->timestamps();
    });
}
```

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

```
PS C:\laragon\www\POS> php artisan migrate

[INFO] Preparing database.

Creating migration table ..... 61ms DONE

[INFO] Running migrations.

2014_10_12_000000_create_users_table ..... 53ms DONE
2014_10_12_100000_create_password_reset_tokens_table ..... 5ms DONE
2019_08_19_000000_create_failed_jobs_table ..... 33ms DONE
2019_12_14_000001_create_personal_access_tokens_table ..... 25ms DONE
2024_03_06_072927_create_m_level_table ..... 19ms DONE
```

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

Table	Action
<input type="checkbox"/> failed_jobs	★ Browse Structure Search Insert Empty
<input type="checkbox"/> migrations	★ Browse Structure Search Insert Empty
<input type="checkbox"/> m_level	★ Browse Structure Search Insert Empty
<input type="checkbox"/> password_reset_tokens	★ Browse Structure Search Insert Empty
<input type="checkbox"/> personal_access_tokens	★ Browse Structure Search Insert Empty
<input type="checkbox"/> users	★ Browse Structure Search Insert Empty
6 tables	Sum

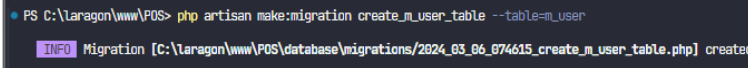

Buat table database dengan migration untuk table m\_kategori yang sama-sama tidak memiliki foreign key

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

[INFO] Migration [C:\laragon\www\POS\database\migrations\2024_03_06_073922_create_m_kategori_table.php] created

public function up(): void
{
    Schema::create('m_kategori', function (Blueprint $table) {
        $table->id('kategori_id');
        $table->string('kategori_kode', 10)->unique();
        $table->string('kategori_nama', 100);
    });
}
```

## Praktikum 2.2 - Pembuatan file migrasi dengan relasi

Steps	Attachments
Buka terminal VSCode kalian, dan buat file migrasi untuk table m_user	
Buka file migrasi untuk table m_user, dan modifikasi seperti berikut	

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

```
PS C:\laragon\www\POS> php artisan migrate:fresh

Dropping all tables .....

INFO Preparing database.

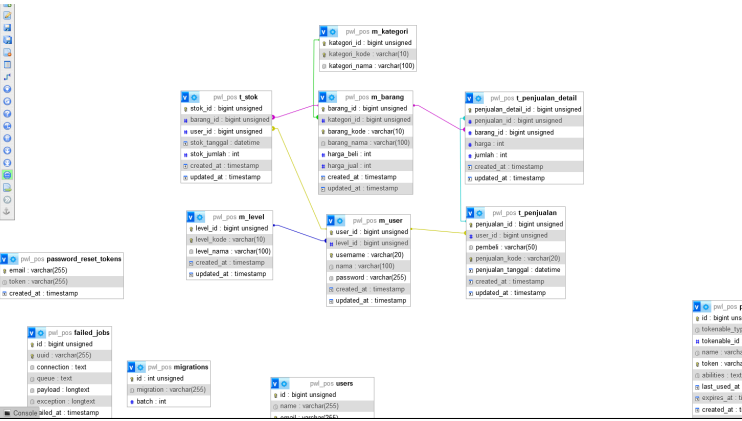
Creating migration table .....

INFO Running migrations.

2014_10_12_000000_create_users_table .....
2014_10_12_100000_create_password_reset_tokens_table .....
2019_08_19_000000_create_failed_jobs_table .....
2019_12_14_000001_create_personal_access_tokens_table .....
2024_03_06_072927_create_m_level_table .....
2024_03_06_073922_create_m_kategori_table .....
2024_03_06_074615_create_m_user_table .....
```

