

**PRAKTIKUM PEMROGRAMAN WEB LANJUT**

**TUGAS PERTEMUAN 4**

**MODEL dan ELOQUENT ORM**

**DISUSUN OLEH :**

**GILANG PURNOMO**

**NIM:2341720042**



**PROGRAM STUDI D-IV TEKNIK INFORMATIKA**

**JURUSAN TEKNOLOGI INFORMASI**

**POLITEKNIK NEGERI MALANG**

**MARET 2025**

## Praktikum 1 - \$fillable

1. Buka file model dengan nama `UserModel.php` dan tambahkan `$fillable` seperti gambar di bawah ini

```
class UserModel extends Model
{
    use HasFactory;

    protected $table = 'm_user';
    protected $primaryKey = 'user_id';
    /**
     * The attributes that are mass assignable.
     *
     * @var array
     */
    protected $fillable = ['level_id', 'username', 'nama', 'password'];
}
```

2. Buka file controller dengan nama `UserController.php` dan ubah script untuk menambahkan data baru seperti gambar di bawah ini

```
1  <?php
2
3  namespace App\Http\Controllers;
4
5  use Illuminate\Http\Request;
6  use App\Models\UserModel;
7  use Illuminate\Support\Facades\Hash;
8
9  class UserController extends Controller
10 {
11     public function index()
12     {
13         $data = [
14             'level_id' => 2,
15             'username' => 'manager_dua',
16             'nama' => 'Manager 2',
17             'password' => Hash::make('12345')
18         ];
19         UserModel::create($data);
20
21         $user = UserModel::all();
22         return view('user', ['data' => $user]);
23     }
24 }
25
```

3. Simpan kode program Langkah 1 dan 2, dan jalankan perintah web server. Kemudian jalankan link `localhostPWL_POS/public/user` pada browser dan amati apa yang terjadi  
Jawab :



### 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
9	manager_dua	MANager 2	2

Penggunaan \$fillable ini memungkinkan pengisian data melalui mass assignment dengan nama atribut yang sudah di atur didalamnya, seperti level\_id, username, nama, dan password

- Ubah file model [UserModel.php](#) seperti pada gambar di bawah ini pada bagian \$fillable

```
protected $fillable = ['level_id', 'username', 'nama'];
```

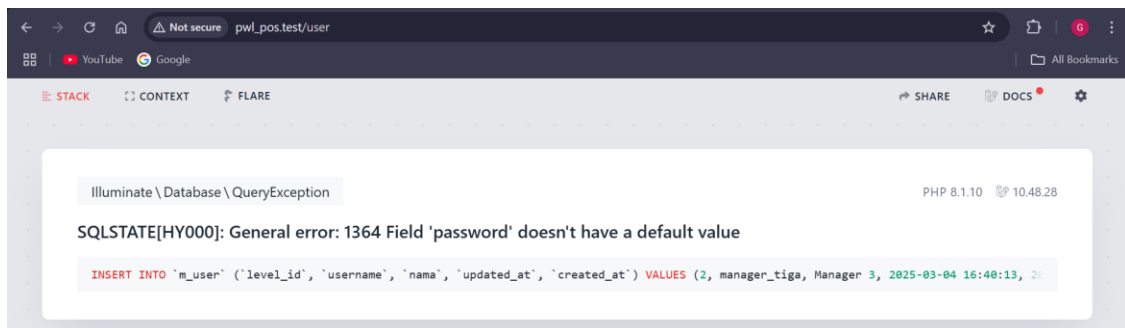
- Ubah kembali file controller [UserController.php](#) seperti pada gambar di bawah hanya bagian array pada \$data

```
public function index()
{
    $data = [
        'level_id' => 2,
        'username' => 'manager_tiga',
        'nama' => 'Manager 3',
        'password' => Hash::make('12345')
    ];
    UserModel::create($data);

    $user = UserModel::all();
    return view('user', ['data' => $user]);
}
```

- Simpan kode program Langkah 4 dan 5. Kemudian jalankan pada browser dan amati apa yang terjadi

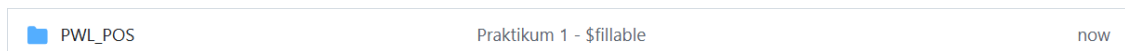
Jawab :



Terjadi error, karena nama kolom password tidak disertakan dalam \$fillable yang berada di model, sehingga laravel mengabaikannya saat create(). Laravel menggunakan mass assignment, yang hanya menyertakan atribut yang ada di \$fillable.

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

Jawab :



## Praktikum 2.1 – Retrieving Single Models

1. Buka file controller dengan nama `UserController.php` dan ubah script seperti gambar di bawah ini

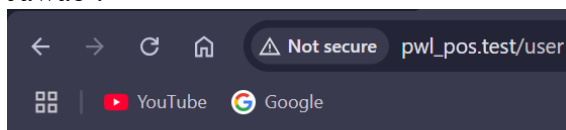
```
class UserController extends Controller
{
    public function index()
    {
        $user = UserModel::find(1);
        return view('user', ['data' => $user]);
    }
}
```

2. Buka file view dengan nama `user.blade.php` dan ubah script seperti gambar di bawah ini

```
<body>
<h1>Data User</h1>
<table border="1" cellpadding="2" cellspacing="0">
    <tr>
        <td>ID</td>
        <td>Username</td>
        <td>Nama</td>
        <td>ID Level Pengguna</td>
    </tr>
    <tr>
        <td>{{ $data->user_id }}</td>
        <td>{{ $data->username }}</td>
        <td>{{ $data->nama }}</td>
        <td>{{ $data->level_id }}</td>
    </tr>
</table>
</body>
```

3. Simpan kode program Langkah 1 dan 2. Kemudian jalankan pada browser dan amati apa yang terjadi dan beri penjelasan dalam laporan

Jawab :



### Data User

ID	Username	Nama	ID Level Pengguna
1	admin	Administrator	1

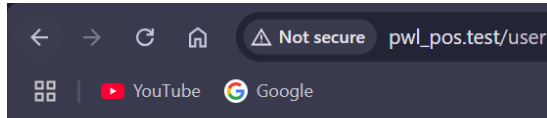
Jadi dari kode tersebut, `find(1)` ini untuk mencari data di tabel dengan `id = 1`.

4. Ubah file controller dengan nama `UserController.php` dan ubah script seperti gambar di bawah ini

```
class UserController extends Controller
{
    public function index()
    {
        $user = UserModel::where('level_id', 1)->first();
        return view('user', ['data' => $user]);
    }
}
```

5. Simpan kode program Langkah 4. Kemudian jalankan pada browser dan amati apa yang terjadi dan beri penjelasan dalam laporan

Jawab :



## Data User

ID	Username	Nama	ID Level Pengguna
1	admin	Administrator	1

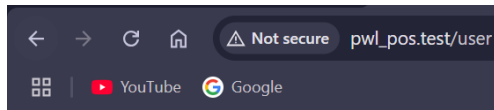
Jadi dari kode tersebut, where('level\_id', 1)->first(); untuk mengfilter data dengan level\_id = 1 dan mengambil satu baris pertama dalam pencarian.

