

PEMROGRAMAN WEB LANJUT

JOBSHEET 4

MODEL dan ELOQUENT ORM

I. PROPERTI \$fillable dan \$guarded

A. \$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'];
}
```

⇒ Source code

```
8  class UserModel extends Model
9  {
10     use HasFactory;
11
12     protected $table = 'm_user'; // Mendefinisikan nama tabel yang digunakan di
13     protected $primaryKey = 'user_id'; // Mendefinisikan primary key dari tabel
14     /**
15      * The attributes that are mass assignable
16      *
17      * @var array
18      */
19
20     protected $fillable = ['level_id', 'username', 'nama', 'password'];
21
22 }
```

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

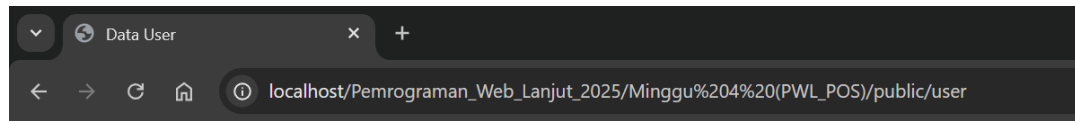
```
1  <?php
2
3  namespace App\Http\Controllers;
4
5  use Illuminate\Http\Request;
6  use App\Models\UserModel;
7  use Illuminate\Support\Facades\Hash;
8
9  class UserController extends Controller
10 {
11     public function index()
12     {
13         $data = [
14             'level_id' => 2,
15  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

⇒ source code

```
Minggu 4 (PWL_POS) > app > Http > Controllers > UserController.php > UserController > index
1  <?php
2
3  namespace App\Http\Controllers;
4
5  use App\Models\UserModel;
6  use Illuminate\Http\Request;
7  use Illuminate\Support\Facades\Hash;
8
9  class UserController extends Controller
10 {
11     public function index() {
12
13
14         $data = [
15             'level_id' => 2,
16             'username' => 'manager_dua',
17             'nama' => 'Manager 2',
18             'password' => Hash::make('12345')
19         ];
20         UserModel::create($data);
21
22         $user = UserModel::all();
23         return view('user', ['data' => $user]);
24     }
25 }
26
```

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

⇒ Halaman Browser



Data User

ID	Username	Nama	ID Level Pengguna
1	admin	Administrator	1
2	manager	Manager	3
3	staff	Staff/Kasir	3
10	customer-1	Pelanggan Pertama	4
11	manager_dua	Manager 2	2

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

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

⇒ Source code

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

5. 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]);
}
```

⇒ Source code

```
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]);
}
```

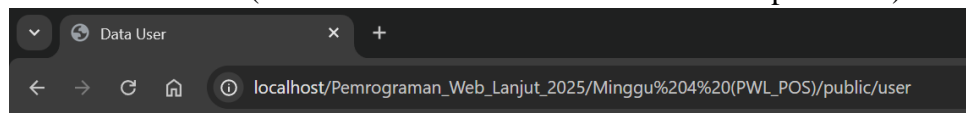
6. Simpan kode program Langkah 4 dan 5. Kemudian jalankan pada *browser* dan amati apa yang terjadi

⇒ Halaman Browser



Terjadi error, hal ini karena sebelumnya UserModel sebelumnya telah dimodifikasi yang mana menghapus kolom password, sehingga ketika ingin menambahkan data baru kolom yang ada kurang dan terjadi error. Untuk menjalankannya harus terlebih dahulu mengembalikan kolom password di UserModel.

Halaman Browser (setelah UserModel ditambahkan kolom password)



Data User

ID	Username	Nama	ID Level Pengguna
1	admin	Administrator	1
2	manager	Manager	3
3	staff	Staff/Kasir	3
10	customer-1	Pelanggan Pertama	4
11	manager_dua	Manager 2	2
12	manager_tiga	Manager 3	2

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

B. \$guarded

Kebalikan dari *\$fillable* adalah *\$guarded*. Semua kolom yang kita tambahkan ke *\$guarded* akan diabaikan oleh Eloquent ketika kita melakukan insert/update. Secara default *\$guarded* isinya `array("*")`, yang berarti semua atribut tidak bisa diset melalui *mass assignment*. *Mass Assignment* adalah fitur canggih yang menyederhanakan proses pengaturan beberapa atribut model sekaligus, menghemat waktu dan tenaga. Pada praktikum ini, kita akan mengeksplorasi konsep penugasan massal di Laravel dan bagaimana hal itu dapat dimanfaatkan secara efektif untuk meningkatkan alur kerja pengembangan Anda.

II. RETRIEVING SINGLE MODELS

A. 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]);
    }
}
```