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

m_barang
t_penjualan
t_stok
t_penjualan_detail



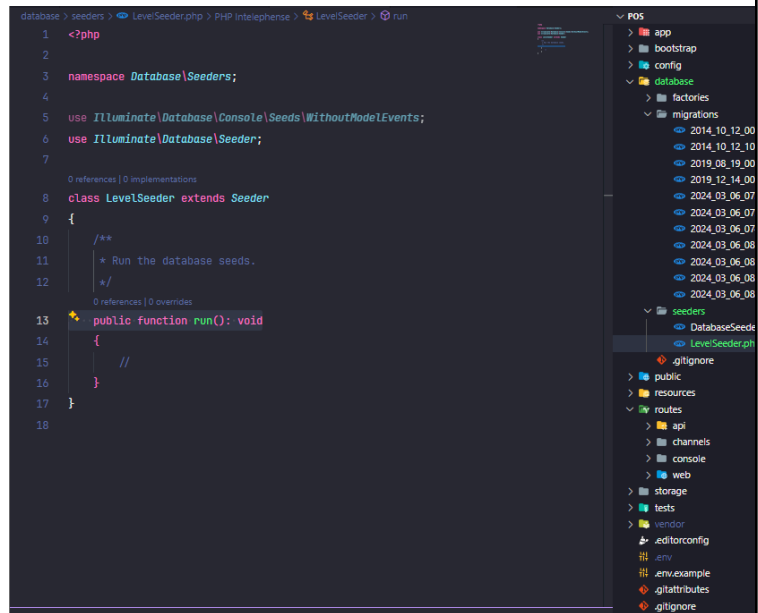
C. SEEDER

Praktikum 3 – Membuat file seeder

Steps

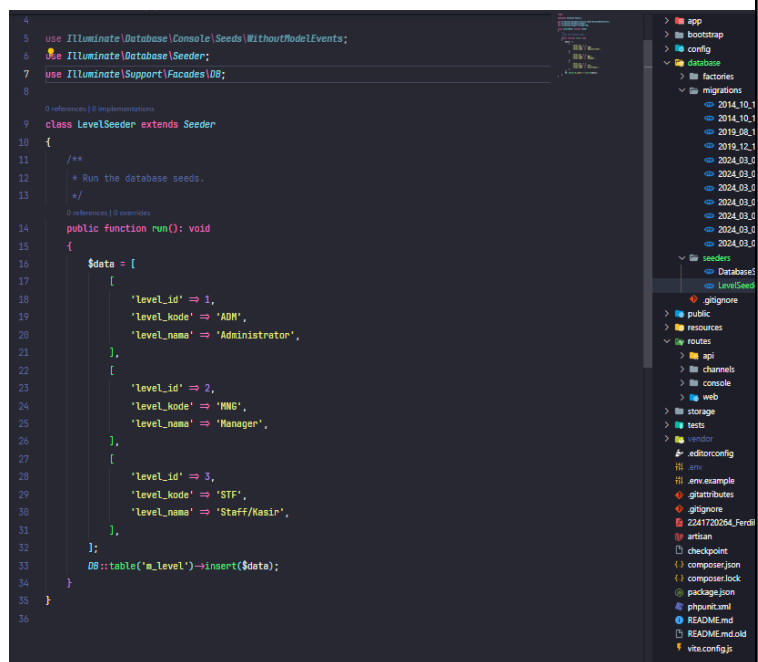
Attachments

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




```
1 <?php
2
3 namespace Database\Seeders;
4
5 use Illuminate\Database\Console\Seeds\WithoutModelEvents;
6 use Illuminate\Database\Seeder;
7
8 class LevelSeeder extends Seeder
9 {
10     /**
11      * Run the database seeds.
12      */
13     public function run(): void
14     {
15         //
16     }
17 }
18
```

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



```
4 use Illuminate\Database\Console\Seeds\WithoutModelEvents;
5 use Illuminate\Database\Seeder;
6 use Illuminate\Support\Facades\DB;
7
8 class LevelSeeder extends Seeder
9 {
10     /**
11      * Run the database seeds.
12      */
13     public function run(): void
14     {
15         $data = [
16             [
17                 'level_id' => 1,
18                 'level_kode' => 'ADM',
19                 'level_nama' => 'Administrator',
20             ],
21             [
22                 'level_id' => 2,
23                 'level_kode' => 'MNG',
24                 'level_nama' => 'Managen',
25             ],
26             [
27                 'level_id' => 3,
28                 'level_kode' => 'STF',
29                 'level_nama' => 'Staff/Kasir',
30             ],
31         ];
32         DB::table('m_level')->insert($data);
33     }
34 }
35
```

Selanjutnya, kita jalankan file seeder untuk table m\_level pada terminal



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

Ketika seeder berhasil dijalankan maka akan tampil data pada table m\_level

<input type="checkbox"/> Show all	Number of rows: 25	Filter rows: Search this table	Sort by ke
Extra options			
<input type="checkbox"/>	<input type="checkbox"/> Edit	<input type="checkbox"/> Copy	<input type="checkbox"/> Delete
1	ADM	Administrator	NULL
2	MNG	Manager	NULL
3	STF	Staff/Kasir	NULL

