Pemrograman Web Lanjut
Jobsheet 4 - Model dan Eloquent Orm

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

Nama : Athallah Ayudya Paramesti
NIM : 2341760061
Program Studi : D-IV Sistem Informasi Bisnis

Jurusan Teknologi Informasi
Politeknik Negeri Malang
2025

A. PROPERTI `$fillable` DAN `$guarded`

Praktikum 1 - $fillable

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

```php
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

```php
<?php

namespace App\Http\Controllers;

use App\Models\UserModel;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Hash;

class UserController extends Controller
{
    public function index(){

        $data = [
            'level_id' => 2,
            'username' => 'manager_dua',
            'nama' => 'Manager 2',
            'password' => Hash::make('12345')
        ];
        UserModel::create($data);

        $user = UserModel::all();
        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

Data User

ID	Username	Nama	ID Level Pengguna
1		Administrator	1
2		Manager	2
3		Staff/Kasir	3
6		Pelanggan Pertama	4
7		Manager 2	2

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

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

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

```php
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]);
}
```

6. Simpan kode program Langkah 4 dan 5. Kemudian jalankan pada browser dan amati apa yang terjadi
   Jawab:
   Terjadi eror karena **`password` tidak ada di `$fillable`**, sistem **mengabaikannya** saat insert, menyebabkan error **"Field 'password' doesn't have a default value"**.
7. Laporkan hasil Praktikum-1 ini dan *commit* perubahan pada git.

B. RETRIEVING SINGLE MODELS

Praktikum 2.1 - *Retrieving Single Models*

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

```php
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

```blade
        <tr>
            <td>{{ $data->user_id }}</td>
            <td>{{ $data->Username }}</td>
            <td>{{ $data->nama }}</td>
            <td>{{ $data->level_id}}</td>
        </tr>
    </table>
```

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		Administrator	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

Data User

ID	Username	Nama	ID Level Pengguna
1		Administrator	1

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

Data User

ID	Username	Nama	ID Level Pengguna
1		Administrator	1

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

```php
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

Data User

ID	Username	Nama	ID Level Pengguna
1		Administrator	1

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

```php
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

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

Praktikum 2.2 – *Not Found Exceptions*

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

```php
class UserController extends Controller
{
    public function index(){

        $user = UserModel::findOrFail(1);
        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

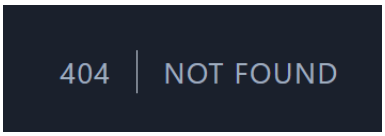
ID	Username	Nama	ID Level Pengguna
1		Administrator	1

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

```php
class UserController extends Controller
{
    public function index(){

        $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



not found karena data tidak ditemukan di dalam database

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

Praktikum 2.3 – *Retreiving Aggregrates*

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

```php
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

```
2 // app\Http\Controllers\UserController.php:14
```

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

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

```php
<!DOCTYPE html>
<html>
<head>
    <title>Data User</title>
</head>
<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>
</html>
```

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

Praktikum 2.4 – *Retreiving or Creating Models*

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

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

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

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

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	2

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

```
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

Data User

ID	Username	Nama	ID Level Pengguna
8		Manager Dua Dua	2

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

```
$user = UserModel::firstOrNew(
    [
        '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	2

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

```php
$user = UserModel::firstOrNew(
    [
        '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
		Manager Tiga Tiga	2

←T→	user_id	level_id	username	nama
Edit Copy Delete	1	1	admin	Administrator
Edit Copy Delete	2	2	manager	Manager
Edit Copy Delete	3	3	staff	Staff/Kasir
Edit Copy Delete	6	4	customer-1	Pelanggan Pertama
Edit Copy Delete	7	2	manager_dua	Manager 2
Edit Copy Delete	8	2	manager22	Manager Dua Dua

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

```php
    ],
);
$user->save();
```

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
9		Manager Tiga Tiga	2

user_id	level_id	username	nama
1	1	admin	Administrator
2	2	manager	Manager
3	3	staff	Staff/Kasir
6	4	customer-1	Pelanggan Pertama
7	2	manager_dua	Manager 2
8	2	manager22	Manager Dua Dua
9	2	manager33	Manager Tiga Tiga

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

Praktikum 2.5 – *Attribute Changes*

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

