

LAPORAN JOBSHEET 4

MODEL & ELOQUENT ORM

MATA KULIAH PEMROGRAMAN WEB LANJUT

Dosen Pengampu : Dimas Wahyu Wibowo, S.T., M.T.



Disusun oleh :

Dahniar Davina SIB-2A / 2341760023

JURUSAN TEKNOLOGI INFORMASI

PROGRAM STUDI D-IV SISTEM INFORMASI BISNIS

POLITEKNIK NEGERI MALANG

2025

A. PRAKTIKUM 1 : \$fillable

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

```
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

```
public function index()
{
    // tambah data user dengan Eloquent Model
    $data = [
        'level_id' => 2,
        'username' => 'manager_dua',
        'nama' => 'Manager 2',
        'password' => Hash::make('12345')
    ];

    UserModel::insert($data); // tambahkan data ke tabel m_u

    // coba akses model UserModel
    $user = UserModel::all(); // ambil semua data dari tabel
    return view('user', ['data' => $user]);
}
```

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

ID	USERNAME	NAMA	LEVEL
6	customer-1	Pelanggan	5
23	manager_dua	Manager 2	2

☐ Edit ☐ Copy ☐ Delete 23 2 manager_dua Manager 2 \$2y\$12\$qBfZSqsMdO.nCA3Q2V

4. Ubah file model UserModel.php seperti pada gambar di bawah ini pada bagian \$fillable







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

5. Ubah kembali file controller UserController.php seperti pada gambar di bawah hanya bagian array pada \$data

```
// tambah data user dengan Eloquent Model
$data = [
    'level_id' => 2,
    'username' => 'manager_tiga',
    'nama' => 'Manager 3',
    'password' => Hash::make('12345')
];
```

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

23	manager_dua	Manager 2	2
24	manager_tiga	Manager 3	2

 Edit	 Copy	 Delete	23	2	manager_dua	Manager 2	\$2y\$12\$qBfZ!
 Edit	 Copy	 Delete	24	2	manager_tiga	Manager 3	\$2y\$12\$byLh

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

B. PRAKTIKUM 2.1 : Retrieving Single Models

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

```
$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

```
<td>{{ $data->user_id }}</td>
<td>{{ $data->username }}</td>
<td>{{ $data->nama }}</td>
<td>{{ $data->level_id }}</td>
```

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

Data User

ID	Username	Nama	ID Level Pengguna
1	admin	Administrator	1

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

```
$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

Data User

ID	Username	Nama	ID Level Pengguna
1	admin	Administrator	1

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

```
$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

Data User

ID	Username	Nama	ID Level Pengguna
1	admin	Administrator	1

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

```
$user = UserModel::findOrFail(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

Data User

ID	Username	Nama	ID Level Pengguna
	admin	Administrator	


➤ \$user = UserModel::findOrFail(1, ['username', 'nama'], function () {abort(404);

hanya memanggil dan menampilkan username dan nama.

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

```
$user = UserModel::findOrFail(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



➤ Error, karena tidak ada data username dan nama dengan ID 20.

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

C. Praktikum 2.2 : Not Found Exceptions

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

```
$user = UserModel::findOrFail(1); // a
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

Data User


ID	Username	Nama	ID Level Pengguna
1	admin	Administrator	1

➤ Kode diminta untuk memanggil dan menampilkan data dengan ID 1, jika data dengan ID 1 tidak ada, maka browser akan menunjukkan error
404|not found

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

```
$user = UserModel::where('username', 'manager9')->firstOrFail();|
return view('user', ['data' => $user]);
```

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



➤ Error, karena tidak ada nama 'manager9' pada database, sehingga data yang diminta tidak muncul.

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

D. PRAKTIKUM 2.3 : Retrieving Aggregates

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

```
$user = UserModel::where('level_id', 2)->count();  
dd($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

```
3 // app\Http\Controllers\UserController.php:36
```

➤ Return view(...) tidak akan dijalankan karena dd(\$user); sudah menghentikan eksekusi sebelumnya.

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

```
$jumlahPengguna = UserModel::where('level_id', 2)->count();  
return view('user', ['jumlahPengguna' => $jumlahPengguna]);
```

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

Data User

Jumlah Pengguna

3

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

E. PRAKTIKUM 2.4 : Retrieving or Creating Models

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

```
$user = UserModel::firstOrCreate([
    'username' => 'manager',
    'nama' => 'Manager',
]);
return view('user', ['data' => $user]);
```

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

```
<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>
```

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

Data User

ID	Username	Nama	ID Level Pengguna
2	manager	Manager	2

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

```
$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

Data User

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

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

```
$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

Data User

ID	Username	Nama	ID Level Pengguna
2	manager	Manager	2

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

```
$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

Data User

ID	Username	Nama	ID Level Pengguna
	manager33	Manager Tiga Tiga	2

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

```
$user = UserModel::firstOrCreate([
    'username' => 'manager33',
    'nama' => 'Manager Tiga Tiga',
    'password' => Hash::make('12345'),
    'level_id' => 2
]);
$user->save();
return view('user', ['data' => $user]);
```

11. 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

Data User

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

<input type="checkbox"/>	 Edit	 Copy	 Delete	23	2	manager_dua	Manager 2	\$2y\$12\$qBf
<input type="checkbox"/>	 Edit	 Copy	 Delete	24	2	manager_tiga	Manager 3	\$2y\$12\$byL
<input type="checkbox"/>	 Edit	 Copy	 Delete	26	2	manager22	Manager Dua Dua	\$2y\$12\$nxvl
<input type="checkbox"/>	 Edit	 Copy	 Delete	27	2	manager33	Manager Tiga Tiga	\$2y\$12\$nbrl

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

F. PRAKTIKUM 2.5 : Attribute Changes

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