Sekarang kita buat file seeder untuk table m\_user yang me-refer ke table m\_level

```
PS C:\laragon\www\POS> php artisan make:seeder UserSeeder
INFO Seeder [C:\laragon\www\POS\database\seeders\UserSeeder.php] created succe
```

Modifikasi file class UserSeeder seperti berikut

```
public function run(): void
{
    $data = [
        [
            'user_id' => 1,
            'level_id' => 1,
            'username' => 'admin',
            'nama' => 'Administrator',
            'password' => Hash::make('admin'),
        ],
        [
            'user_id' => 2,
            'level_id' => 2,
            'username' => 'manager',
            'nama' => 'Manager',
            'password' => Hash::make('manager'),
        ],
        [
            'user_id' => 3,
            'level_id' => 3,
            'username' => 'staff',
            'nama' => 'Staff/Kasir',
            'password' => Hash::make('staff'),
        ],
    ];
    DB::table('m_user')->insert($data);
}
```

Jalankan perintah untuk mengeksekusi class UserSeeder

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

Perhatikan hasil seeder pada table  
m\_user

	user_id	level_id	username	nama	password
<input type="checkbox"/>	1	1	admin	Administrator	\$2y\$12\$ph1Bpl3w6cGELjsNIBcuJillb/pINS9gs
<input type="checkbox"/>	2	2	manager	Manager	\$2y\$12\$H.jqyX1V02wBc.Nt793rOtkM9Peq6ms
<input type="checkbox"/>	3	3	staff	Staff/Kasir	\$2y\$12\$4aoAeXfb4SKN1ZPF2DDQ3.3oGsCJq

☐ Check all    With selected: Edit Copy Delete Export

```
9 class KategoriSeeder extends Seeder
10
11     /**
12      * Run the database seeds.
13      */
14     public function run(): void
15     {
16         $data = [
17             [
18                 'kategori_id' => 1,
19                 'kategori_kode' => 'MKN',
20                 'kategori_nama' => 'Makanan',
21             ],
22             [
23                 'kategori_id' => 2,
24                 'kategori_kode' => 'MNM',
25                 'kategori_nama' => 'Minuman',
26             ],
27             [
28                 'kategori_id' => 3,
29                 'kategori_kode' => 'JKT',
30                 'kategori_nama' => 'Jaket',
31             ],
32             [
33                 'kategori_id' => 4,
34                 'kategori_kode' => 'CLN',
35                 'kategori_nama' => 'Celana',
36             ],
37             [
38                 'kategori_id' => 5,
39                 'kategori_kode' => 'TPI',
40                 'kategori_nama' => 'Topi',
41             ]
42         ];
43         DB::table('m_kategori')->insert($data);
44     }
45 }
```



```

database > seeders > PenjualanDetailSeeder.php >
7 use Illuminate\Support\Facades\DB;
8
9 class PenjualanDetailSeeder extends Seeder
10 {
11     /**
12      * Run the database seeds.
13      */
14     public function run(): void
15     {
16         $data = [
17             [
18                 'penjualan_id' => 1,
19                 'barang_id' => 1,
20                 'harga' => 10000,
21                 'jumlah' => 10,
22             ],
23             [
24                 'penjualan_id' => 1,
25                 'barang_id' => 2,
26                 'harga' => 20000,
27                 'jumlah' => 10,
28             ],
29             [
30                 'penjualan_id' => 1,
31                 'barang_id' => 3,
32                 'harga' => 30000,
33                 'jumlah' => 10,
34             ],
35             [
36                 'penjualan_id' => 2,
37                 'barang_id' => 4,
38                 'harga' => 40000,
39                 'jumlah' => 10,
40             ],
41             [
42                 'penjualan_id' => 2,
43                 'barang_id' => 5

```

```
<?php
```

```
namespace Database\Seeders;
```

```
use Illuminate\Database\Console\Seeds\WithoutModelEvents;
```

```
use Illuminate\Database\Seeder;
```

```
use Illuminate\Support\Facades\DB;
```

0 references | 0 implementations

```
class StokSeeder extends Seeder
```

```
{
```

```
    /**
```

```
     * Run the database seeds.
```

```
    */
```

