

**TUGAS**  
**Web Lanjut**

**PERTEMUAN 4**

Olequent ORM

**OLEH**

**MUHAMMAD NUR AZIZ NIM. 2341720237**



**PROGRAM STUDI D-IV TEKNIK INFORMATIKA**  
**JURUSAN TEKNOLOGI INFORMASI**  
**POLITEKNIK NEGERI MALANG**  
**3 Maret 2025**

## A. Properti \$fillable dan \$guarded

### Praktikum-1

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

```
class UserModel extends Model
{
    //praktikum week 3
    use HasFactory;

    protected $table = 'm_user'; // mendefinisikan tabel yang akan digunakan
    protected $primaryKey = 'user_id'; // mendefinisikan primary key

    //praktikum week 4
    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

```
// Tambah Data user dengan Eloquent Model
$data = [
    'level_id' => 2,
    'username' => 'manager_dua',
    'nama' => 'Manager 2',
    // 'nama' => 'Pelanggan Pertama', // week 3
    'password' => Hash::make('12345'),
];
UserModel::create($data);

// UserModel::where('username', 'customer-1')->update($data); //update
// akses model UserModel
$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	admin	Administrator	1
2	manager	Manager	2
3	staff	Staff/Kasir	3
9	customer-1	Pelanggan Pertama	4
10	manager_dua	Manager 2	2

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

```
//praktikum week 4
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
$data = [
    'level_id' => 2,
    'username' => 'manager_tiga',
    'nama' => 'Manager 3',
    // 'nama' => 'Pelanggan Pertama', // week 3
    'password' => Hash::make('12345'),
];
UserModel::create($data);

// UserModel::where('username', 'customer-1')->update($data); //update
// $user = UserModel::find($id);
```

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

```
Illuminate \ Database \ QueryException PHP 8.1.10 10.48.28

SQLSTATE[HY000]: General error: 1364 Field 'password' doesn't have a default
value

INSERT INTO `m_user` (`level_id`, `username`, `nama`, `updated_at`, `created_at`) VALUES
```

Error, field password tidak ditemukan

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

```
// week-4 prak 2.1
$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

```
{{-- week 4 --}}
<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>
```

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

- Mencari data User dengan ID 1
4. Ubah file controller dengan nama UserController.php dan ubah script seperti gambar di bawah ini

```
// $user = UserModel::find(1);
$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

- Menampilkan data user dengan level\_id 1
6. Ubah file controller dengan nama UserController.php dan ubah script seperti gambar di bawah ini

```
// $user = UserModel::where('level_id', 1)->first();  
$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

- Menampilkan data user dengan level\_id 1
8. Ubah file controller dengan nama UserController.php dan ubah script seperti gambar di bawah ini

```
$user = UserModel::findOr(1,['username','nama'], function(){  
    abort(404);  
});
```

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

## Data User

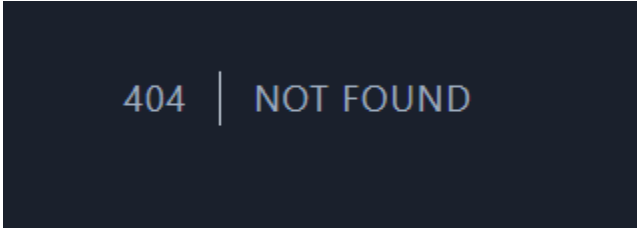
ID	Username	Nama	ID Level Pengguna
	admin	Administrator	

Menampilkan data dengan ID 1 kemudian hanya ditampilkan nama dan username, jika data tidak ditemukan maka akan menjalankan function abort(404)

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

```
$user = UserModel::findOr(20,['username','nama'], function(){  
    abort(404);  
});
```

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



404 | NOT FOUND

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

```
// week-4 prak 2.2  
$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

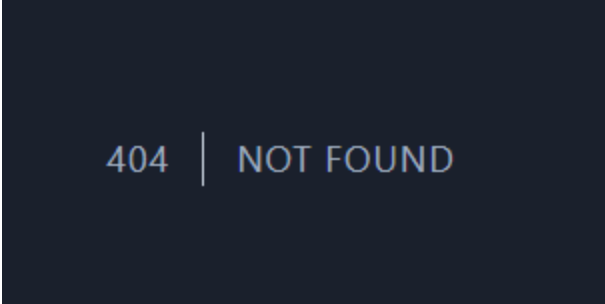
ID	Username	Nama	ID Level Pengguna
1	admin	Administrator	1

Menampilkan data dengan ID 1

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

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



404 | NOT FOUND

Menampilkan error not found karena data tidak ditemukan dengan menggunakan firstOrFail


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

### Praktikum 2.3 – Retrieving Aggregates

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

```
// week-4 prak 2.3 Retrieving Aggregates
$user = UserModel::where('level_id', 2)->count();
dd($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:54
```

Menghitung total user dengan level ID 2, kemudian dd(\$user) menampilkan lokasi 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

```
// week-4 prak 2.3 Retrieving Aggregates
$user = UserModel::where('level_id', 2)->count();
// dd($user);
return view('user', ['data' => $user]);

{{-- week 4 prak 2.3 --}}
<tr>
    <th>jumlah pengguna</th>
</tr>
<tr>
    <td>{{ $data }}</td>
</tr>
</table>
```

## Data User

jumlah pengguna
3

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

### Praktikum 2.4 – Retrieving or Creating Models

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

```
// week-4 prak 2.4 Retrieving or Creating Models
$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



```

{{-- week 4 prak 2.4 --}}
<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	Manager	2

Menampilkan data dengan username manager dan nama manager, jika belum ditemukan maka akan membuat data baru jika sesuai dengan field.