```
$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());

```

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

```

false // app\Http\Controllers\UserController.php:101

```

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

```

$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

```

true // app\Http\Controllers\UserController.php:118

```

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

G. PRAKTIKUM 2.6 : Create, Read, Update, Delete (CRUD)

1. Buka file view pada user.blade.php dan buat scripnya menjadi seperti di bawah ini

```
<html>
<head>
  <title>Data User</title>
</head>
<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>
</html>
```

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

```
//CRUD
$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

Data User

[+ Tambah User](#)

ID	Username	Nama	ID Level Pengguna	Aksi
1	admin	Administrator	1	Ubah Hapus
2	manager	Manager	2	Ubah Hapus
3	staff	Staff/Kasir	3	Ubah Hapus
6	customer-1	Pelanggan	5	Ubah Hapus
23	manager_dua	Manager 2	2	Ubah Hapus
24	manager_tiga	Manager 3	2	Ubah Hapus
26	manager22	Manager Dua Dua	2	Ubah Hapus
27	manager33	Manager Tiga Tiga	2	Ubah Hapus
28	manager56	Manager55	2	Ubah Hapus
29	manager12	Manager11	2	Ubah Hapus

- 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

```
<!DOCTYPE html>
<html>
<head>
  <title>Data User</title>
</head>
<body>
  <h1>Form Tambah Data User</h1>
  <form method="post" action="/user/tambah_simpan">
    {{ csrf_field() }}

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

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

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

- 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

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

- 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

Form Tambah Data User

Username	<input type="text" value="Masukkan Username"/>
Nama	<input type="text" value="Masukkan Nama"/>
Password	<input type="password" value="Masukkan Password"/>
Level ID	<input type="text" value="Masukkan ID Level"/>
<input type="button" value="Simpan"/>	

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 diletakan dibawah 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

Form Tambah Data User

Username
Nama
Password
Level ID

29	manager12	Manager11	2	Ubah Hapus
30	Admin2	Admin 99	1	Ubah Hapus

<input type="checkbox"/>	Edit	Copy	Delete	30	1 Admin2	Admin 99	\$2y\$12\$26Hw3E2'
--------------------------	----------------------	----------------------	------------------------	----	----------	----------	--------------------

➤ Ketika form tambah diisi, data akan terkirim dan tampil pada database.

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

```

<!DOCTYPE html>
<html>
<head>
  <title>Data User</title>
</head>
<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="Masukkan Username" value="{{ $data->username }}">
    <br>
    <label>Nama</label>
    <input type="text" name="nama" placeholder="Masukkan Nama" value="{{ $data->nama }}">
    <br>
    <label>Password</label>
    <input type="password" name="password" placeholder="Masukkan Password" value="{{ $data->password }}">
    <br>
    <label>Level ID</label>
    <input type="number" name="level_id" placeholder="Masukkan ID Level" value="{{ $data->level_id }}">
    <br><br>
    <input type="submit" class="btn btn-success" value="Ubah">

  </form>
</body>
</html>

```

12. Tambahkan script pada routes dengan nama file web.php. Tambahkan seperti gambar dibawah 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

Form Ubah Data User

[Kembali](#)

> Username

Nama

Password

Level ID

15. Tambahkan script pada routes dengan nama file web.php. Tambahkan seperti gambar dibawah 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 dibawah 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 atau 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

Form Ubah Data User

[Kembali](#)

> Username
Nama
Password
Level ID

27	manager12	manager11	1	Ubah hapus
30	Admin2	Admin 199	1	Ubah Hapus

☐  Edit  Copy  Delete 30 1 Admin2 Admin 199 \$2y\$12\$NjVcJWsl'

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 diletakkan 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

28	manager56	Manager55	2	Ubah Hapus
29	manager12	Manager11	2	Ubah Hapus

Data dengan username "Admin 199" telah hilang dari database

 Edit  Copy  Delete 29 2 manager12 Manager11 \$2y\$12\$h2pum

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

H. PRAKTIKUM 2.7 : Relations

1. Buka file model pada UserModel.php dan tambahkan scripnya menjadi seperti di bawah ini

```
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

```
<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Model;
use Illuminate\Database\Eloquent\Relations\HasMany;

class LevelModel extends Model
{
    protected $table = 'm_level';
    protected $primaryKey = 'level_id';
```

Menambah file
"LevelModels"
pada folder Model


```

        protected $fillable = ['level_kode', 'level_name']; //Foreign key

        public function users(): HasMany
        {
            return $this->hasMany(UserModel::class, 'level_id', 'level_id');
        }
    }

```

```

    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

```

Illuminate\Database\Eloquent\Collection {#320 ▼ // app\Http\Controllers\UserController.php:172
  #items: array:10 [▶]
  #escapeWhenCastingToString: false
}

```

4. 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]);
    }

```

5. Buka file view pada user.blade.php dan ubah script menjadi seperti di bawah ini
6. Simpan kode program Langkah 4 dan 5. Kemudian jalankan link pada browser, kemudian amati apa yang terjadi dan beri penjelasan dalam laporan

```

<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/ubah/{{ $d->user_id }}">Hapus</a></td>
    </tr>
@endforeach

```

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