0 references | 0 overrides

```
    public function run(): void
```

```
    {
```

```
        $data = [
```

```
            [
```

```
                'barang_id' => 1,
```

```
                'user_id' => 1,
```

```
                'stok_tanggal' => '2024-03-07',
```

```
                'stok_jumlah' => 100,
```

```
            ],
```

```
            [
```

```
                'barang_id' => 2,
```

```
                'user_id' => 1,
```

```
                'stok_tanggal' => '2024-03-07',
```

```
                'stok_jumlah' => 100,
```

```
            ],
```

```
            [
```

```
                'barang_id' => 3,
```

```
                'user_id' => 1,
```

```
                'stok_tanggal' => '2024-03-07',
```

```
                'stok_jumlah' => 100,
```

```
            ],
```

```
            [
```

```
                'barang_id' => 4,
```

```

6 use Illuminate\Database\Seeder;
7 use Illuminate\Support\Facades\DB;
8
9 class BarangSeeder extends Seeder
10 {
11     /**
12      * Run the database seeds.
13      */
14     public function run(): void
15     {
16         $data = [
17             [
18                 'barang_id' => 1,
19                 'kategori_id' => 1,
20                 'barang_kode' => 'MKN-001',
21                 'barang_nama' => 'Nasi Goreng',
22                 'harga_beli' => 10000,
23                 'harga_jual' => 15000,
24             ],
25             [
26                 'barang_id' => 2,
27                 'kategori_id' => 1,
28                 'barang_kode' => 'MKN-002',
29                 'barang_nama' => 'Mie Goreng',
30                 'harga_beli' => 8000,
31                 'harga_jual' => 12000,
32             ],
33             [
34                 'barang_id' => 3,
35                 'kategori_id' => 2,
36                 'barang_kode' => 'MNM-001',
37                 'barang_nama' => 'Es Teh',
38                 'harga_jual' => 5000,
39                 'harga_beli' => 3000,
40             ],
41             [

```

	<pre>9 class PenjualanSeeder extends Seeder 10 { 11     /** 12      * Run the database seeds. 13      */ 14     0 references   0 overrides 15     public function run(): void 16     { 17         \$data = [ 18             [ 19                 'penjualan_id' =&gt; 1, 20                 'user_id' =&gt; 3, 21                 'pembeli' =&gt; 'Rian', 22                 'penjualan_kode' =&gt; '00000000000000000001', 23                 'penjualan_tanggal' =&gt; '2024-03-06' 24             ], 25             [ 26                 'penjualan_id' =&gt; 2, 27                 'user_id' =&gt; 3, 28                 'pembeli' =&gt; 'Rian', 29                 'penjualan_kode' =&gt; '00000000000000000002', 30                 'penjualan_tanggal' =&gt; '2024-03-06' 31             ], 32             [ 33                 'penjualan_id' =&gt; 3, 34                 'user_id' =&gt; 3, 35                 'pembeli' =&gt; 'Rian', 36                 'penjualan_kode' =&gt; '00000000000000000003', 37                 'penjualan_tanggal' =&gt; '2024-03-06' 38             ], 39             [ 40                 'penjualan_id' =&gt; 4, 41                 'user_id' =&gt; 3, 42                 'pembeli' =&gt; 'Rian', 43                 'penjualan_kode' =&gt; '00000000000000000004', 44                 'penjualan_tanggal' =&gt; '2024-03-06' 45             ] 46         ] 47     } 48 }</pre>
--	--

D. DB FACADE

Praktikum 4 – Implementasi DB Facade

Steps

Kita buat controller dahulu untuk mengelola data pada table m\_level

Attachments

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

Kita modifikasi dulu untuk routing-nya, ada di PWL\_POS/routes/web.php

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

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

Selanjutnya, kita modifikasi file LevelController untuk menambahkan 1 data ke table m\_level

```
class LevelController extends Controller
{
    1 reference | 0 overrides
    public function index() {
        DB::insert('insert into m_level(level_kode, level_nama,
            created_at) values(?, ?, ?)', ['CUS', 'Pelanggan', now()]);

        return 'Insert data baru berhasil';
    }
}
```

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

localhost:7000/POS/publ

Insert data baru berhasil

	level_id	level_kode	level_nama	created_at
<input type="checkbox"/> Edit Copy Delete	1	ADM	Administrator	NULL
<input type="checkbox"/> Edit Copy Delete	2	MNG	Manager	NULL
<input type="checkbox"/> Edit Copy Delete	3	STF	Staff/Kasir	NULL
<input type="checkbox"/> Edit Copy Delete	4	CUS	Pelanggan	2024-03-07 05:11:19

