**Laporan Jobsheet Pemrograman Web Lanjut**

**JOBSHEET 03**

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

**ELOQUENT ORM**

**Dosen Pembimbing:**

**Dimas Wahyu Wibowo, S.T., M.T.**



**Disusun oleh:**

**Achmad Maulana Hamzah / 2A**

**2341720172**

**JURUSAN TEKNOLOGI INFORMASI**

**PROGRAM STUDI D-IV TEKNIK INFORMATIKA**

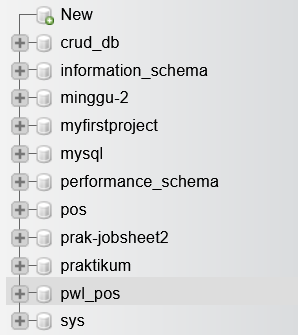
**POLITEKNIK NEGERI MALANG**

**2024**

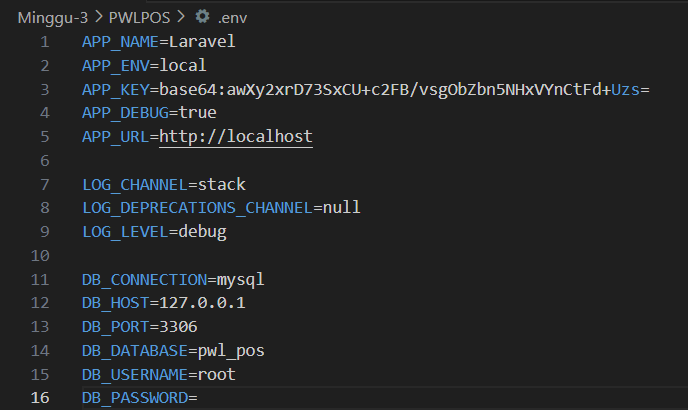
1. **Pengaturan Database**

**Praktikum 1 - Pengaturan database:**

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



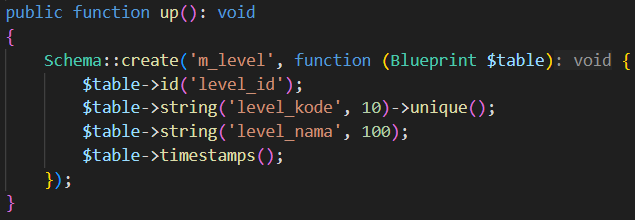
1. Buka aplikasi VSCode dan buka folder project PWL\_POS yang sudah kita buat
2. Copy file .env.example menjadi .env
3. Buka file .env, dan pastikan konfigurasi APP\_KEY bernilai. Jika belum bernilai silahkan kalian generate menggunakan php artisan.
4. Edit file .env dan sesuaikan dengan database yang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



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

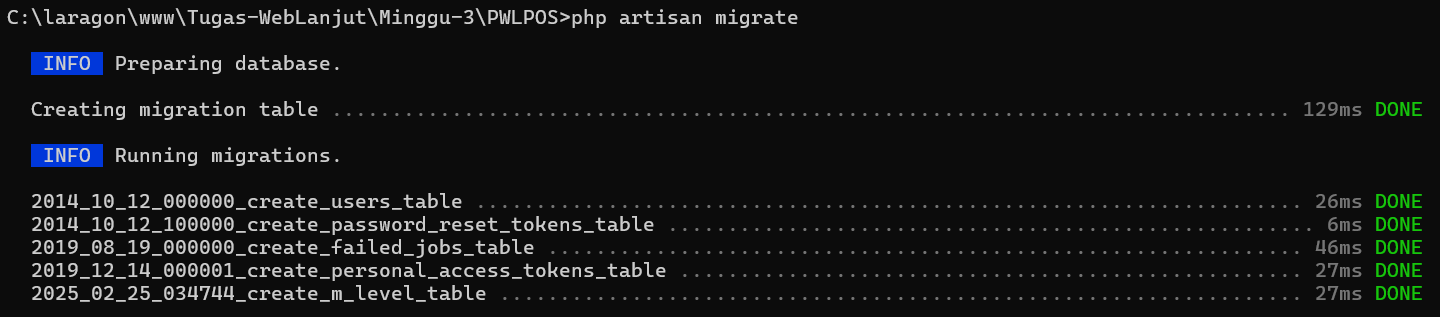
**Praktikum 2.1 - Pembuatan file migrasi tanpa relasi:**