```php
public function index(){

    $user = UserModel::create(
        [
            'username' => 'manager55',
            'nama' => 'Manager55',
            'password' => Hash::make('12345'),
            'level_id' => 2,
        ]
    );

    $user->username = 'manager56';

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

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

    $user->save();

    $user->isDirty();
    $user->isClean();
    dd($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:38
```

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

```php
public function index(){

    $user = UserModel::create(
        [
            'username' => 'manager11',
            'nama' => 'Manager11',
            'password' => Hash::make('12345'),
            'level_id' => 2,
        ]
    );

    $user->username = 'manager12';

    $user->save();

    $user->wasChanged();
    $user->wasChanged('username');
    $user->wasChanged(['username', 'level_id']);
    $user->wasChanged('nama');
    $user->wasChanged(['nama', 'username']);
}
```

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

kosong, karena true

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

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

1. Buka file view pada `user.blade.php` dan buat scritpnya 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

Data User

+ Tambah User

ID	Username	Nama	ID Level Pengguna	Aksi
1		Administrator	1	Ubah | Hapus
2		Manager	2	Ubah | Hapus
3		Staff/Kasir	3	Ubah | Hapus
6		Pelanggan Pertama	4	Ubah | Hapus
7		Manager 2	2	Ubah | Hapus
8		Manager Dua Dua	2	Ubah | Hapus
9		Manager Tiga Tiga	2	Ubah | Hapus
10		Manager55	2	Ubah | Hapus
11		Manager11	2	Ubah | Hapus

4. Langkah berikutnya membuat create atau tambah data user dengan cara membuat 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="Masukkan Username">
    <br>
    <label>Nama</label>
    <input type="text" name="name" placeholder="Masukkan Nama">
    <br>
    <label>Password</label>
    <input type="text" name="username" placeholder="Masukkan Password">
    <br>
    <label>Level ID</label>
    <input type="number" name="username" placeholder="Masukkan ID Level">
    <br><br>
    <input type="submit" class="btn  btn-success" value="Simpan">
    </form>
</body>
```

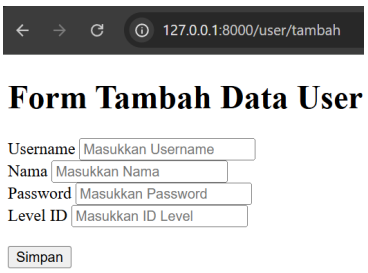
5. Tambahkan script pada routes dengan nama file `web.php`. Tambahkan seperti gambar di bawah ini

```
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

```
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



ketika + Tambah User ditekan, akan langsung diarahkan ke halaman tambah data user.

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

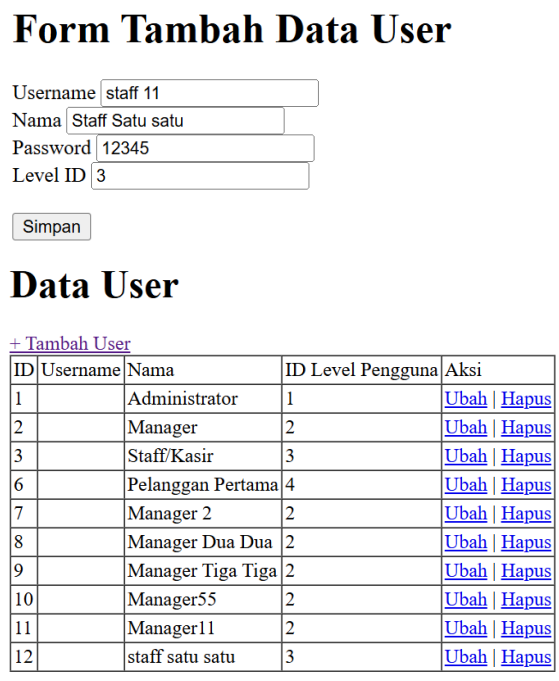
```
Route::post('/user/tambah_simpan', [UserController::class, 'tambah_simpan'
```