Selanjutnya, kita modifikasi lagi file LevelController untuk meng-update data di table m\_level seperti berikut

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

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

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:7000/POS/pub

Update data berhasil. Jumlah data yang diupdate 1 baris

← T →	level_id	level_kode	level_nama	created_at
<input type="checkbox"/> Edit Copy Delete	1	ADM	Administrator	NULL
<input type="checkbox"/> Edit Copy Delete	2	MNG	Manager	NULL
<input type="checkbox"/> Edit Copy Delete	3	STF	Staff/Kasir	NULL
<input type="checkbox"/> Edit Copy Delete	4	CUS	Customer	2024-03-07 05:11:19

Kita coba modifikasi lagi file LevelController untuk melakukan proses hapus data

```
class LevelController extends Controller
{
    1 reference | 0 overrides
    public function index() {
        // DB::insert('insert into m_level(level_kode, level_nama,
        // created_at) values(?, ?, ?)', ['CUS', 'Pelanggan', now()])
        // return 'Insert data baru berhasil';

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

        $row = DB::delete('delete from m_level where level_kode =
        (keyword) return
        return "Delete data berhasil. Jumlah data yang dihapus $row
        baris";
    }
}
```

← → ↻ 🛡️ 📄 localhost:7000/POS/publ

Delete data berhasil. Jumlah data yang dihapus 1 baris

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

Method terakhir yang kita coba adalah untuk menampilkan data yang ada di table

m\_level. Kita modifikasi file LevelController seperti berikut

```
class LevelController extends Controller
{
    1 reference | 0 overrides
    public function index() {
        // DB::insert('insert into m_level(level_kode, level_nama,
        created_at) values(?, ?, ?)', ['CUS', 'Pelanggan', now()]);
        // return 'Insert data baru berhasil';

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

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

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

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

```
@extends('base')

@section('content')
    <div class="container">
        <div class="row">
            <div class="col-md-12">
                <h1>Data Level Pengguna</h1>
                <table border="1" cellpadding="2" cellspacing="1">
                    <tr>
                        <th>ID</th>
                        <th>Kode Level</th>
                        <th>Nama Level</th>
                    </tr>
                    @foreach($data as $d)
                        <tr>
                            <td>{{ $d->level_id }}</td>
                            <td>{{ $d->level_kode }}</td>
                            <td>{{ $d->level_nama }}</td>
                        </tr>
                    @endforeach
                </table>
            </div>
        </div>
    </div>
@endsection
```

Silahkan dicoba pada browser dan amati apa yang terjadi

## Data Level Pengguna

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

### E. QUERY BUILDER

#### Praktikum 5 – Implementasi Query Builder

Steps

Attachments



Kita buat controller dahulu untuk mengelola data pada table m_kategori	<pre>PS C:\laragon\www\POS&gt; php artisan make:controller KategoriCor</pre>																																			
Kita modifikasi dulu untuk routing-nya, ada di PWL_POS/routes/web.php	<pre>Route::get('/', function() {     return view('welcome'); });  Route::get('/level', [LevelController::class, 'index']) Route::get('/kategori', [KategoriController::class, 'in</pre>																																			
Selanjutnya, kita modifikasi file KategoriController untuk menambahkan 1 data ke table m_kategori	<pre>class KategoriController extends Controller {     1 reference   0 overrides     public function index() {         \$data = [             'kategori_kode' =&gt; 'SNK',             'kategori_nama' =&gt; 'Snack/Makanan Ringan',             'created_at' =&gt; now()         ];         DB::table('m_kategori')-&gt;insert(\$data);         return 'Insert data baru berhasil';     } }</pre>																																			
Kita coba jalankan di browser dengan url localhost/PWL_POS/public/kategori dan amati apa yang terjadi pada table m_kategori di database, screenshot perubahan yang ada pada table m_kategori	<div><div>← → ↺ 🛡️ 📄 localhost:7000/POS/public/k</div><div>Insert data baru berhasil</div><div><div>← →</div><div>▼ kategori_id</div><table><thead><tr><th></th><th>kategori_id</th><th>kategori_kode</th><th>kategori_nama</th><th>created_at</th></tr></thead><tbody><tr><td><input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete</td><td>1</td><td>MKN</td><td>Makanan</td><td>NULL</td></tr><tr><td><input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete</td><td>2</td><td>MNM</td><td>Minuman</td><td>NULL</td></tr><tr><td><input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete</td><td>3</td><td>JKT</td><td>Jaket</td><td>NULL</td></tr><tr><td><input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete</td><td>4</td><td>CLN</td><td>Celana</td><td>NULL</td></tr><tr><td><input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete</td><td>5</td><td>TPI</td><td>Topi</td><td>NULL</td></tr><tr><td><input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete</td><td>6</td><td>SNK</td><td>Snack/Makanan Ringan</td><td>2024-03-07 05:38:4</td></tr></tbody></table></div></div>		kategori_id	kategori_kode	kategori_nama	created_at	<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	1	MKN	Makanan	NULL	<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	2	MNM	Minuman	NULL	<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	3	JKT	Jaket	NULL	<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	4	CLN	Celana	NULL	<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	5	TPI	Topi	NULL	<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	6	SNK	Snack/Makanan Ringan	2024-03-07 05:38:4
	kategori_id	kategori_kode	kategori_nama	created_at																																
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	1	MKN	Makanan	NULL																																
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	2	MNM	Minuman	NULL																																
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	3	JKT	Jaket	NULL																																
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	4	CLN	Celana	NULL																																
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	5	TPI	Topi	NULL																																
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	6	SNK	Snack/Makanan Ringan	2024-03-07 05:38:4																																

Selanjutnya, kita modifikasi lagi file KategoriController untuk meng-update data di table m\_kategori seperti berikut