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

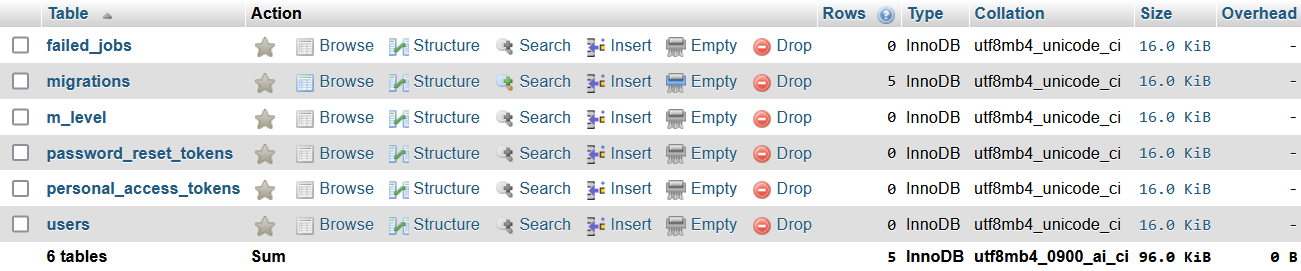


1. Simpan kode pada tahapan 4 tersebut, kemudian jalankan perintah ini pada terminal

VSCode untuk melakukan migrasi

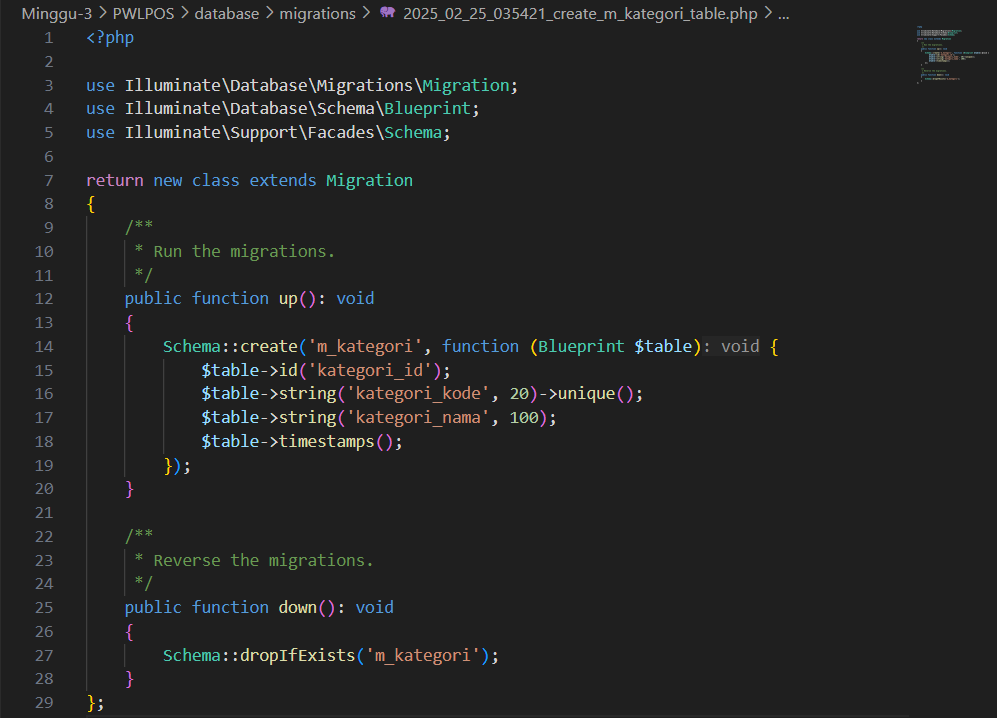


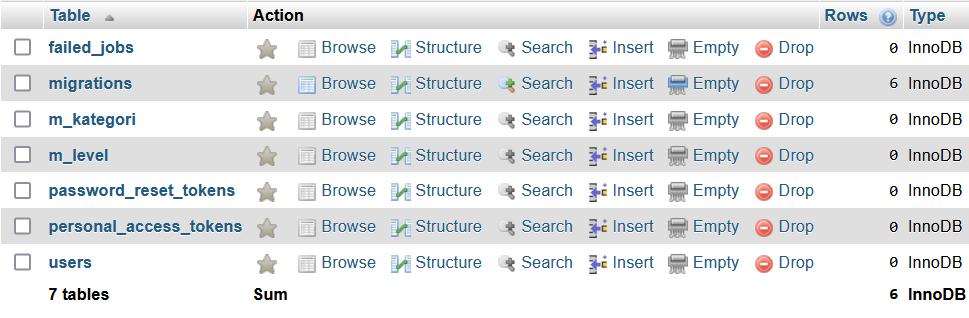
1. Kemudian kita cek di phpMyAdmin apakah table sudah ter-generate atau belum



1. Ok, table sudah dibuat di database
2. Buat table database dengan migration untuk table m\_kategori yang sama-sama tidak