⇒ Source code

```
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

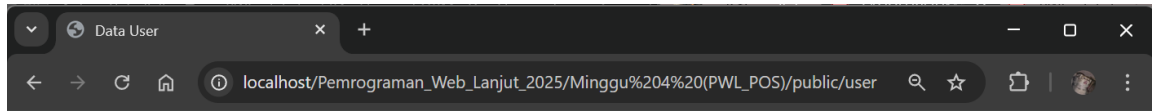
```
<body>
<h1>Data User</h1>
<table border="1" cellpadding="2" cellspacing="0">
    <tr>
        <td>ID</td>
        <td>Username</td>
        <td>Nama</td>
        <td>ID Level Pengguna</td>
    </tr>
    <tr>
        <td>{{ $data->user_id }}</td>
        <td>{{ $data->username }}</td>
        <td>{{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

⇒ Source code

```
<body>
<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>
</body>
```

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

⇒ Halaman browser



Data User

ID	Username	Nama	ID Level Pengguna
1	admin	Administrator	1

Dari hasil yang ditampilkan dapat disimpulkan bahwa, penggunaan `find(1)` akan menampilkan data berdasarkan `user_id` -nya, sehingga hanya menampilkan data dengan `user_id` 1.

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

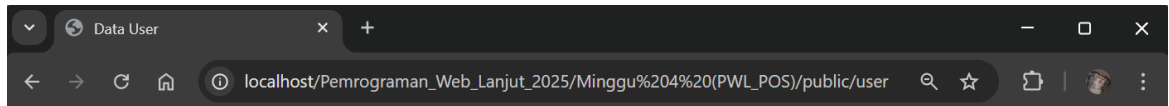
```
class UserController extends Controller
{
    public function index()
    {
        $user = UserModel::where('level_id', 1)->first();
        return view('user', ['data' => $user]);
    }
}
```

⇒ Source code

```
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

⇒ Halaman Browser



Data User

ID	Username	Nama	ID Level Pengguna
1	admin	Administrator	1

Hasil yang ditampilkan, adalah hasil dari program yang mencari berdasarkan `level_id` yang bernilai 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]);
    }
}
```

⇒ Source code

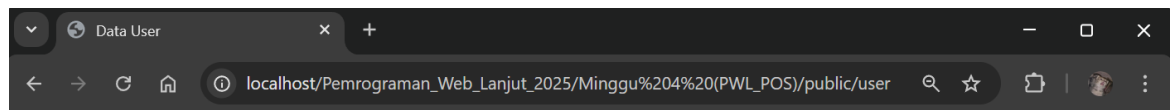
```
class UserController extends Controller
{
    public function index() {

        $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

⇒ Halaman Browser



Data User

ID	Username	Nama	ID Level Pengguna
1	admin	Administrator	1

Hasil tersebut, diakrenakan program mencari hanya pada baris pertama berdasarkan filtermya (*firstwhere*). Dan menampilkan hasil dengan *level_id* 1 di baris pertama.

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]);
    }
}
```

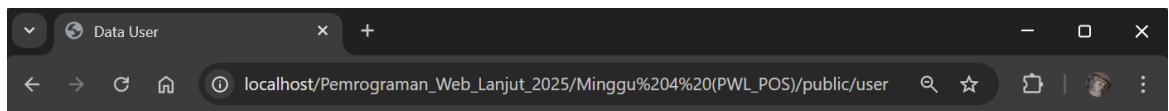
⇒ Source code

```
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

⇒ Halaman Browser



Data User

ID	Username	Nama	ID Level Pengguna
1	admin	Administrator	1

Halaman browser masih menampilkan hasil yang sama, karena perintah yang dimasukkan pada kode program memiliki maksud yang sama yaitu mencari data yang memiliki id / nilai 1.

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]);
    }
}
```