6. Ubah file controller dengan nama **UserController.php** dan ubah script seperti gambar di bawah ini

```
class UserController extends Controller
{
    public function index()
    {
        $user = UserModel::firstWhere('level_id', 1);
        return view('user', ['data' => $user]);
    }
}
```

7. Simpan kode program Langkah 6. Kemudian jalankan pada browser dan amati apa yang terjadi dan beri penjelasan dalam laporan

Jawab :



## Data User

ID	Username	Nama	ID Level Pengguna
1	admin	Administrator	1

Output dan pencarian datanya sama seperti jawaban no 5, tetapi firstWhere() ini lebih singkat dan langsung mencari satu data berdasarkan kondisi.

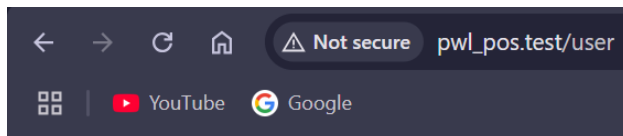
8. Ubah file controller dengan nama [UserController.php](#) dan ubah script seperti gambar di bawah ini

```
class UserController extends Controller
{
    public function index()
    {
        $user = UserModel::findOr(1, ['username', 'nama'], function () {
            abort(404);
        });

        return view('user', ['data' => $user]);
    }
}
```

9. Simpan kode program Langkah 8. Kemudian pada browser dan amati apa yang terjadi dan beri penjelasan dalam laporan

Jawab :



## Data User

ID	Username	Nama	ID Level Pengguna
	admin	Administrator	

Dari kode tersebut, `findOr(1, ['username', 'name'] function() {abort(404)})`; ini untuk mencari data berdasarkan primary key yaitu `id = 1` dan hanya kolom `username` dan `name` yang di tampilkan. Jika data tidak ditemukan maka akan menampilkan fungsi `abort(404)`

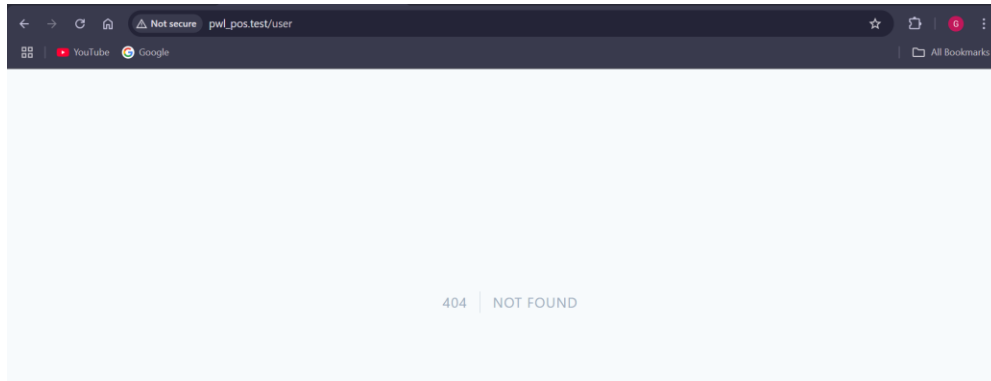
10. Ubah file controller dengan nama [UserController.php](#) dan ubah script seperti gambar di bawah ini

```
class UserController extends Controller
{
    public function index()
    {
        $user = UserModel::findOr(20, ['username', 'nama'], function () {
            abort(404);
        });

        return view('user', ['data' => $user]);
    }
}
```

11. Simpan kode program Langkah 10. Kemudian jalankan pada browser dan amati apa yang terjadi dan beri penjelasan dalam laporan

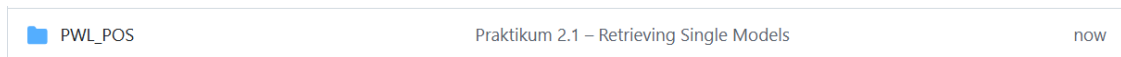
Jawab :



Hal ini bisa terjadi, karena primary key dengan id = 20 tidak ditemukan, oleh karena itu fungsi abort(404) dijalankan.

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

Jawab :



## Praktikum 2.2 – Not Found Exceptions

- Ubah file controller dengan nama [UserController.php](#) dan ubah script seperti gambar di bawah ini

```
class UserController extends Controller
{
    public function index()
    {
        $user = UserModel::findOrFail(1);
        return view('user', ['data' => $user]);
    }
}
```

- Simpan kode program Langkah 1. Kemudian jalankan pada browser dan amati apa yang terjadi dan beri penjelasan dalam laporan

Jawab :



## Data User

ID	Username	Nama	ID Level Pengguna
1	admin	Administrator	1

Dari kode tersebut untuk mencari satu data dengan id = 1, dengan menggunakan laravel eloquent ORM. Jika data tidak ada/tidak ditemukan maka laravel akan menampilkan

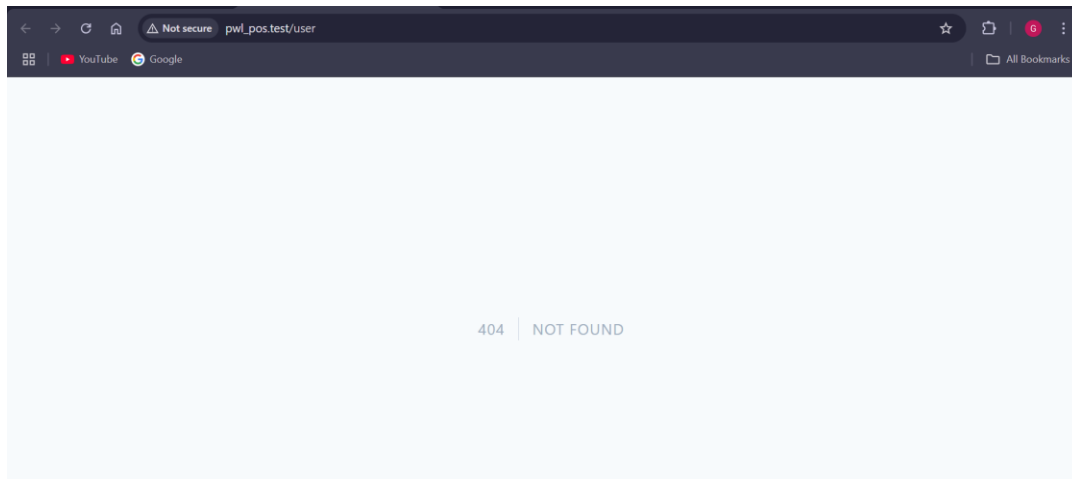
error `ModelNotFoundException`, tetapi secara default laravel menampilkan halaman 404 | NOT FOUND

- Ubah file controller dengan nama `UserController.php` dan ubah script seperti gambar di bawah ini

```
class UserController extends Controller
{
    public function index()
    {
        $user = UserModel::where('username', 'manager9')->firstOrFail();
        return view('user', ['data' => $user]);
    }
}
```