memiliki foreign key



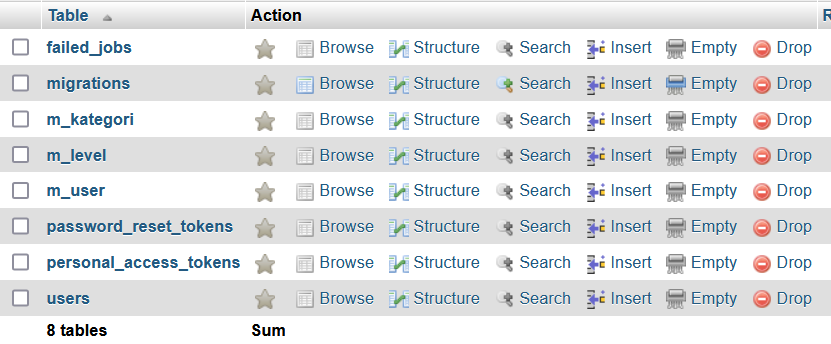


1. Laporkan hasil Praktikum-2.1 ini dan commit perubahan pada git.

**Praktikum 2.2 - Pembuatan file migrasi dengan relasi**

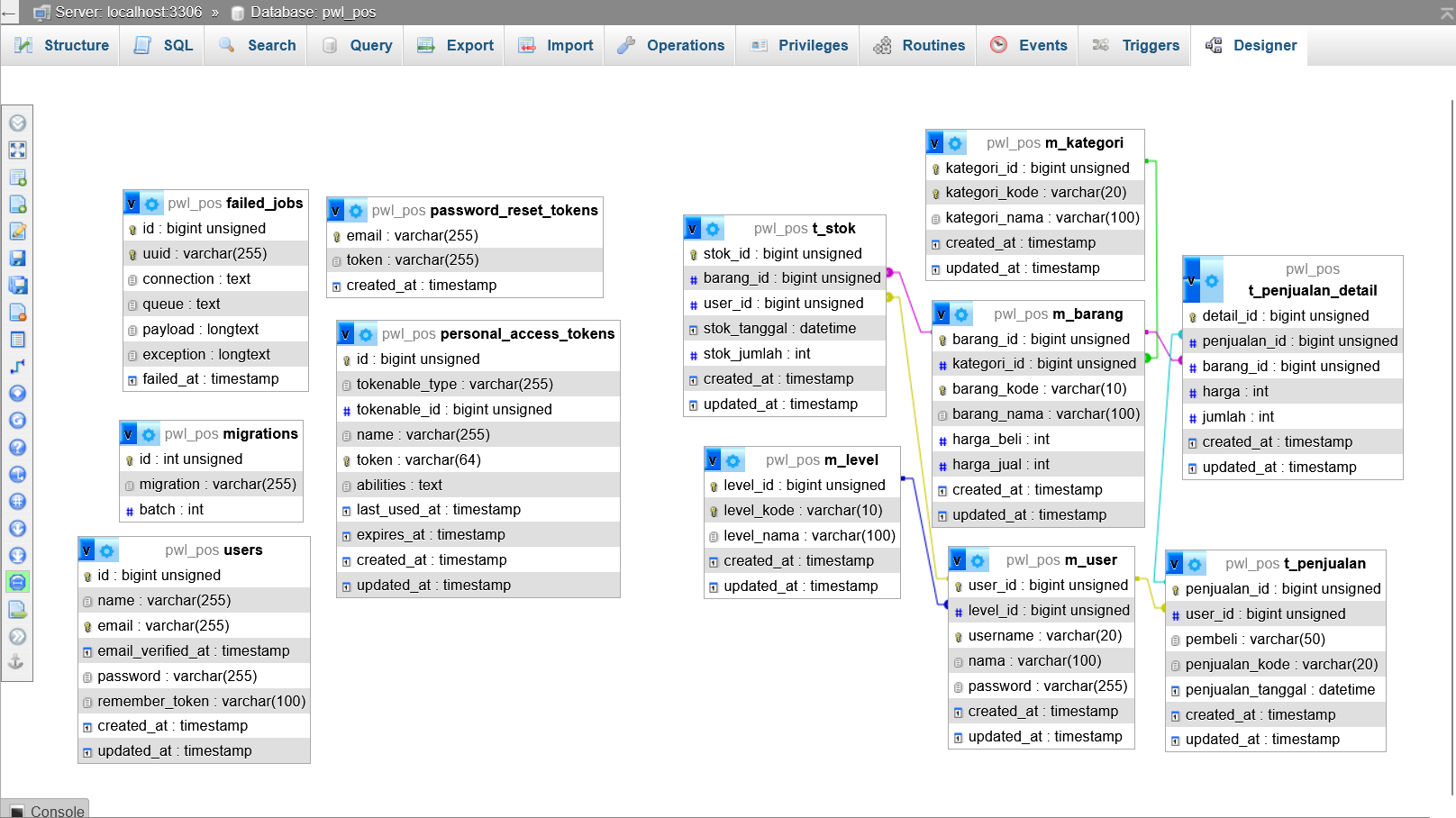
1. Buka terminal VSCode kalian, dan buat file migrasi untuk table m\_user
2. Buka file migrasi untuk table m\_user, dan modifikasi seperti berikut.
3. Simpan kode program Langkah 2, dan jalankan perintah php artisan migrate. Amati

apa yang terjadi pada database.



1. Buat table database dengan migration untuk table-tabel yang memiliki foreign key
2. Jika semua file migrasi sudah di buat dan dijalankan maka bisa kita lihat tampilan

designer pada phpMyAdmin seperti berikut



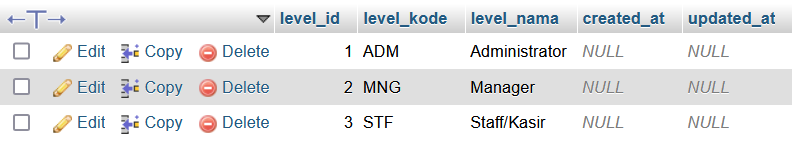
1. Laporkan hasil Praktikum-2.2 ini dan commit perubahan pada git.
2. **Seeder**

**Praktikum 3 – Membuat file seeder**

1. Kita akan membuat file seeder untuk table m\_level dengan mengetikkan perintah
2. Selanjutnya, untuk memasukkan data awal, kita modifikasi file tersebut di dalam

function run()