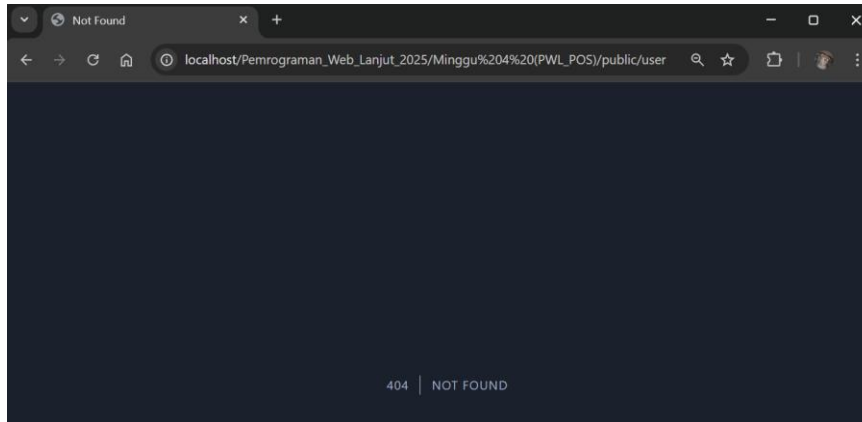
⇒ Source code

```
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

⇒ Halaman Browser



Dari tampilan tersebut dapat dilihat bahwa data tidak ditemukan, Hal ini dikarenakan tidak adanya data yang memiliki nilai / id 20. Sehingga yang ditampilkan adalah NOT FOUND atau data tidak ditemukan.

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

B. Praktikum 2.2 – Not found Exceptions

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

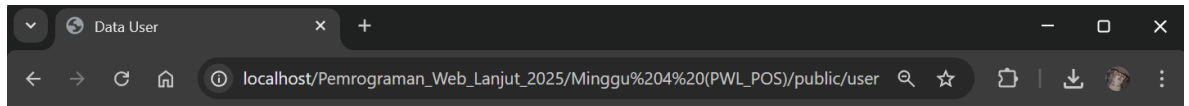
```
class UserController extends Controller
{
    public function index()
    {
        $user = UserModel::findOrFail(1);
        return view('user', ['data' => $user]);
    }
}
```

⇒ Source code

```
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

⇒ Halaman Browser



Data User

ID	Username	Nama	ID Level Pengguna
1	admin	Administrator	1

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

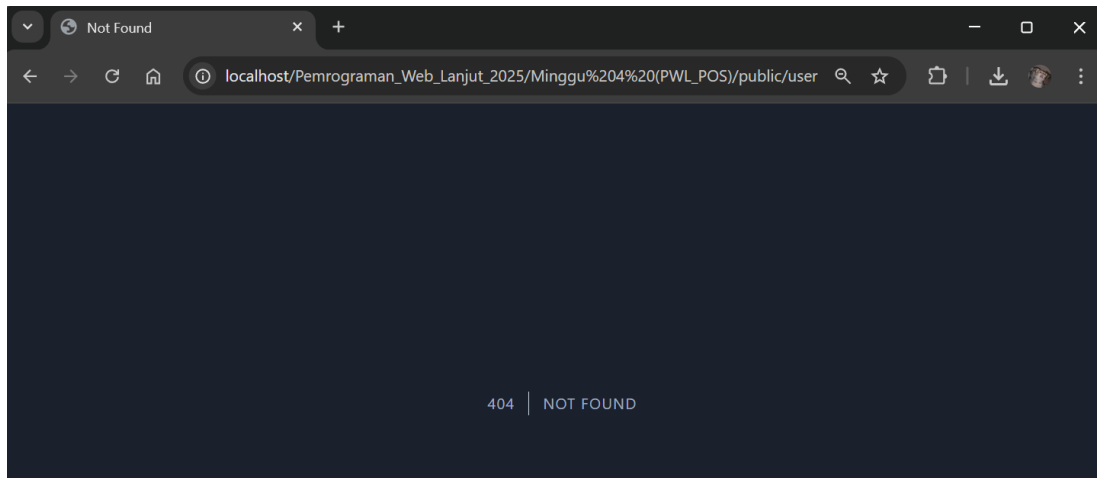
```
class UserController extends Controller
{
    public function index()
    {
        $user = UserModel::where('username', 'manager9')->firstOrFail();
        return view('user', ['data' => $user]);
    }
}
```

⇒ Source code

```
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

⇒ Halaman Browser



Dari hasil pengamatan diatas, program tersebut mencari username yang bernama 'manager9'. Lalu, menggunakan `firstOrFail` ketika sudah menemukan username tersebut akan diambil pada baris pertama dan jika tidak ditemukan akan langsung mengeluarkan status 404 not found.

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

C. Praktikum 2.3 – Retrieving Aggregates

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

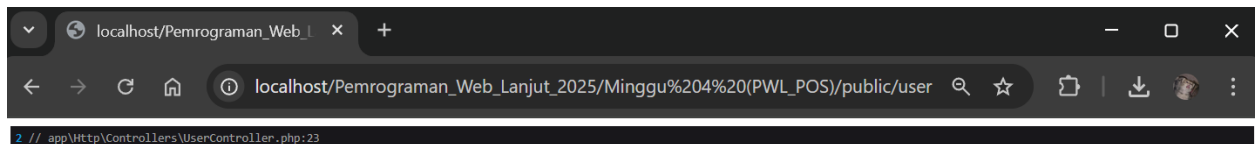
```
class UserController extends Controller
{
    public function index()
    {
        $user = UserModel::where('level_id', 2)->count();
        dd($user);
        return view('user', ['data' => $user]);
    }
}
```

⇒ Source code