```
2 references | 0 implementations
class KategoriController extends Controller
{
    1 reference | 0 overrides
    public function index() {
        // $data = [
        //     'kategori_kode' => 'SNK',
        //     'kategori_nama' => 'Snack/Makanan
        Ringan',
        //     'created_at' => now()
        // ];
        // DB::table('m_kategori')->insert($data);
        // return 'Insert data baru berhasil';

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

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

← → ↻ localhost:7000/POS/public/

**Update data berhasil. Jumlah data yang diupdate 1 baris**

		kategori_id	kategori_kode	kategori_nama	created_at
<input type="checkbox"/>	Edit Copy Delete	1	MKN	Makanan	NULL
<input type="checkbox"/>	Edit Copy Delete	2	MNM	Minuman	NULL
<input type="checkbox"/>	Edit Copy Delete	3	JKT	Jaket	NULL
<input type="checkbox"/>	Edit Copy Delete	4	CLN	Celana	NULL
<input type="checkbox"/>	Edit Copy Delete	5	TPI	Topi	NULL
<input type="checkbox"/>	Edit Copy Delete	6	SNK	Snack	2024-03-07 05:38:50

Kita coba modifikasi lagi file KategoriController untuk melakukan proses hapus data

```
class KategoriController extends Controller
{
    1 reference | 0 overrides
    public function index() {
        // $data = [
        //     'kategori_kode' => 'SNK',
        //     'kategori_nama' => 'Snack/Makanan Ringan',
        //     'created_at' => now()
        // ];
        // DB::table('m_kategori')->insert($data);
        // return 'Insert data baru berhasil';

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

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



localhost:7000/POS/public/ka

Delete data berhasil. Jumlah data yang dihapus 1 baris

				kategori_id	kategori_kode	kategori_nama	created_at
<input type="checkbox"/>				1	MKN	Makanan	NULL
<input type="checkbox"/>				2	MNM	Minuman	NULL
<input type="checkbox"/>				3	JKT	Jaket	NULL
<input type="checkbox"/>				4	CLN	Celana	NULL
<input type="checkbox"/>				5	TPI	Topi	NULL

Method terakhir yang kita coba adalah untuk menampilkan data yang ada di table

m\_kategori. Kita modifikasi file KategoriController seperti berikut

```
class KategoriController extends Controller
{
    1 reference | 0 overrides

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

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

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

        $row = DB::table('m_kategori')->get();
        return view('kategori', ['data' => $row]);
    }
}
```

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

```
@extends('base')