- Simpan kode program Langkah 3. Kemudian jalankan pada browser dan amati apa yang terjadi dan beri penjelasan dalam laporan

Jawab :



Menampilkan error, karena data dengan username = manager9 tidak ada di dalam tabel. Jika data yang di cari dalam tabel ada maka akan menampilkan data tersebut. Seperti



## Data User

ID	Username	Nama	ID Level Pengguna
2	manager	Manager	2

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

Jawab :





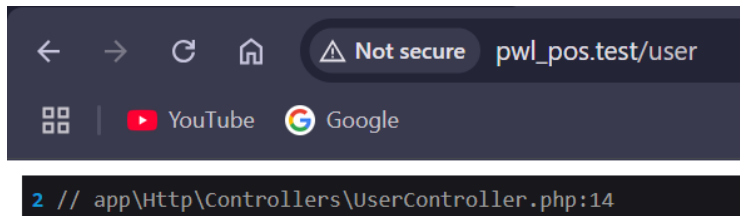
## Praktikum 2.3 – Retrieving Aggregates

1. Ubah file controller dengan nama `UserController.php` dan ubah script seperti gambar di bawah ini

```
class UserController extends Controller
{
    public function index()
    {
        $user = UserModel::where('level_id', 2)->count();
        dd($user);
        return view('user', ['data' => $user]);
    }
}
```

2. Simpan kode program Langkah 1. Kemudian jalankan pada browser dan amati apa yang terjadi dan beri penjelasan dalam laporan

Jawab :



Jadi itu menyaring data di tabel users dan mengambil data yang memiliki `level_id = 2`. Lalu `count()` menghitung jumlah data dan dilanjutkan `dd()` (dump and die) yang digunakan untuk menampilkan hasil perhitungan dan menghentikan eksekusi kode.

3. Buat agar jumlah script pada langkah 1 bisa tampil pada halaman browser, sebagai contoh bisa lihat gambar di bawah ini dan ubah script pada file view supaya bisa muncul datanya

### Data User

Jumlah Pengguna
2

Jawab :

`UserController.php`

```
class UserController extends Controller
{
    public function index()
    {
        $user = UserModel::where('level_id', 2)->count();
        return view('user', ['data' => $user]);
    }
}
```

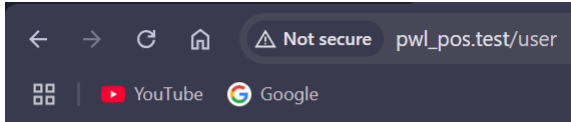
`User.blade.php`

```

<body>
  <h1>Data User</h1>
  <table border="1" cellpadding="2" cellspacing="0">
    <tr>
      <th>Jumlah Pengguna</th>
    </tr>
    <tr>
      <td>{{ $data }}</td>
    </tr>
  </table>
</body>

```

Output :

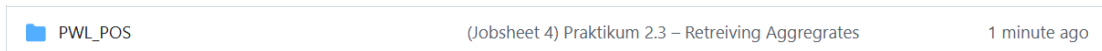


## Data User

Jumlah Pengguna
2

- Laporkan hasil Praktikum-2.3 ini dan commit perubahan pada git.

Jawab :



## Praktikum 2.4 – Retrieving or Creating Models

- Ubah file controller dengan nama **UserController.php** dan ubah script seperti gambar di bawah ini

```

class UserController extends Controller
{
    public function index()
    {
        $user = UserModel::firstOrCreate(
            [
                'username' => 'manager',
                'nama' => 'Manager',
            ],
        );
        return view('user', ['data' => $user]);
    }
}

```

- Ubah kembali file view dengan nama **user.blade.php** dan ubah script seperti gambar di bawah ini

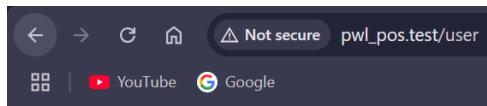
```

<body>
<h1>Data User</h1>
<table border="1" cellpadding="2" cellspacing="0">
<tr>
<td>ID</td>
<td>Username</td>
<td>Nama</td>
<td>ID Level Pengguna</td>
</tr>
<tr>
<td>{{ $data->user_id }}</td>
<td>{{ $data->username }}</td>
<td>{{ $data->nama }}</td>
<td>{{ $data->level_id }}</td>
</tr>
</table>
</body>

```

3. Simpan kode program Langkah 1 dan 2. Kemudian jalankan pada browser dan amati apa yang terjadi dan beri penjelasan dalam laporan

Jawab :



## Data User

ID	Username	Nama	ID Level Pengguna
2	manager	Manager	2

Jadi penggunaan firstOrCreate, jika username = manager dan nama = manager sudah ada di database, maka tidak akan ada perubahan di database dan hanya mengambil data yang sudah ada. Tetapi, jika data belum ada, maka Laravel akan menambahkan data tersebut ke database.

4. Ubah file controller dengan nama **UserController.php** dan ubah script seperti gambar di bawah ini

```

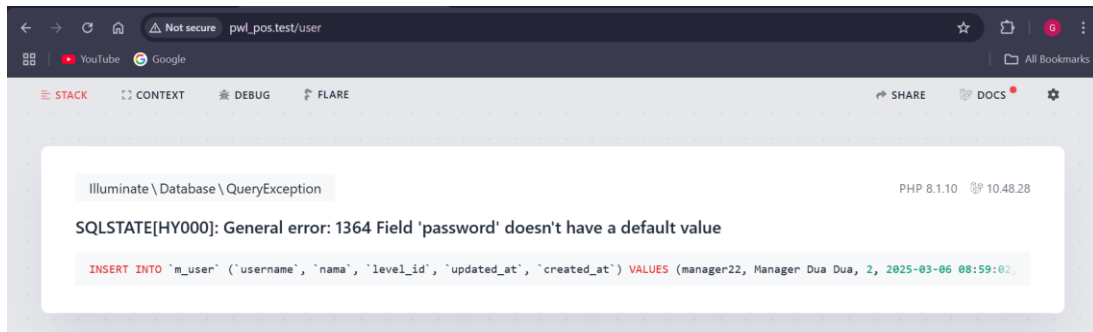
class UserController extends Controller
{
    public function index()
    {
        $user = UserModel::firstOrCreate(
            [
                'username' => 'manager22',
                'nama' => 'Manager Dua Dua',
                'password' => Hash::make('12345'),
                'level_id' => 2
            ],
        );

        return view('user', ['data' => $user]);
    }
}

```

5. Simpan kode program Langkah 4. Kemudian jalankan pada browser dan amati apa yang terjadi dan cek juga pada phpMyAdmin pada tabel **m\_user** serta beri penjelasan dalam laporan

Jawab :



Error, karena pada UserModel.php nama kolom password belum ada.

```
class UserModel extends Model
{
    use HasFactory;

    protected $table = 'm_user';
    protected $primaryKey = 'user_id';

    protected $fillable = ['level_id', 'username', 'nama'];
}
```

Jika ingin penginputan data, maka kolom password harus ada, karena kolom password juga tidak diperbolehkan untuk bernilai null.