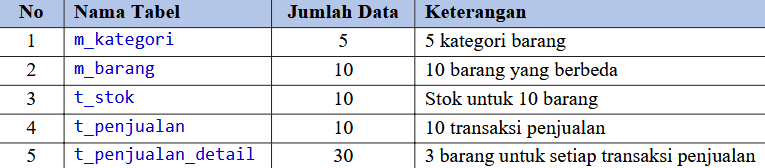
1. Selanjutnya, kita jalankan file seeder untuk table m\_level pada terminal
2. Ketika seeder berhasil dijalankan maka akan tampil data pada table m\_level



1. Sekarang kita buat file seeder untuk table m\_user yang me-refer ke table m\_level
2. Modifikasi file class UserSeeder seperti berikut
3. Jalankan perintah untuk mengeksekusi class UserSeeder
4. Perhatikan hasil seeder pada table m\_user



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













1. Jika sudah, laporkan hasil Praktikum-3 ini dan commit perubahan pada git
2. **DB Façade**

**Praktikum 4 – Implementasi DB Façade**

1. Kita buat controller dahulu untuk mengelola data pada table m\_level
2. Kita modifikasi dulu untuk routing-nya, ada di PWL\_POS/routes/web.php
3. Selanjutnya, kita modifikasi file LevelController untuk menambahkan 1 data ke table

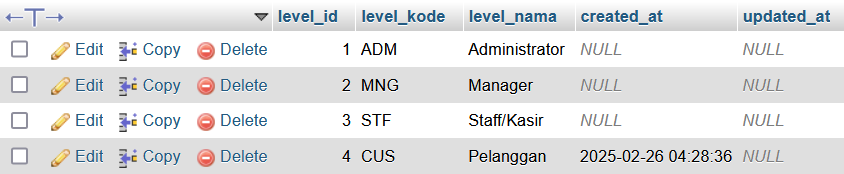
m\_level

1. 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





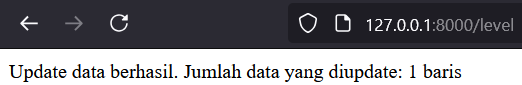
1. Selanjutnya, kita modifikasi lagi file LevelController untuk meng-update data di table

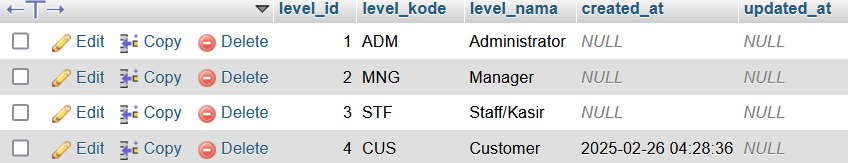
m\_level seperti berikut

1. Kita coba jalankan di browser dengan url localhost/PWL\_POS/public/level lagi dan

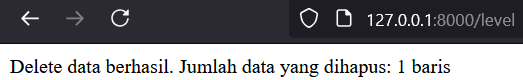
amati apa yang terjadi pada table m\_level di database, screenshot perubahan yang ada

pada table m\_level





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



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

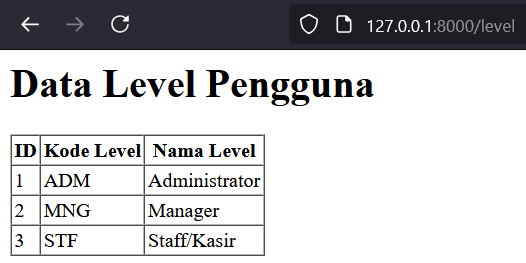
m\_level. Kita modifikasi file LevelController seperti berikut

1. 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

1. Silahkan dicoba pada browser dan amati apa yang terjadi



1. Laporkan hasil Praktikum-4 ini dan commit perubahan pada git.
2. **Query Builder**

**Praktikum 5 – Implementasi Query Builder**

1. Kita buat controller dahuku untuk mengelola data pada table m\_kategori
2. Kita modifikasi dulu untuk routing-nya, ada di PWL\_POS/routes/web.php
3. Selanjutnya, kita modifikasi file KategoriController untuk menambahkan 1 data ke

table m\_kategori

1. 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



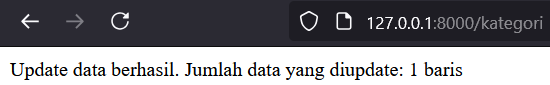
1. Selanjutnya, kita modifikasi lagi file KategoriController untuk meng-update data di