@section('content')
    <div class="container">
        <div class="row">
            <div class="col-md-12">
                <h1>Data Kategori</h1>
                <table border="1" cellpadding="2" cellspacing="2">
                    <tr>
                        <th>ID</th>
                        <th>Kode Kategori</th>
                        <th>Nama Kategori</th>
                    </tr>
                    @foreach($data as $d)
                        <tr>
                            <td>{{ $d->kategori_id }}</td>
                            <td>{{ $d->kategori_kode }}</td>
                            <td>{{ $d->kategori_nama }}</td>
                        </tr>
                    @endforeach
                </table>
            </div>
        </div>
    </div>
@endsection
```

Silahkan dicoba pada browser dan amati apa yang terjadi.

## Data Kategori

ID	Kode Kategori	Nama Kategori
1	MKN	Makanan
2	MNM	Minuman
3	JKT	Jaket
4	CLN	Celana
5	TPI	Topi

### F. ELOQUENT ORM

#### Praktikum 6 – Implementasi Eloquent ORM

Steps	Attachments
<p>Kita buat file model untuk tabel m_user dengan mengetikkan perintah</p>	<pre>PS C:\laragon\www\POS&gt; php artisan make:model UserModel</pre> <p>INFO Model [C:\laragon\www\POS\app\Models\UserModel.php] created succes</p>
<p>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</p>	
<p>Kita buka file UserModel.php dan modifikasi seperti berikut</p>	<pre>&lt;?php  namespace App\Models;  use Illuminate\Database\Eloquent\Factories\HasFactory; use Illuminate\Database\Eloquent\Model;  0 references   0 implementations class UserModel extends Model {     use HasFactory;      0 references     protected \$table = 'm_user';      0 references     protected \$primaryKey = 'user_id'; }</pre>
<p>Kita modifikasi route web.php untuk mencoba routing ke controller UserController</p>	<pre>Route::get('/', function() {     return view('welcome'); });  Route::get('/level', [LevelController::class, 'index']); Route::get('/kategori', [KategoriController::class, 'index']); Route::get('/user', [UserController::class, 'index']);</pre>

Sekarang, kita buat file controller UserController dan memodifikasinya seperti berikut

```
class UserController extends Controller
{

    1 reference | 0 overrides

    public function index()
    {
        $user = UserModel::all();
        return view('user.index', ['data' => $user])
    }
}
```

Kemudian kita buat view user.blade.php

```
@extends('base')

@section('content')
    <div class="container">
        <div class="row">
            <div class="col-12">
                <h1>User Profile</h1>
                <table border="1" cellpadding="2" cellspacing="2">
                    <tr>
                        <th>ID</th>
                        <th>Username</th>
                        <th>Nama</th>
                        <th>ID Level Pengguna</th>
                    </tr>
                    @foreach($data as $d)
                        <tr>
                            <td>{{ $d->user_id }}</td>
                            <td>{{ $d->username }}</td>
                            <td>{{ $d->nama }}</td>
                            <td>{{ $d->level_id }}</td>
                        </tr>
                    @endforeach
                </table>
            </div>
        </div>
    </div>
@endsection
```

Jalankan di browser, catat dan laporkan apa yang terjadi

← → ↻ 🛡️ 📄 localhost:7000/POS/public

## Data User

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

Setelah itu, kita modifikasi lagi file UserController

```
[reference] U overrides  
public function index()  
{  
    $data = [  
        'username' => 'customer-1',  
        'nama' => 'Pelanggan',  
        'password' => Hash::make('12345'),  
        'level_id' => 4,  
    ];  
    UserModel::insert($data);  
  