Output jika kolom password sudah ditambahkan



## Data User

ID	Username	Nama	ID Level Pengguna
12	manager22	Manager Dua Dua	2

	user_id	level_id	username	nama	password	created_at	updated_at
<input type="checkbox"/>	1	1	admin	Administrator	\$2y\$12\$WxqAz2t7EBYtHYc88Kd4Cel2wrynL317EpJ5lsyKWkU...	NULL	NULL
<input type="checkbox"/>	2	2	manager	Manager	\$2y\$12\$Gxm0FIUSQuqf1W0svUJrweyUX5Y37A4qpZmFUe.11YU...	NULL	NULL
<input type="checkbox"/>	3	3	staff	Staff/Kasir	\$2y\$12\$UBjXmb7j88Dxb.gf.Nl8eEzvP8HTsRKv8s2BsMZdp...	NULL	NULL
<input type="checkbox"/>	4	4	customer-1	Pelanggan Pertama	\$2y\$12\$32TVAOpnjKGg.RF.w8cX.UQn7H-KWRptml1uNiV56m...	NULL	NULL
<input type="checkbox"/>	9	2	manager_dua	Manager 2	\$2y\$12\$V7Aa.Kf1utQSVwosdye8.1P4ewSjH9W.GS1Gim08pT...	2025-03-04 16:19:27	2025-03-04 16:19:27
<input type="checkbox"/>	12	2	manager22	Manager Dua Dua	\$2y\$12\$0gn31EPfW4LgOR8i3XVBcJZ4pGg5pZzCXa1b4fnSn0...	2025-03-06 09:11:51	2025-03-06 09:11:51

Dikarenakan data manager22 belum ada, maka saat menjalankannya di browser data manager22 ditambahkan dalam database dan kemudian data tersebut ditampilkan

- Ubah file controller dengan nama **UserController.php** dan ubah script seperti gambar di bawah ini

```

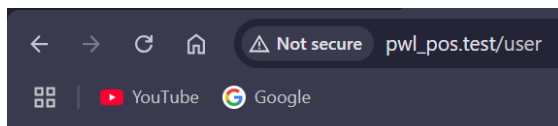
class UserController extends Controller
{
    public function index()
    {
        $user = UserModel::firstOrCreate(
            [
                'username' => 'manager',
                'nama' => 'Manager',
            ],
        );

        return view('user', ['data' => $user]);
    }
}

```

7. Simpan kode program Langkah 6. Kemudian jalankan pada browser dan amati apa yang terjadi dan beri penjelasan dalam laporan

Jawab :



## Data User

ID	Username	Nama	ID Level Pengguna
2	manager	Manager	2

Jadi, penggunaan firstOrCreate ini jika data dengan username = manager dan nama = Manager sudah ada di database, maka Laravel akan mengembalikan datanya dan menampilkan data yang ditemukan tersebut. Tetapi jika data belum ada di database, maka Laravel akan membuat objek baru, tetapi tidak langsung menyimpannya. Jika ingin menyimpan data maka melakukannya secara manual dengan : \$user->save();

8. Ubah file controller dengan nama **UserController.php** dan ubah script seperti gambar di bawah ini

```

class UserController extends Controller
{
    public function index()
    {
        $user = UserModel::firstOrCreate(
            [
                'username' => 'manager33',
                'nama' => 'Manager Tiga Tiga',
                'password' => Hash::make('12345'),
                'level_id' => 2
            ],
        );

        return view('user', ['data' => $user]);
    }
}

```

9. Simpan kode program Langkah 8. Kemudian jalankan pada browser dan amati apa yang terjadi dan cek juga pada phpMyAdmin pada tabel m\_user serta beri penjelasan dalam laporan

Jawab :



## Data User

ID	Username	Nama	ID Level Pengguna
	manager33	Manager Tiga Tiga	2

← T →	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\$WxvqAz2t7EBYHyc86Kd4CeL2wrynL317EpJ5isyKWkU...	NULL	NULL
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	2	2	manager	Manager	\$2y\$12\$GXm0FIUSQuqf1W0svUJrneyUX5Y37A4qpZmFue.11YU...	NULL	NULL
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	3	3	staff	Staff/Kasir	\$2y\$12\$UBjIXmb7j88Dxb.gf.Nf8eEzvP8HTsRKv8s2BsmZdp...	NULL	NULL
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	4	4	customer-1	Pelanggan Pertama	\$2y\$12\$32TvAOpnjKGg.RF.w8cX.UQn7HkWRptml1uNiiv56m...	NULL	2025-03-03 04:41:47
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	9	2	manager_dua	Manager 2	\$2y\$12\$V7Aa.Kf1utQSVwosdye/8.1P4ewSjH9W.GS1Gim08pT...	2025-03-04 16:19:27	2025-03-04 16:19:27
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	12	2	manager22	Manager Dua Dua	\$2y\$12\$0gn31IEPv4LgOR6i3XVBe.JZ4pGg5pZzCXa1b4fnSn0...	2025-03-06 09:11:51	2025-03-06 09:11:51

Nah, datanya ditampilkan. Namun di database masih belum ada, karena belum menambahkan fungsi save();. Oleh karena itu IDnya masih belum ada juga.

- Ubah file controller dengan nama **UserController.php** dan ubah script seperti gambar di bawah ini

```
class UserController extends Controller
{
    public function index()
    {
        $user = UserModel::firstOrCreate(
            [
                'username' => 'manager33',
                'nama' => 'Manager Tiga Tiga',
                'password' => Hash::make('12345'),
                'level_id' => 2
            ],
        );
        $user->save();
        return view('user', ['data' => $user]);
    }
}
```

- Simpan kode program Langkah 9. Kemudian jalankan pada browser dan amati apa yang terjadi dan cek juga pada phpMyAdmin pada tabel **m\_user** serta beri penjelasan dalam laporan

Jawab :



## Data User

ID	Username	Nama	ID Level Pengguna
13	manager33	Manager Tiga Tiga	2

	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\$WxvqAz217EBYHyc86Kd4CeL2wrynL317EpJ5isyKWkU...	NULL	NULL
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	2	2	manager	Manager	\$2y\$12\$Gxm0FIUSQuqf1W0svUJrweyUX5Y37A4qpZmFue.11YU...	NULL	NULL
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	3	3	staff	Staff/Kasir	\$2y\$12\$UBjIXmb7j88Db.gf.Nl8eEzvP8HTsRKv8s2BsmZdp...	NULL	NULL
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	4	4	customer-1	Pelanggan Pertama	\$2y\$12\$32TvAOpnjKg..RF.w8cX.UQn7HkWRptml1uNliV56m...	NULL	2025-03-03 04:41:47
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	9	2	manager_dua	Manager Dua Dua	\$2y\$12\$V7Aa.Kf1utQSVwosdye/8.1P4ewSjH9W.GS1Gim08pT...	2025-03-04 16:19:27	2025-03-04 16:19:27
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	12	2	manager22	Manager Dua Dua	\$2y\$12\$0gn31IEPfv4LgOR6t3XVBeJZ4pGg5pZzCXa1b4fnSn0...	2025-03-06 09:11:51	2025-03-06 09:11:51
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	13	2	manager33	Manager Tiga Tiga	\$2y\$12\$UbyxN4x5XO.pZKQXYbTub2wIC.NXg4JlGl/9vKX7...	2025-03-06 09:34:46	2025-03-06 09:34:46

Setelah penambahan fungsi `save()`, maka data akan ditambahkan di dalam database dan ID sudah auto increment.