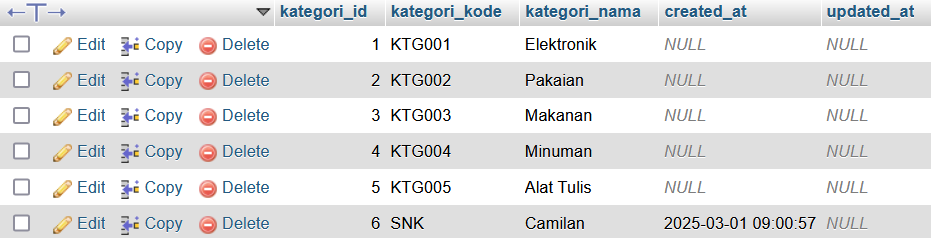
table m\_kategori seperti berikut

1. Kita coba jalankan di browser dengan url localhost/PWL\_POS/public/kategori lagi

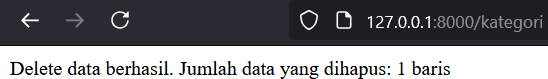
dan amati apa yang terjadi pada table m\_kategori di database, screenshot perubahan

yang ada pada table m\_kategori





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





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

m\_kategori. Kita modifikasi file KategoriController seperti berikut

1. 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

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



1. Laporkan hasil Praktikum-5 ini dan commit perubahan pada git
2. **Eloquent Orm**

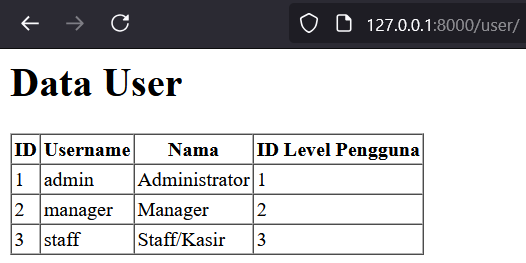
**Praktikum 6 – Implementasi Eloquent ORM**

1. Kita buat file model untuk tabel m\_user dengan mengetikkan perintah
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

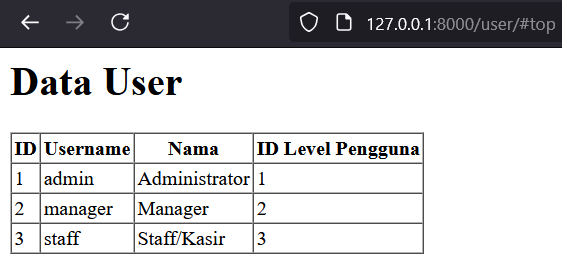
menggunakan file UserModel.php

1. Kita buka file UserModel.php dan modifikasi seperti berikut
2. Kita modifikasi route web.php untuk mencoba routing ke controller UserController
3. Sekarang, kita buat file controller UserController dan memodifikasinya seperti berikut
4. Kemudian kita buat view user.blade.php
5. Jalankan di browser, catat dan laporkan apa yang terjadi



Muncul data user pada website sesuai dengan yang ada di database

1. Setelah itu, kita modifikasi lagi file UserController
2. Jalankan di browser, amati dan laporkan apa yang terjadi
3. Kita modifikasi lagi file UserController menjadi seperti berikut
4. Jalankan di browser, amati dan laporkan apa yang terjadi



1. Jika sudah, laporkan hasil Praktikum-6 ini dan commit perubahan pada git
2. **Penutup**

**Jawablah pertanyaan berikut sesuai pemahaman materi di atas**

1. Pada Praktikum 1 - Tahap 5, apakah fungsi dari APP\_KEY pada file setting .env Laravel?

Jawab:

APP\_KEY digunakan untuk mengenkripsi data sensitif pada aplikasi Laravel, seperti password dan session. Nilai ini penting untuk memastikan keamanan data yang disimpan dalam aplikasi.