9. Tambahkan *script* pada controller dengan nama file `UserController.php`. Tambahkan *script* dalam class dan buat method baru dengan nama tambah_simpan dan diletakan 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



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

```
<form action="/user/ubah_simpan{{ $data->user_id }}" method="post">
    {{ 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>
```

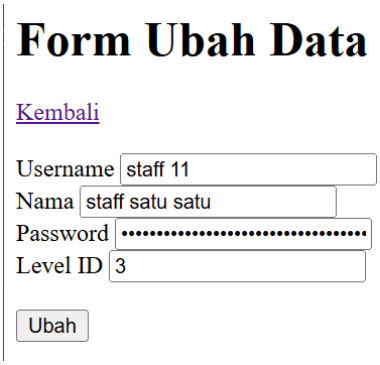
12. Tambahkan *script* pada *routes* dengan nama file `web.php`. Tambahkan seperti gambar di bawah ini

```
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



form untuk mengubah data user

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

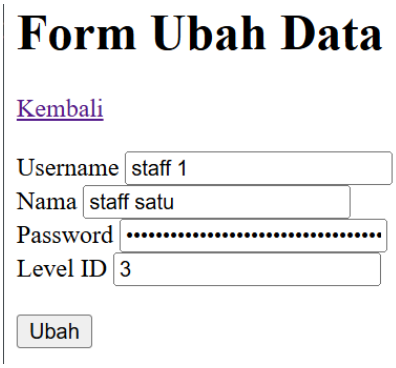
```php
public function ubah_simpan($id, Request $request){
    $user = UserModel::find($id);

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



18. Berikut untuk langkah *delete* . Tambahkan *script* pada *routes* dengan nama file `web.php`. Tambahkan seperti gambar di bawah ini

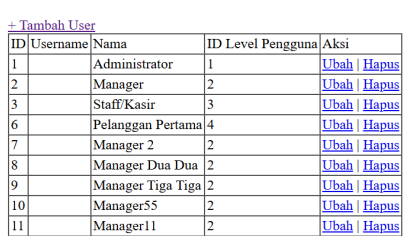
```php
Route::get('/user/hapus/{id}', [UserController::class, 'hapus']);
```

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

```php
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



Data User

+ Tambah User

ID	Username	Nama	ID Level Pengguna	Aksi
1		Administrator	1	Ubah | Hapus
2		Manager	2	Ubah | Hapus
3		Staff/Kasir	3	Ubah | Hapus
6		Pelanggan Pertama	4	Ubah | Hapus
7		Manager 2	2	Ubah | Hapus
8		Manager Dua Dua	2	Ubah | Hapus
9		Manager Tiga Tiga	2	Ubah | Hapus
10		Manager55	2	Ubah | Hapus
11		Manager11	2	Ubah | Hapus

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

Praktikum 2.7 – *Relationships*

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

```
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

```
Illuminate\Database\Eloquent\Collection {#318 ▼ // app\Http\Controllers\UserController.php:14
  #items: array:9 [▶]
  #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

```
<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->leve_nama }}</td>
    <td><a href="/user/ubah/{{ $d->user_i
</tr>
```

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

Data User

+ Tambah User

ID	Username	Nama	ID Level Pengguna	Kode Level	Nama Level	Aksi
1		Administrator	1	ADM		Ubah | Hapus
2		Manager	2	MNG		Ubah | Hapus
3		Staff/Kasir	3	STF		Ubah | Hapus
6		Pelanggan Pertama	4	CUS		Ubah | Hapus
7		Manager 2	2	MNG		Ubah | Hapus
8		Manager Dua Dua	2	MNG		Ubah | Hapus
9		Manager Tiga Tiga	2	MNG		Ubah | Hapus
10		Manager55	2	MNG		Ubah | Hapus
11		Manager11	2	MNG		Ubah | Hapus

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