## 12. Laporkan hasil Praktikum-2.4 ini dan commit perubahan pada git.

Jawab :

PWL_POS	(Jobsheet 4) Praktikum 2.4 – Retrieving or Creating Models	now
---------	--	-----

## Praktikum 2.5 – Attribute Changes

- Ubah file controller dengan nama `UserController.php` dan ubah script seperti gambar di bawah ini

```
class UserController extends Controller
{
    public function index()
    {
        $user = UserModel::create([
            'username' => 'manager55',
            'nama' => 'Manager55',
            'password' => Hash::make('12345'),
            'level_id' => 2,
        ]);

        $user->username = 'manager56';

        $user->isDirty(); // true
        $user->isDirty('username'); // true
        $user->isDirty('nama'); // false
        $user->isDirty(['nama', 'username']); // true

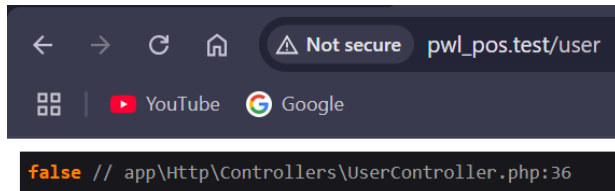
        $user->isClean(); // false
        $user->isClean('username'); // false
        $user->isClean('nama'); // true
        $user->isClean(['nama', 'username']); // false

        $user->save();

        $user->isDirty(); // false
        $user->isClean(); // true
        dd($user->isDirty());
    }
}
```

- Simpan kode program Langkah 1. Kemudian jalankan pada browser dan amati apa yang terjadi dan beri penjelasan dalam laporan

Jawab :



Jadi output tersebut bisa false, karena setelah melakukan penambahan data lalu melakukan perubahan username tetapi belum disimpan di database.

- Pengecekan status Dirty :

1. `$user->isDirty();` : untuk mengecek apakah ada perubahan pada objek model. Lalu `isDirty()` mengembalikan true karena username diubah dari manager55 ke manager56.
2. `$user->isDirty('username');` : lalu mengecek apakah 'username' telah berubah dan hasilnya mengembalikan true.
3. `$user->isDirty('nama');` : untuk mengecek apakah 'nama' telah berubah dan hasilnya mengembalikan false, karena nama tidak diubah.
4. `$user->isDirty(['nama', 'username']);` : Mengecek apakah 'nama' dan 'username' berubah dan akan mengembalikan true, karena (username) tadi telah berubah.

- Setelah itu pengecekan status clean

1. `$user->isClean();` : mengecek apakah tidak ada perubahan dan akan mengembalikan false, karena ada perubahan (username) .
2. `$user->isClean('username');` : Mengecek apakah 'username' tidak berubah dan akan mengembalikan false.
3. `$user->isClean('nama');` : Mengecek apakah 'nama' tidak berubah dan akan mengembalikan true, karena nama tidak berubah.
4. `$user->isClean(['nama', 'username']);` : mengecek apakah 'nama' dan 'username' tidak berubah dan akan mengembalikan false, karena username berubah.

- Lalu menyimpan perubahan ke database dengan `$user->save();`

- Pengecekan kembali setelah penyimpanan, jadi setelah penyimpanan tidak ada perubahan lagi.

- `isDirty()` akan mengembalikan false.

- `isClean()` akan mengembalikan true.

Oleh karena itu hasil outputnya false.

3. Ubah file controller dengan nama [UserController.php](#) dan ubah script seperti gambar di bawah ini



```

class UserController extends Controller
{
    public function index()
    {
        $user = UserModel::create([
            'username' => 'manager11',
            'nama' => 'Manager11',
            'password' => Hash::make('12345'),
            'level_id' => 2,
        ]);

        $user->username = 'manager12';

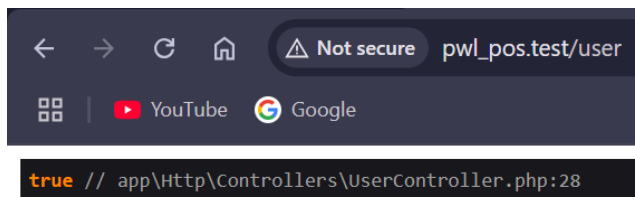
        $user->save();

        $user->wasChanged(); // true
        $user->wasChanged('username'); // true
        $user->wasChanged(['username', 'level_id']); // true
        $user->wasChanged('nama'); // false
        dd($user->wasChanged(['nama', 'username'])); // true
    }
}

```

4. Simpan kode program Langkah 3. Kemudian jalankan pada browser dan amati apa yang terjadi dan beri penjelasan dalam laporan

Jawab :



Jadi output tersebut bisa true, karena setelah melakukan penambahan data lalu melakukan perubahan username tetapi belum disimpan di database.

- Lalu \$user->save(); melakukan perubahan ke dalam database
- Pengecekan dengan wasChanged()
  1. \$user->wasChanged(); : hasilnya true, karena ada perubahan yang disimpan.
  2. \$user->wasChanged('username'); : hasilnya true, karena username berubah dari manager11 ke manager12.
  3. \$user->wasChanged(['username', 'level\_id']); : hasilnya true, karena username berubah tetapi level\_id tetap.
  4. \$user->wasChanged('nama'); : hasilnya false, karena nama tidak berubah.
  5. dd(\$user->wasChanged(['nama', 'username'])); : hasilnya true, karena salah satu atribut (username) telah berubah.