```
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

⇒ Halaman browser



Dari hasil pengamatan program tersebut tidak dapat ditampilkan dikarenakan karena `dd()` yang menghentikan baris eksekusi program sehingga program dibawahnya tidak bisa berjalan

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

⇒ Source code

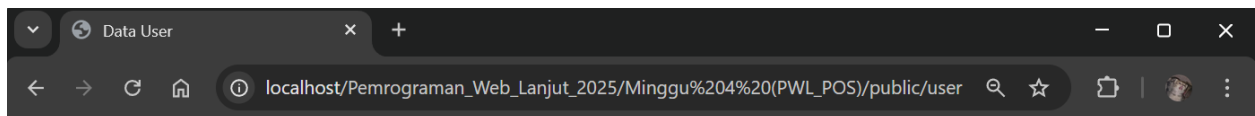
```
public function index() {
    $user = UserModel::where('level_id', 2)->count();
    return view('user', ['data' => $user]);
}
```

```

resources > views > user.blade.php > ...
1  <!DOCTYPE html>
2  <html>
3  <head>
4    <title>Data User</title>
5  </head>
6  <body>
7    <h1>Data User</h1>
8    <table border="1" cellpadding="2" cellspacing="0">
9      {{-- <tr>
10         <th>ID</th>
11         <th>Username</th>
12         <th>Nama</th>
13         <th>ID Level Pengguna</th>
14       </tr>
15       <tr>
16         <td>{{ $data->user_id }}</td>
17         <td>{{ $data->username }}</td>
18         <td>{{ $data->nama }}</td>
19         <td>{{ $data->Level_id }}</td>
20       </tr> --}}
21
22       <tr>
23         <th>Jumlah Pengguna</th>
24       </tr>
25       <tr>
26         <td>{{ $data }}</td>
27       </tr>
28     </table>
29  </body>
30 </html>
31
32

```

Halaman Browser



Data User

Jumlah Pengguna
2

D. Praktikum 2.4 – Retrieving or Creating Models

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

⇒ Source code

```
public function index(){
    $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

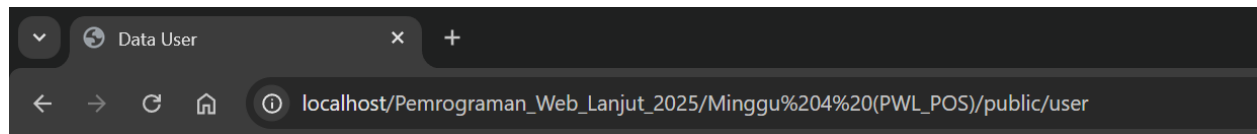
```
<body>
<h1>Data User</h1>
<table border="1" cellpadding="2" cellspacing="0">
    <tr>
        <td>ID</td>
        <td>Username</td>
        <td>Nama</td>
        <td>ID Level Pengguna</td>
    </tr>
    <tr>
        <td>{{ $data->user_id }}</td>
        <td>{{ $data->username }}</td>
        <td>{{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

⇒ Source code

```
<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

⇒ Halaman browser



Data User

ID	Username	Nama	ID Level Pengguna
2	manager	Manager	3

Dari hasil pengamatan berikut didapatkan bahwa program tersebut akan mencari pada baris pertama berdasarkan username dan nama yang telah didefinisikan. Jika program tersebut tidak menemukan nilai yang sesuai, maka akan membuat data baru atau memasukkan data baru dengan username dan nama yang telah didefinisikan.

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
            ],
            []
        );

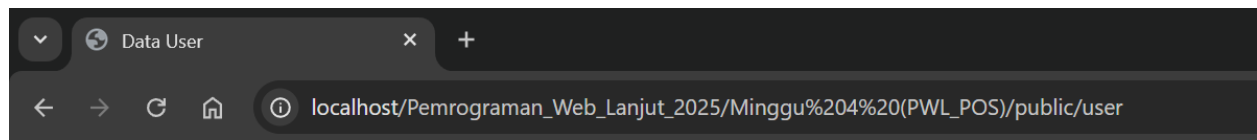
        return view('user', ['data' => $user]);
    }
}
```