    $user = UserModel::all();  
    return view('user', ['data' => $user]);  
}
```

Jalankan di browser, amati dan laporkan apa yang terjadi

← → ↻ 🛡️ 📄 localhost:7000/POS/public

## 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	4



Kita modifikasi lagi file UserController menjadi seperti berikut	<pre>public function index()      \$data = [         'nama' =&gt; 'Pelanggan Pertama'     ];     UserModel::where('username', 'customer-1')-&gt;update(\$data);      \$user = UserModel::all();     return view('user', ['data' =&gt; \$user]);</pre>																				
Jalankan di browser, amati dan laporkan apa yang terjadi	<div>localhost:7000/POS/public/u</div> <h2>Data User</h2> <table><tr><th>ID</th><th>Username</th><th>Nama</th><th>ID Level Pengguru</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><tr><td>4</td><td>customer-1</td><td>Pelanggan Pertama</td><td>4</td></tr></table>	ID	Username	Nama	ID Level Pengguru	1	admin	Administrator	1	2	manager	Manager	2	3	staff	Staff/Kasir	3	4	customer-1	Pelanggan Pertama	4
ID	Username	Nama	ID Level Pengguru																		
1	admin	Administrator	1																		
2	manager	Manager	2																		
3	staff	Staff/Kasir	3																		
4	customer-1	Pelanggan Pertama	4																		
G. Penutup																					
No	Questions	Answers																			
1.	Pada Praktikum 1 - Tahap 5, apakah fungsi dari APP_KEY pada file setting .env Laravel?	<div>1. Digunakan untuk enkripsi dan dekripsi data menggunakan fungsi encrypt() dan decrypt().</div> <div>2. Digunakan untuk melakukan generate string acak yang aman menggunakan fungsi Str::random().</div> <div>3. Digunakan untuk menanda-tangani dan verifikasi data menggunakan fungsi hash_hmac().</div> <div>4. Digunakan untuk generate token autentikasi unik menggunakan fungsi Hash::make().</div>																			
2.	Pada Praktikum 1, bagaimana kita men-generate nilai untuk APP_KEY?	Dengan menjalankan command berikut pada terminal`php artisan key:generate`																			
3.	Pada Praktikum 2.1 - Tahap 1, secara default Laravel memiliki berapa file migrasi?	<div>4 file migrasi default</div> <div>1. Create_users_table: membuat table users</div> <div>2. Create_password_reset_token: membuat table</div>																			

	dan untuk apa saja file migrasi tersebut?	<p>password_reset_token</p> <p>3. Create_failed_jobs: membuat table failed_jobs digunakan untuk menyimpan job queue yang gagal</p> <p>4. Create_personal_access_tokens: membuat table personal_access_tokens</p>
4.	Secara default, file migrasi terdapat kode \$table->timestamps();, apa tujuan/output dari fungsi tersebut?	<b>\$table-&gt;timestamps()</b> digunakan untuk menambahkan field nullable created_at dan updated_at timestamps ke dalam table.
5.	Pada File Migrasi, terdapat fungsi \$table->id(); Tipe data apa yang dihasilkan dari fungsi tersebut?	Hasil dari fungsi <b>\$table-&gt;id()</b> bertipe data big integer (8-byte).
6.	Apa bedanya hasil migrasi pada table m_level, antara menggunakan \$table->id(); dengan menggunakan \$table->id('level_id'); ?	Perbedaan \$table->id(); dengan \$table->id('level_id'); hanya terletak pada nama kolom dimana \$table->id(); akan memiliki kolom 'id' dan \$table->id('level_id'); adalah 'level_id'
7.	Pada migration, Fungsi ->unique() digunakan untuk apa?	Fungsi ->unique(); digunakan untuk menambahkan indeks unik pada kolom.
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') ?	Karena table m_user lah yang melakukan referensi pada table m_level, \$table->unsignedBigInteger('level_id') bertujuan untuk membuat kolom dengan tipe data big integer yang selalu bernilai positif, sedangkan m_level menggunakan \$table->id('level_id') untuk mendeklarasikan kolom primary key.
9.	Pada Praktikum 3 - Tahap 6, apa tujuan dari Class Hash? dan apa maksud dari kode program Hash::make('1234');?	<p>Class Hash Facade merupakan class built in dari laravel bertujuan untuk menghasilkan hashing Bcrypt dan Argon2 untuk penyimpanan user password.</p> <p>Maksud dari kode Hash::make('1234'); adalah untuk melakukan hashing pada string '1234' menggunakan Hash Facade.</p>
10.	Pada Praktikum 4 - Tahap 3/5/7, pada query builder terdapat tanda tanya (?), apa kegunaan dari tanda tanya (?) tersebut?	Tanda tanya (?) pada praktikum 4 merupakan sebuah place holder untuk parameterized query.

11.	Pada Praktikum 6 - Tahap 3, apa tujuan penulisan kode protected \$table = 'm_user'; dan protected \$primaryKey = 'user_id'; ?	Properti protected pada deklarasi variable tersebut dimaksudkan untuk mengganti access pada variable menjadi hanya dalam class tersebut dan juga class turunannya.
12.	Menurut kalian, lebih mudah menggunakan mana dalam melakukan operasi CRUD ke database (DB Façade / Query Builder / Eloquent ORM) ? jelaskan	Menurut saya Eloquent ORM terlihat jauh lebih mudah dan maintainable karena beberapa hal dibawah: <ol style="list-style-type: none"> <li>1. Autocomplete PHP pada kode editor yang membantu pembuatan.</li> <li>2. Query jauh menjadi lebih pendek.</li> <li>3. Memiliki struktur yang mudah.</li> </ol>