5. Laporkan hasil Praktikum-2.5 ini dan commit perubahan pada git.

Jawab :



## Praktikum 2.6 – Create, Read, Update, Delete (CRUD)

1. Buka file view pada [user.blade.php](#) dan buat scriptnya menjadi seperti di bawah ini

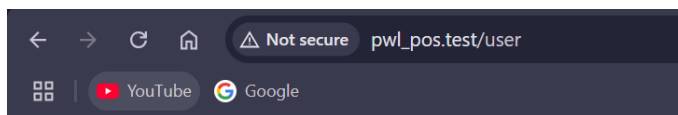
```
<body>
<h1>Data User</h1>
<a href="/user/tambah">+ Tambah User</a>
<table border="1" cellpadding="2" cellspacing="0">
  <tr>
    <td>ID</td>
    <td>Username</td>
    <td>Nama</td>
    <td>ID Level Pengguna</td>
    <td>Aksi</td>
  </tr>
  @foreach ($data as $d)
    <tr>
      <td>{{ $d->user_id }}</td>
      <td>{{ $d->username }}</td>
      <td>{{ $d->nama }}</td>
      <td>{{ $d->level_id }}</td>
      <td><a href="/user/ubah/{{ $d->user_id }}">Ubah</a> | <a href="/user/hapus/{{ $d->user_id }}">Hapus</a></td>
    </tr>
  @endforeach
</table>
</body>
```

2. Buka file controller pada UserController.php dan buat scriptnya untuk read menjadi seperti di bawah ini

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

3. Simpan kode program Langkah 1 dan 2. Kemudian jalankan pada browser dan amati apa yang terjadi dan beri penjelasan dalam laporan

Jawab :



### Data User

[+ Tambah User](#)

ID	Username	Nama	ID Level Pengguna	Aksi
1	admin	Administrator	1	<a href="#">Ubah</a>   <a href="#">Hapus</a>
2	manager	Manager	2	<a href="#">Ubah</a>   <a href="#">Hapus</a>
3	staff	Staff/Kasir	3	<a href="#">Ubah</a>   <a href="#">Hapus</a>
4	customer-1	Pelanggan Pertama	4	<a href="#">Ubah</a>   <a href="#">Hapus</a>
9	manager_dua	MANager 2	2	<a href="#">Ubah</a>   <a href="#">Hapus</a>
12	manager22	Manager Dua Dua	2	<a href="#">Ubah</a>   <a href="#">Hapus</a>
13	manager33	Manager Tiga Tiga	2	<a href="#">Ubah</a>   <a href="#">Hapus</a>
14	manager56	Manager55	2	<a href="#">Ubah</a>   <a href="#">Hapus</a>
16	manager12	Manager11	2	<a href="#">Ubah</a>   <a href="#">Hapus</a>

Jadi, UserController.php untuk mengambil semua data dari tabel user dan kemudian dikirim ke view user.blade.php. Sedangkan di user.blade.php ini melooping menampilkan data setiap user dalam bentuk tabel dan tiap baris terdapat Aksinya.

4. Langkah berikutnya membuat create atau tambah data user dengan cara bikin file baru pada view dengan nama `user_tambah.blade.php` dan buat scriptnya menjadi seperti di bawah ini

```
<body>
<h1>Form Tambah Data User</h1>
<form method="post" action="/user/tambah_simpan">

    {!! csrf_field() !!}

    <label>Username</label>
    <input type="text" name="username" placeholder="Masukan Username">
    <br>
    <label>Nama</label>
    <input type="text" name="nama" placeholder="Masukan Nama">
    <br>
    <label>Password</label>
    <input type="password" name="password" placeholder="Masukan Password">
    <br>
    <label>Level ID</label>
    <input type="number" name="level_id" placeholder="Masukan ID Level">
    <br><br>
    <input type="submit" class="btn btn-success" value="Simpan">

</form>
</body>
```

5. Tambahkan script pada routes dengan nama file `web.php`. Tambahkan seperti gambar di bawah ini

```
Route::get('/user/tambah', [UserController::class, 'tambah']);
```

6. Tambahkan script pada controller dengan nama file `UserController.php`. Tambahkan script dalam class dan buat method baru dengan nama `tambah` dan diletakan di bawah method `index` seperti gambar di bawah ini

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

    public function tambah()
    {
        return view('user_tambah');
    }
}
```

7. Simpan kode program Langkah 4 s/d 6. Kemudian jalankan pada browser dan klik link “+ **Tambah User**” amati apa yang terjadi dan beri penjelasan dalam laporan Jawab :



## Form Tambah Data User

Username

Nama

Password

ID Level

Jadi, jika klik tambah maka akan diarahkan ke /user/tambah yang memanggil method tambah di class UserController. Di method tambah akan mengembalikan view(user\_tambah.blade.php) yang berisi form untuk menambah data user.

8. Tambahkan script pada routes dengan nama file [web.php](#). Tambahkan seperti gambar di bawah ini

```
Route::post('/user/tambah_simpan', [UserController::class, 'tambah_simpan']);
```

9. Tambahkan script pada controller dengan nama file UserController.php. Tambahkan script dalam class dan buat method baru dengan nama tambah\_simpan dan diletakkan di bawah method tambah seperti gambar di bawah ini

```
public function tambah_simpan(Request $request)
{
    UserModel::create([
        'username' => $request->username,
        'nama' => $request->nama,
        'password' => Hash::make('$request->password'),
        'level_id' => $request->level_id
    ]);

    return redirect('/user');
}
```

10. Simpan kode program Langkah 8 dan 9. Kemudian jalankan link [localhost:8000/user/tambah](#) atau [localhost/PWL\\_POS/public/user/tambah](#) pada browser dan input formnya dan simpan, kemudian amati apa yang terjadi dan beri penjelasan dalam laporan

Jawab :



## Form Tambah Data User

Username

Nama

Password

ID Level

**Data User**

ID	Username	Nama	ID Level Pengguna	Aksi
1	admin	Administrator	1	<a href="#">Ubah</a> <a href="#">Hapus</a>
2	manager	Manager	2	<a href="#">Ubah</a> <a href="#">Hapus</a>
3	staff	Staff/Kasir	3	<a href="#">Ubah</a> <a href="#">Hapus</a>
4	customer-1	Pelanggan Pertama	4	<a href="#">Ubah</a> <a href="#">Hapus</a>
9	manager_dua	MAanager 2	2	<a href="#">Ubah</a> <a href="#">Hapus</a>
12	manager22	Manager Dua Dua	2	<a href="#">Ubah</a> <a href="#">Hapus</a>
13	manager33	Manager Tiga Tiga	2	<a href="#">Ubah</a> <a href="#">Hapus</a>
14	manager56	Manager55	2	<a href="#">Ubah</a> <a href="#">Hapus</a>
16	manager12	Manager11	2	<a href="#">Ubah</a> <a href="#">Hapus</a>
17	Customer09	Gilang	4	<a href="#">Ubah</a> <a href="#">Hapus</a>

**Save password?**

Username

Password

You can use saved passwords on any device. They're saved to Google Password Manager for gilangpurnomo0505@gmail.com.

Jadi setelah mengisi form dan klik Simpan. Data dikirim ke /user/tambah\_simpan menggunakan metode post, lalu Laravel akan menambahkan data baru ke database. Setelah berhasil, akan dialihkan ke halaman daftar user(/user) dan ditampilkan dalam bentuk tabel.

- Langkah berikutnya membuat update atau ubah data user dengan cara bikin file baru pada view dengan nama `user_ubah.blade.php` dan buat scripturnya menjadi seperti di bawah ini

```
<body>
<h1>Form Ubah Data User</h1>
<a href="/user">Kembali</a>
<br><br>

<form method="post" action="/user/ubah_simpan/{{ $data->user_id }}">
    {{ csrf_field() }}
    {{ method_field('PUT') }}