⇒ Source code

```
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

⇒ Halaman browser



Data User

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

Dari hasil pengamatan diatas didapatkan bahwa tidak ada data yang sesuai dengan kriteria yang ditulis didalam program sehingga data tersebut di insertkan ke dalam database dan dikeluarkan atau ditampilkan pada halaman website.

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::firstOrCreate(
            [
                'username' => 'manager',
                'nama' => 'Manager',
            ],
        );

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

⇒ Source code

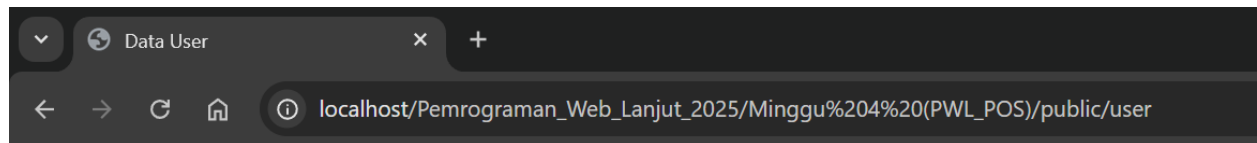
```

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

⇒ Halaman Browser



Data User

ID	Username	Nama	ID Level Pengguna
2	manager	Manager	3

Dari hasil pengamatan pada program diatas program akan mencari berdasarkan kriteria yang dimasukkan pada kasus diatas yaitu username dan nama. Lalu, akan mengembalikan hasilnya pada halaman website pada baris pertama. Jika tidak ditemukan maka data tersebut akan dibuat namun tidak akan menyimpannya di dalam database.

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]);
    }
}

```

⇒ Source code

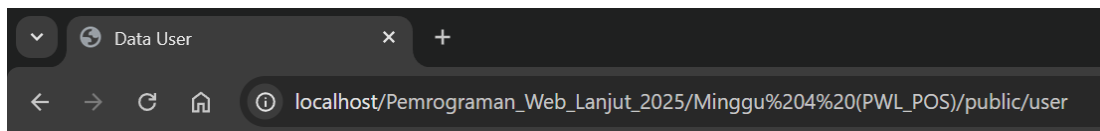

```

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

⇒ Halaman Browser



Data User

ID	Username	Nama	ID Level Pengguna
	manager33	Manager Tiga Tiga	2

Database

	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\$SLK O1bSRC33uAMgoF8ms.YmJjyoM3PwlvuCcKhHGzH...	NULL	NULL
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	2	3	manager	Manager	\$2y\$12\$UhpJhesp.YQsRspzC6kT.m4gdd9ZIAH3A1ToLjREgz...	NULL	NULL
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	3	3	staff	Staff/Kasir	\$2y\$12\$waPHJSDs3jhvmqgJUSA1v.gbMb1P5CC1EYKZZNbRMxh...	NULL	NULL
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	10	4	customer-1	Pelanggan Pertama	\$2y\$12\$TviNXNmhGSYjOTSDPI0Me00PZvT3gStXsyxiYalGIF...	NULL	2025-03-06 05:57:42
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	11	2	manager_dua	Manager 2	\$2y\$12\$ILfBECXt0oMUIZgYxGZOboCeKIUP7SZg7Y4s.slsekF...	2025-03-06 04:48:18	2025-03-06 04:48:18
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	12	2	manager_tiga	Manager 3	\$2y\$12\$OijWMQTHIFpr3xSggLMHX.2WT2qKyFCZo9PA8ACzf...	2025-03-06 06:53:32	2025-03-06 06:53:32
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	13	2	manager22	Manager Dua Dua	\$2y\$12\$uX7K03OuGLtszX6CPLAxZelA8TUwBvflQHwsZzp4Cs...	2025-03-08 09:16:15	2025-03-08 09:16:15

Seperti penjelasan sebelumnya jika data yang dimasukkan ke dalam kriteria tidak ada maka data tersebut akan dibuat dan hanya ditampilkan pada halaman website dan C. tidak akan menyimpannya ke dalam database.

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

```

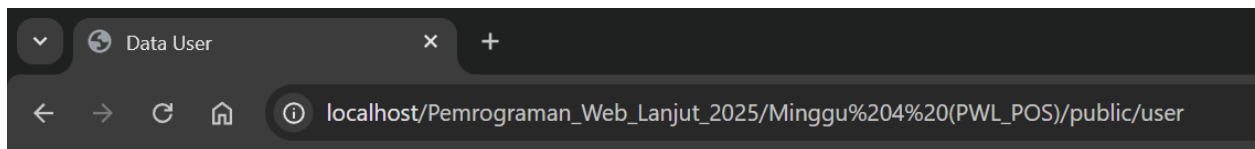
⇒ Source code