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

```

// week-4 prak 2.4 Retrieving or Creating Models
$user = UserModel::firstOrCreate([
    // 'username' => 'manager',
    // 'nama' => 'Manager',
    '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
12	manager22	Manager Dua Dua	2

Membuat user baru sesuai dengan parameter yang diberikan

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

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

- 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

Menampilkan data user manager

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

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

- 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

		user_id	level_id	username	nama	password
<input type="checkbox"/>	Edit Copy Delete	1	1	admin	Administrator	\$2y\$12\$.c7TKpSGy
<input type="checkbox"/>	Edit Copy Delete	2	2	manager	Manager	\$2y\$12\$lv.kWhUVF
<input type="checkbox"/>	Edit Copy Delete	3	3	staff	Staff/Kasir	\$2y\$12\$G5FM2wA
<input type="checkbox"/>	Edit Copy Delete	9	4	customer-1	Pelanggan Pertama	\$2y\$12\$sRlfB7YbF
<input type="checkbox"/>	Edit Copy Delete	10	2	manager_dua	Manager 2	\$2y\$12\$2oE34LjNb
<input type="checkbox"/>	Edit Copy Delete	11	2	manager_tiga	Manager 3	\$2y\$12\$7NFjZGloN
<input type="checkbox"/>	Edit Copy Delete	12	2	manager22	Manager Dua Dua	\$2y\$12\$uvmACUD
<input type="checkbox"/>	Check all	With selected: Edit Copy Delete Export				

Menampilkan data yang baru dibuat, akan tetapi data tidak masuk didalam phpmyadmin

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

```
// 173
// return view('user', ['data' => $user]);
$user = UserModel::firstOrCreate([
    // 'username' => 'manager',
    // 'nama' => 'Manager',
    '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
13	manager33	Manager Tiga Tiga	2

Menampilkan data yang baru dibuat dan disimpan ke dalam database

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

```
// week 4 Praktikum 2.5 - Attribute Changes
$user = UserModel::create([
    'username' => 'manager55',
    'nama' => 'Manager55',
    'password' => Hash::make('12345'),
    'level_id' => 2
]);
$user->username = 'manager55';

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

$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:100
```

Menampilkan hasil kode apakah sudah pernah diubah atau belum

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

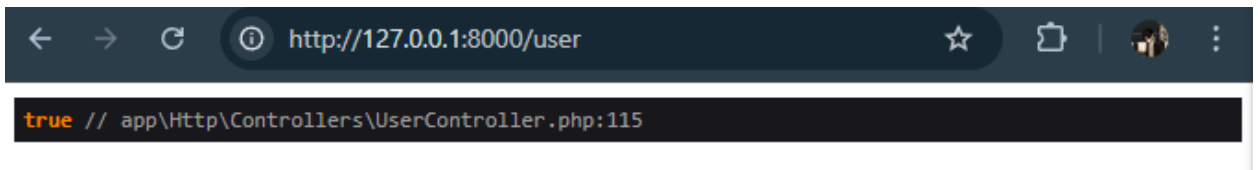
```

$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



Menampilkan true, karena data pernah dirubah pada bagian username

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 scriptnya menjadi seperti di bawah ini

```

{{-- week 4 prak 2.6 --}}
<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>
</foreach>

```

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

```
// Praktikum 2.6 - Create, Read, Update, Delete (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	<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>
9	customer-1	Pelanggan Pertama	4	<a href="#">Ubah</a>   <a href="#">Hapus</a>
10	manager_dua	Manager 2	2	<a href="#">Ubah</a>   <a href="#">Hapus</a>
11	manager_tiga	Manager 3	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	manager55	Manager55	2	<a href="#">Ubah</a>   <a href="#">Hapus</a>
15	manager12	Manager11	2	<a href="#">Ubah</a>   <a href="#">Hapus</a>

Menampilkan data semua user dan terdapat tombol aksi untuk tambah, ubah dan hapus, tetapi belum berfungsi

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>
  {{-- Week 4 prak 2.6 --}}
  <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

```

// week 4 Praktikum 2.6 - Create, Read, Update, Delete (CRUD)
Codeium: Refactor | Explain | X
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

**Form Tambah Data User**

Username

Nama

Password

Level ID

Ketika kita mengklik tambah user pada halaman user maka akan diarahkan ke halaman form tambah user

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