    <label>Username</label>
    <input type="text" name="username" placeholder="Masukan Username" value="{{ $data->username }}">
    <br>
    <label>Nama</label>
    <input type="text" name="nama" placeholder="Masukan Nama" value="{{ $data->username }}">
    <br>
    <label>Password</label>
    <input type="password" name="password" placeholder="Masukan Password" value="{{ $data->password }}">
    <br>
    <label>Level ID</label>
    <input type="number" name="level_id" placeholder="Masukan ID Level" value="{{ $data->level_id }}">
    <br><br>
    <input type="submit" class="btn btn-success" value="Ubah">
</form>
</body>
```

12. Tambahkan script pada routes dengan nama file `web.php`. Tambahkan seperti gambar di bawah ini

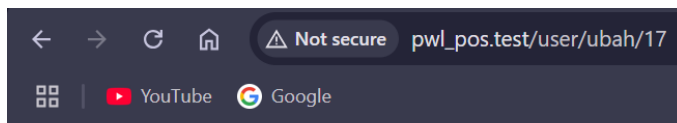
```
Route::get('/user/ubah/{id}', [UserController::class, 'ubah']);
```

13. Tambahkan script pada controller dengan nama file `UserController.php`. Tambahkan script dalam class dan buat method baru dengan nama `ubah` dan diletakan di bawah method `tambah_simpan` seperti gambar di bawah ini

```
public function ubah($id)
{
    $user = UserModel::find($id);
    return view('user_ubah', ['data' => $user]);
}
```

14. Simpan kode program Langkah 11 sd 13. Kemudian jalankan pada browser dan klik link “Ubah” amati apa yang terjadi dan beri penjelasan dalam laporan

Jawab :



## Form Ubah Data Userr

[Kembali](#)

Username   
Nama   
Password   
ID Level

Jadi ketika klik ubah, maka akan diarahkan ke halaman `/user/ubah/{id}` dan akan menampilkan data yang sesuai di dalam input field. Jika klik Ubah, maka Laravel akan menampilkan not found karena route `/user/ubah_simpan/{id}` belum dibuat.

15. Tambahkan script pada routes dengan nama file `web.php`. Tambahkan seperti gambar di bawah ini

```
Route::put('/user/ubah_simpan/{id}', [UserController::class, 'ubah_simpan']);
```

16. Tambahkan script pada controller dengan nama file `UserController.php`. Tambahkan script dalam class dan buat method baru dengan nama `ubah_simpan` dan diletakan di bawah method `ubah` seperti gambar di bawah ini

```

public function ubah_simpan($id, Request $request)
{
    $user = UserModel::find($id);

    $user->username = $request->username;
    $user->nama = $request->nama;
    $user->password = Hash::make($request->password);
    $user->level_id = $request->level_id;

    $user->save();

    return redirect('/user');
}

```

17. Simpan kode program Langkah 15 dan 16. Kemudian jalankan link [localhost:8000/user/ubah/1](http://localhost:8000/user/ubah/1) atau [localhost/PWL\\_POS/public/user/ubah/1](http://localhost/PWL_POS/public/user/ubah/1) pada browser dan ubah input formnya dan klik tombol ubah, kemudian amati apa yang terjadi dan beri penjelasan dalam laporan

Jawab :

- Melakukan pengubahan data



## Form Ubah Data Userr

[Kembali](#)

Username

Nama

Password

ID Level

- Setelah klik ubah.

Not secure pwl\_pos.test/user

YouTube Google

### Data User

[+ Tambah User](#)

ID	Username	Nama	ID Level Pengguna	Aksi
1	admin	Administrator	1	<a href="#">Ubah</a>   <a href="#">Hapus</a>
2	manager	Manager	2	<a href="#">Ubah</a>   <a href="#">Hapus</a>
3	staff	Staff/Kasir	3	<a href="#">Ubah</a>   <a href="#">Hapus</a>
4	customer-1	Pelanggan Pertama	4	<a href="#">Ubah</a>   <a href="#">Hapus</a>
9	manager_dua	MANager 2	2	<a href="#">Ubah</a>   <a href="#">Hapus</a>
12	manager22	Manager Dua Dua	2	<a href="#">Ubah</a>   <a href="#">Hapus</a>
13	manager33	Manager Tiga Tiga	2	<a href="#">Ubah</a>   <a href="#">Hapus</a>
14	manager56	Manager55	2	<a href="#">Ubah</a>   <a href="#">Hapus</a>
16	manager12	Manager11	2	<a href="#">Ubah</a>   <a href="#">Hapus</a>
17	Cust09	GILANG PURNOMO	4	<a href="#">Ubah</a>   <a href="#">Hapus</a>

Save password?

Username

Password

You can use saved passwords on any device. They're saved to Google Password Manager for gilangpurnomo0505@gmail.com.

Data berhasil diubah.

Jadi ketika klik ubah data dikirim ke `/user/ubah_simpan/{id}` menggunakan metode PUT dan data berhasil diperbarui, lalu halaman dialihkan kembali ke halaman `/user`.

18. Berikut untuk langkah delete . Tambahkan script pada routes dengan nama file [web.php](#). Tambahkan seperti gambar di bawah ini

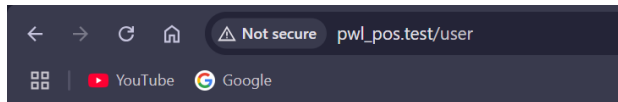
```
Route::get('/user/hapus/{id}', [UserController::class, 'hapus']);
```

19. Tambahkan script pada controller dengan nama file [UserController.php](#). Tambahkan script dalam class dan buat method baru dengan nama hapus dan diletakan di bawah method ubah\_simpan seperti gambar di bawah ini

```
public function hapus($id)
{
    $user = UserModel::find($id);
    $user->delete();

    return redirect('/user');
}
```

20. Simpan kode program Langkah 18 dan 19. Kemudian jalankan pada browser dan klik tombol hapus, kemudian amati apa yang terjadi dan beri penjelasan dalam laporan Jawab :



## Data User

[+ Tambah User](#)

ID	Username	Nama	ID Level Pengguna	Aksi
1	admin	Administrator	1	<a href="#">Ubah</a>   <a href="#">Hapus</a>
2	manager	Manager	2	<a href="#">Ubah</a>   <a href="#">Hapus</a>
3	staff	Staff/Kasir	3	<a href="#">Ubah</a>   <a href="#">Hapus</a>
4	customer-1	Pelanggan Pertama	4	<a href="#">Ubah</a>   <a href="#">Hapus</a>
9	manager_dua	MANager 2	2	<a href="#">Ubah</a>   <a href="#">Hapus</a>
12	manager22	Manager Dua Dua	2	<a href="#">Ubah</a>   <a href="#">Hapus</a>
13	manager33	Manager Tiga Tiga	2	<a href="#">Ubah</a>   <a href="#">Hapus</a>
14	manager56	Manager55	2	<a href="#">Ubah</a>   <a href="#">Hapus</a>
16	manager12	Manager11	2	<a href="#">Ubah</a>   <a href="#">Hapus</a>

Data saya dengan id = 17 sudah berhasil dihapus.

Jadi ketika klik hapus akan diarahkan ke URL /user/hapus/{id} dan Laravel mengeksekusi method hapus() di UserController dengan id yang sesuai. Setelah terhapus akan diarahkan kembali ke halaman/user.

21. Laporkan hasil Praktikum-2.6 ini dan commit perubahan pada git.

Jawab :





## Praktikum 2.7 – Relationships

1. Buka file model pada [UserModel.php](#) dan tambahkan scripnya menjadi seperti di bawah ini