```
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]);
}
```

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

⇒ Halaman Browser



Data User

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

Database

		user_id	level_id	username	nama	password	created_at	updated_at
<input type="checkbox"/>	Edit Copy Delete	1	1	admin	Administrator	\$2y\$12\$SLK01bSRC33uAMgoF18ms.YmJjyoM3PwlvuCckhHGzH...	NULL	NULL
<input type="checkbox"/>	Edit Copy Delete	2	3	manager	Manager	\$2y\$12\$UhpJhesp.YQsRspzC6kT.m4gdd9ZIAH3A1ToLjREgz...	NULL	NULL
<input type="checkbox"/>	Edit Copy Delete	3	3	staff	Staff/Kasir	\$2y\$12\$waPHJSdS3jhvmqgJUSA1v.gbMb1P5CC1EYKZZNbRMxh...	NULL	NULL
<input type="checkbox"/>	Edit Copy Delete	10	4	customer-1	Pelanggan Pertama	\$2y\$12\$TfVtNXnmhGSYJOTSDPI0Me00PZvT3gStXsyxiYalGIF...	NULL	2025-03-06 05:57:42
<input type="checkbox"/>	Edit Copy Delete	11	2	manager_dua	Manager 2	\$2y\$12\$ilfbECxT0oMUIZgYxGZObOCeKIUP7SZg7Y4s.slxeF...	2025-03-06 04:48:18	2025-03-06 04:48:18
<input type="checkbox"/>	Edit Copy Delete	12	2	manager_tiga	Manager 3	\$2y\$12\$OijWMQTHIFpr3xSggMHX/2WT2q/KyFCZo9PA8ACzf...	2025-03-06 06:53:32	2025-03-06 06:53:32
<input type="checkbox"/>	Edit Copy Delete	13	2	manager22	Manager Dua Dua	\$2y\$12\$uX7K03OuGLtszX6CPLAxZelA8TUwBvllQHwiSZzp4Cs...	2025-03-08 09:16:15	2025-03-08 09:16:15
<input type="checkbox"/>	Edit Copy Delete	14	2	manager33	Manager Tiga Tiga	\$2y\$12\$2TW0RIW4sw0qwlYLdqNfrepYOmVUY7rKJ6W0YNV3xtO...	2025-03-08 09:27:45	2025-03-08 09:27:45

Dari hasil pengamatan diatas didapatkan bahwa data tersebut yang tadinya tidak disimpan ke dalam database dan hanya ditampilkan di halaman website kini menjadi tersimpan. Hal tersebut bisa terjadi dikarenakan pada program menggunakan function *save()*; yang dimana hal tersebut yang membuat data nya menjadi tersimpan di dalam database.

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

E. Praktikum 2.5 – Attribute Changes

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::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());
    }
}
```

⇒ Source code

```
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

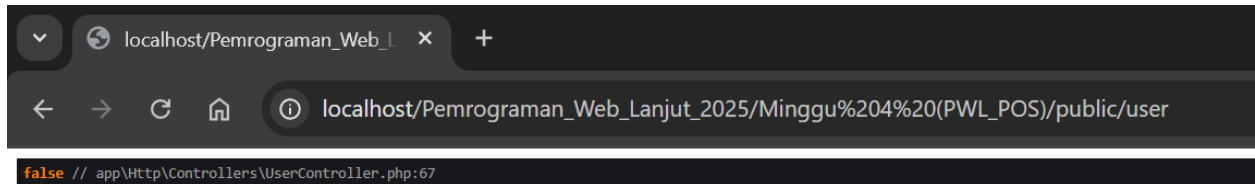
    $user->isClean(); // false
    $user->isClean('username'); // false
    $user->isClean('nama'); // true
    $user->isClean(['nama', 'username']); // false

    $user->save();

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

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

⇒ Halaman Browser



Dari hasil pengamatan diatas kenapa hasil nya adalah false dikarenakan setelah perubahan disimpan, tidak ada lagi perubahan baru yang belum disimpan pada objek \$user.

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
    }
}
```

⇒ Source code

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

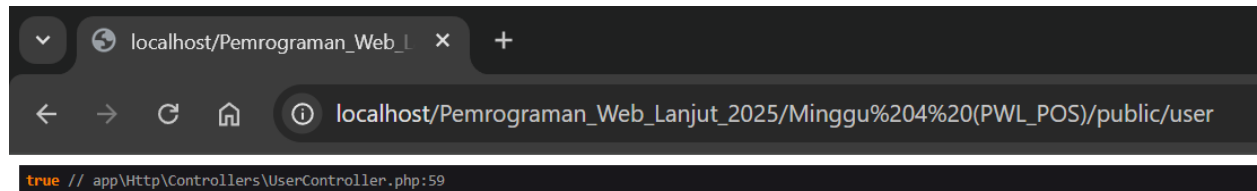
    $user->username = 'manager12';

    $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

⇒ Halaman Browser



Dari hasil pengamatan pada program diatas kenapa hasilnya adalah true dikarenakan terdapat perubahan dari atribut nama atau username sejak di insert ke database. Walaupun perubahan pada kolom username telah disimpan namun wasChanged akan mendeteksi kolom tersebut sejak pertama kali di insertkan pada database.