1. Pada Praktikum 1, bagaimana kita men-generate nilai untuk APP\_KEY?

Jawab:

Untuk men-generate APP\_KEY, kita menggunakan perintah berikut di terminal:

php artisan key:generate

Perintah ini akan membuat kunci baru dan otomatis menyimpannya di file .env pada baris:

APP\_KEY=base64:xxxxxxxxxxxxxxxxxxxx

1. Pada Praktikum 2.1 - Tahap 1, secara default Laravel memiliki berapa file migrasi?

dan untuk apa saja file migrasi tersebut?

Jawab:

Secara default, Laravel memiliki 3 file migrasi:

* create\_users\_table → Membuat tabel pengguna (users).
* create\_password\_reset\_tokens\_table → Membuat tabel untuk reset password.
* create\_failed\_jobs\_table → Membuat tabel untuk menyimpan pekerjaan (job) yang gagal.

1. Secara default, file migrasi terdapat kode $table->timestamps();, apa tujuan/output

dari fungsi tersebut?

Jawab:

Kode $table->timestamps(); berfungsi untuk membuat dua kolom otomatis:

* created\_at → Menyimpan tanggal dan waktu saat data ditambahkan.
* updated\_at → Menyimpan tanggal dan waktu saat data diubah.

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

fungsi tersebut?

Jawab:

Fungsi $table->id(); menghasilkan kolom dengan tipe data BIGINT (Auto Increment Primary Key).

1. Apa bedanya hasil migrasi pada table m\_level, antara menggunakan $table->id();

dengan menggunakan $table->id('level\_id'); ?

Jawab:

* $table->id(); → Akan membuat kolom id sebagai Primary Key secara default.
* $table->id('level\_id'); → Akan membuat kolom level\_id sebagai Primary Key dengan nama kolom yang sesuai.

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

Jawab:

Fungsi ->unique() digunakan untuk memastikan bahwa nilai dalam kolom tersebut tidak boleh ada yang sama (unik).

1. Pada Praktikum 2.2 - Tahap 2, kenapa kolom level\_id pada tabel m\_user

menggunakan $tabel->unsignedBigInteger('level\_id'), sedangkan kolom level\_id

pada tabel m\_level menggunakan $tabel->id('level\_id') ?

Jawab:

* Pada tabel m\_level, $table->id('level\_id') digunakan untuk membuat Primary Key.
* Sedangkan pada tabel m\_user, $table->unsignedBigInteger('level\_id') digunakan untuk membuat Foreign Key yang mengacu pada kolom level\_id di tabel m\_level.

1. Pada Praktikum 3 - Tahap 6, apa tujuan dari Class Hash? dan apa maksud dari kode

program Hash::make('1234');?

Jawab:

* Class Hash digunakan untuk mengenkripsi data password sebelum disimpan ke dalam database.
* Kode Hash::make('1234'); berfungsi untuk mengenkripsi password 1234 menggunakan metode bcrypt.

1. Pada Praktikum 4 - Tahap 3/5/7, pada query builder terdapat tanda tanya (?), apa

kegunaan dari tanda tanya (?) tersebut?

Jawab:

Tanda tanya ? digunakan sebagai placeholder dalam Query Builder untuk mencegah SQL Injection. Nilai parameter akan diisi secara otomatis pada saat query dieksekusi.

1. Pada Praktikum 6 - Tahap 3, apa tujuan penulisan kode protected $table =

‘m\_user’; dan protected $primaryKey = ‘user\_id’; ?

Jawab:

* $table → Digunakan untuk menentukan nama tabel yang digunakan pada model.
* $primaryKey → Digunakan untuk menentukan kolom yang menjadi Primary Key pada tabel

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

Jawab:

Eloquent ORM lebih mudah digunakan karena menggunakan konsep Object-Oriented Programming (OOP). Sedangkan pada Query Builder atau DB Façade, kita harus menulis query manual yang lebih panjang.