```
class UserModel extends Model
{
    use HasFactory;

    protected $table = 'm_user';
    protected $primaryKey = 'user_id';
    /**
     * The attributes that are mass assignable.
     *
     * @var array
     */
    protected $fillable = ['level_id', 'username', 'nama', 'password'];

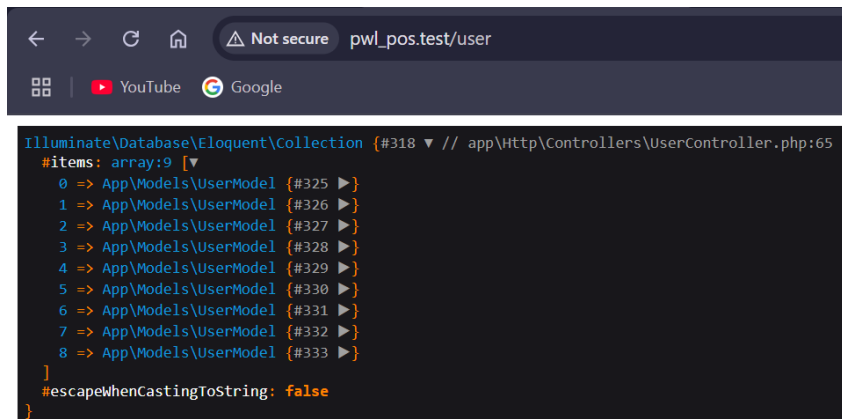
    public function level(): BelongsTo
    {
        return $this->belongsTo(LevelModel::class, 'level_id', 'level_id');
    }
}
```

2. Buka file controller pada [UserController.php](#) dan ubah method script menjadi seperti di bawah ini

```
public function index()
{
    $user = UserModel::with('level')->get();
    dd($user);
}
```

3. Simpan kode program Langkah 2. Kemudian jalankan link pada browser, kemudian amati apa yang terjadi dan beri penjelasan dalam laporan

Jawab :



```
Illuminate\Database\Eloquent\Collection {#318 ▼ // app\Http\Controllers\UserController.php:65
  #items: array:9 [▼]
    0 => App\Models\UserModel {#325 ▶}
    1 => App\Models\UserModel {#326 ▶}
    2 => App\Models\UserModel {#327 ▶}
    3 => App\Models\UserModel {#328 ▶}
    4 => App\Models\UserModel {#329 ▶}
    5 => App\Models\UserModel {#330 ▶}
    6 => App\Models\UserModel {#331 ▶}
    7 => App\Models\UserModel {#332 ▶}
    8 => App\Models\UserModel {#333 ▶}
  ]
  #escapeWhenCastingToString: false
}
```

Jadi, relasi belongsTo ini menghubungkan UserModel dengan LevelModel. Kolom level\_id di tabel users akan dicocokkan dengan level\_id di tabel level. Lalu Fungsi ini memungkinkan kita untuk mengakses data level dari user menggunakan \$user->level. Di UserController.php, ini mengambil semua data user dengan relasi level menggunakan with('level') dan menampilkannya dalam bentuk array menggunakan dd(\$user);

- Buka file controller pada [UserController.php](#) dan ubah method script menjadi seperti di bawah ini

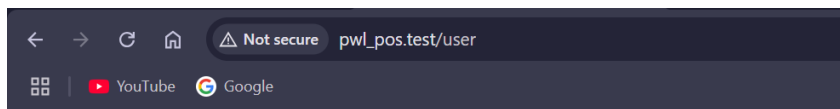
```
public function index()
{
    $user = UserModel::with('level')->get();
    return view('user', ['data' => $user]);
}
```

- Buka file view pada [user.blade.php](#) dan ubah script menjadi seperti di bawah ini

```
<body>
<h1>Data User</h1>
<a href="/user/tambah">+ Tambah User</a>
<table border="1" cellpadding="2" cellspacing="0">
  <tr>
    <td>ID</td>
    <td>Username</td>
    <td>Nama</td>
    <td>ID Level Pengguna</td>
    <td>Kode Level</td>
    <td>Nama Level</td>
    <td>Aksi</td>
  </tr>
  @foreach ($data as $d)
    <tr>
      <td>{{ $d->user_id }}</td>
      <td>{{ $d->username }}</td>
      <td>{{ $d->nama }}</td>
      <td>{{ $d->level_id }}</td>
      <td>{{ $d->level->level_kode }}</td>
      <td>{{ $d->level->level_nama }}</td>
      <td><a href="/user/ubah/{{ $d->user_id }}">Ubah</a> | <a href="/user/hapus/{{ $d->user_id }}">Hapus</a></td>
    </tr>
  @endforeach
</table>
</body>
```

- Simpan kode program Langkah 4 dan 5. Kemudian jalankan link pada browser, kemudian amati apa yang terjadi dan beri penjelasan dalam laporan

Jawab :



## Data User

[+ Tambah User](#)

ID	Username	Nama	ID Level Pengguna	Kode Level	Nama Level	Aksi
1	admin	Administrator	1	ADM	Administrator	<a href="#">Ubah</a>   <a href="#">Hapus</a>
2	manager	Manager	2	MNG	Manager	<a href="#">Ubah</a>   <a href="#">Hapus</a>
3	staff	Staff/Kasir	3	STF	Staff/Kasir	<a href="#">Ubah</a>   <a href="#">Hapus</a>
4	customer-1	Pelanggan Pertama	4	CUS	Customer	<a href="#">Ubah</a>   <a href="#">Hapus</a>
9	manager_dua	MANager 2	2	MNG	Manager	<a href="#">Ubah</a>   <a href="#">Hapus</a>
12	manager22	Manager Dua Dua	2	MNG	Manager	<a href="#">Ubah</a>   <a href="#">Hapus</a>
13	manager33	Manager Tiga Tiga	2	MNG	Manager	<a href="#">Ubah</a>   <a href="#">Hapus</a>
14	manager56	Manager55	2	MNG	Manager	<a href="#">Ubah</a>   <a href="#">Hapus</a>
16	manager12	Manager11	2	MNG	Manager	<a href="#">Ubah</a>   <a href="#">Hapus</a>

Di UserController method index untuk mengambil semua data user dengan relasi level yang menggunakan with('level') dan mengirimkan data ke view. View menampilkan tabel daftar user dan informasi tabel level.

- Laporkan hasil Praktikum-2.7 ini dan commit perubahan pada git.

Jawab :