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

F. Praktikum 2.6 – Create, Read, Update, Delete (CRUD)

1. Buka file *view* pada *user.blade.php* dan buat scripnya 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>
```

⇒ Source code

```
<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

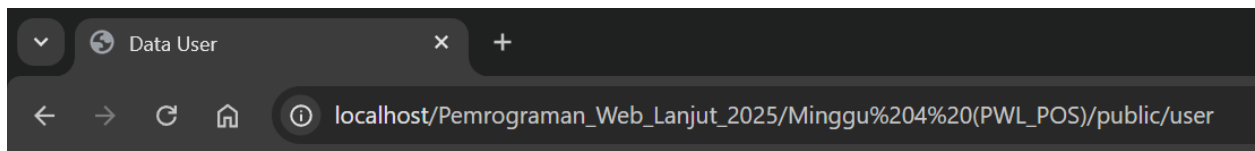
```
class UserController extends Controller
{
    public function index()
    {
        $user = UserModel::all();
        return view('user', ['data' => $user]);
    }
}
```

⇒ Source code

```
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

⇒ Halaman Browser



Data User

[+ Tambah User](#)

ID	Username	Nama	ID Level Pengguna	Aksi
1	admin	Administrator	1	Ubah Hapus
2	manager	Manager	3	Ubah Hapus
3	staff	Staff/Kasir	3	Ubah Hapus
10	customer-1	Pelanggan Pertama	4	Ubah Hapus
11	manager_dua	Manager 2	2	Ubah Hapus
12	manager_tiga	Manager 3	2	Ubah Hapus
13	manager22	Manager Dua Dua	2	Ubah Hapus
14	manager33	Manager Tiga Tiga	2	Ubah Hapus
15	manager56	Manager55	2	Ubah Hapus
16	manager12	Manager11	2	Ubah Hapus

Dari hasil pengamatan diatas diatas kita menampilkan semua data pada database ke dalam halaman website yang nantinya akan kita lakukan proses aksi seperti ubah dan hapus. Namun, pada saat ini proses tersebut belum bisa diakses belum dibuat untuk link nya.

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>
```

⇒ Source code

```
PS D:\laragon\www\Pemrograman_Web_Lanjut_2025\Minggu 4 (PWL_P05)> php artisan make:view user_tambah
```

```
INFO View [D:\laragon\www\Pemrograman_Web_Lanjut_2025\Minggu 4 (PWL_P05)\resources\views\user_tambah.blade.php] created successfully.
```

```
resources > views > user_tambah.blade.php > ...
```

```
1  <html>
2    <body>
3      <h1>Form Tambah Data User</h1>
4      <form method="post" action="/user/tambah_simpan">
5
6        {{ csrf_field() }}
7
8        <label>Username</label>
9        <input type="text" name="username" placeholder="Masukan Username">
10       <br>
11       <label>Nama</label>
12       <input type="text" name="nama" placeholder="Masukan Nama">
13       <br>
14       <label>Password</label>
15       <input type="password" name="password" placeholder="Masukan Password">
16       <br>
17       <label>Level ID</label>
18       <input type="number" name="level_id" placeholder="Masukan ID Level">
19       <br><br>
20       <input type="submit" class="btn btn-success" value="Simpan">
21
22     </form>
23   </body>
24 </html>
```

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

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

⇒ Source code

```
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');
    }
}
```