```
Route::get('/user', [UserController::class, 'index']);  
// week 4 praktikum 2.6  
Route::get('/user/tambah', [UserController::class, 'tambah']);  
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

```
}  
// week 4 Praktikum 2.6  
Codeium: Refactor | Explain | X  
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

---

## 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>
9	customer-1	Pelanggan Pertama	4	<a href="#">Ubah</a>   <a href="#">Hapus</a>
10	manager_dua	Manager 2	2	<a href="#">Ubah</a>   <a href="#">Hapus</a>
11	manager_tiga	Manager 3	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	manager55	Manager55	2	<a href="#">Ubah</a>   <a href="#">Hapus</a>
15	manager12	Manager11	2	<a href="#">Ubah</a>   <a href="#">Hapus</a>
16	ren	renhwa	1	<a href="#">Ubah</a>   <a href="#">Hapus</a>

Setelah mengisi form tambah, data nanti akan disimpan dan diarahkan ke halaman data user

- 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

```

<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->nama }}">
    <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>

```

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 diletakkan di bawah method tambah\_simpan seperti gambar di bawah ini

```

// week 4 Praktikum 2.6
Codeium: Refactor | Explain | X
public function ubah($id)
{
    $user = UserModel::find($id);
    return view('user_ubah', ['data' => $user]);
}
// week 4 Praktikum 2.6
Codeium: Refactor | Explain | X

```

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

Menampilkan form ubah user dan belum bisa disimpan

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 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

Sebelum di ubah

16	ren	renhwaaa	1	<a href="#">Ubah</a>   <a href="#">Hapus</a>
----	-----	----------	---	--

Proses ubah

## Form Ubah Data User

[Kembali](#)

Username

Nama

Password

Level ID

Setelah diubah

16	ren	renhwa	1	<a href="#">Ubah</a>   <a href="#">Hapus</a>
----	-----	--------	---	--

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

```
// week 4 Praktikum 2.6
Codeium: Refactor | Explain | X
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

16	ren	renhwa	1	<a href="#">Ubah</a>   <a href="#">Hapus</a>
11	manager_tiga	Manager 3	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	manager55	Manager55	2	<a href="#">Ubah</a>   <a href="#">Hapus</a>
15	manager12	Manager11	2	<a href="#">Ubah</a>   <a href="#">Hapus</a>

User ren langsung terhapus ketika dihapus

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

## Praktikum 2.7 – Relationships

Buat file level model dulu

```

Codeium: Refactor | Explain
class LevelModel extends Model
{
    use HasFactory;
    protected $table = 'level';
    protected $primaryKey = 'level_id';
    Codeium: Refactor | Explain | Generate Function Comment | X
    public function user(): BelongsTo
    {
        return $this->belongsTo(UserModel::class);
    }
}

```

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

```

Codeium: Refactor | Explain | Generate Function Comment | X
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

```

// Praktikum 2.7 - Relationships
$user = UserModel::with('level')->get();
dd($user);
// week 4 Praktikum 2.6 - Create Read Update D

```

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:123
  #items: array:10 [▶]
  #escapeWhenCastingToString: false
}

```

Menampilkan data one to many dengan belongsto

4. Buka file controller pada UserController.php dan ubah method script menjadi seperti di bawah ini

```

// Praktikum 2.7 - Relationships
$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

```

{{-- week 4 prak 2.7 --}}
<td>Kode Level</td>
<td>Nama Level</td>

{{-- week 4 prak 2.6 --}}
<td>ID Level Pengguna</td>
<td>Aksi</td>
</tr>
<tr>
  <td colspan="7">
    {{#each $data as $d}}
      <tr>
        <td>{{ $d->user_id }}</td>
        <td>{{ $d->username }}</td>
        <td>{{ $d->nama }}</td>
        <td>{{ $d->level_id }}</td>

        {{-- week 4 prak 2.7 --}}
        <td>{{ $d->level->level_kode }}</td>
        <td>{{ $d->level->level_nama }}</td>

        {{-- week 4 prak 2.6 --}}
        <td><a href="/user/ubah/{{ $d->user_id }}">Ubah</a> |
          <a href="/user/hapus/{{ $d->user_id }}">Hapus</a>
        </td>
      </tr>
    </tr>
  </td>
</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	Kode Level	Nama Level	ID Level Pengguna	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>
9	customer-1	Pelanggan Pertama	4	CUS	pelanggan	<a href="#">Ubah</a>   <a href="#">Hapus</a>
10	manager_dua	Manager 2	2	MNG	Manager	<a href="#">Ubah</a>   <a href="#">Hapus</a>
11	manager_tiga	Manager 3	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	manager55	Manager55	2	MNG	Manager	<a href="#">Ubah</a>   <a href="#">Hapus</a>
15	manager12	Manager11	2	MNG	Manager	<a href="#">Ubah</a>   <a href="#">Hapus</a>

menampilkan seluruh data user dengan tambahan kolom kode level

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