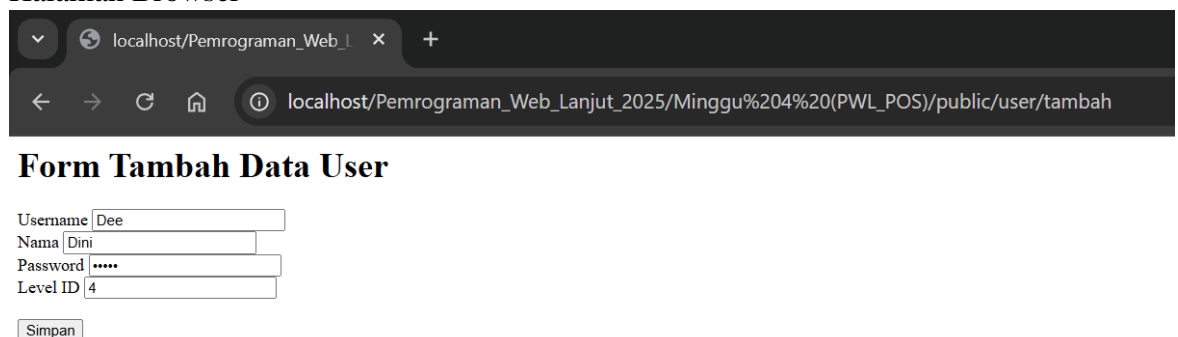
⇒ Source code

```
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

⇒ Halaman Browser



localhost/Pemrograman_Web_Lanjut_2025/Minggu%204%20(PWL_POS)/public/user/tambah

Form Tambah Data User

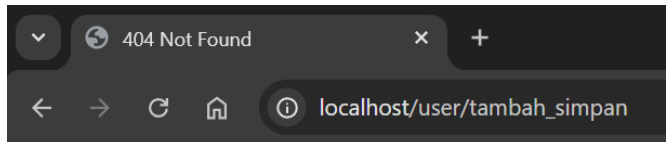
Username

Nama

Password

Level ID

Tetapi tidak bisa tersimpan dikarenakan belum ada konfigurasi untuk link `tambah_simpan` dan juga belum ada nya query untuk menyimpan ke dalam database jadi hanya sebatas form biasa.



Not Found

The requested URL was not found on this server.

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']);
```

⇒ Source code

```
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
    ]);

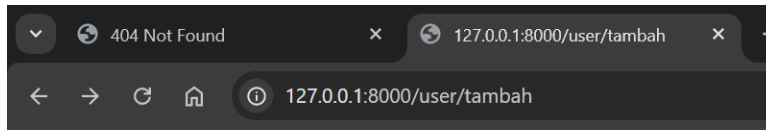
    return redirect('/user');
}
```

⇒ Source code

```
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

⇒ Halaman Browser



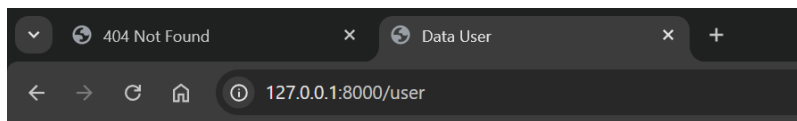
Form Tambah Data User

Username

Nama

Password

Level ID



Data User

[+ Tambah User](#)

ID	Username	Nama	ID Level Pengguna	Aksi
1	admin	Administrator	1	Ubah Hapus
2	manager	Manager	3	Ubah Hapus
3	staff	Staff/Kasir	3	Ubah Hapus
10	customer-1	Pelanggan Pertama	4	Ubah Hapus
11	manager_dua	Manager 2	2	Ubah Hapus
12	manager_tiga	Manager 3	2	Ubah Hapus
13	manager22	Manager Dua Dua	2	Ubah Hapus
14	manager33	Manager Tiga Tiga	2	Ubah Hapus
15	manager56	Manager55	2	Ubah Hapus
16	manager12	Manager11	2	Ubah Hapus
17	Dee	Dini	4	Ubah Hapus

<input type="checkbox"/>	Edit	Copy	Delete	14	2	manager33	Manager Tiga Tiga	\$2y\$12\$2TW0RIW4sw0q
<input type="checkbox"/>	Edit	Copy	Delete	15	2	manager56	Manager55	\$2y\$12\$H6iKFIDlrMEDkc
<input type="checkbox"/>	Edit	Copy	Delete	16	2	manager12	Manager11	\$2y\$12\$88xZhZf9UNTwt
<input type="checkbox"/>	Edit	Copy	Delete	17	4	Dee	Dini	\$2y\$12\$zQHI8t1Kui/7C6r

Dari hasil tersebut dapat dilihat bahwa ketika menambahkan user, sudah bisa tersimpan di data usernya. Jadi dimulai dari form yang diisi data baru kemudian dikirimkan kedalam sebuah controller yang akan memasukkannya ke dalam database dan mendirect kembali ketika proses tersebut berhasil dilakukan dan menampilkan datanya seperti pada gambar diatas.

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

```
<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>
```

⇒ Source code

```
PS D:\laragon\www\Pemrograman_Web_Lanjut_2025\Minggu 4 (PWL_P05)> php artisan make:view user_ubah

INFO View [D:\laragon\www\Pemrograman_Web_Lanjut_2025\Minggu 4 (PWL_P05)\resources\views\user_ubah.blade.php] created successfully.

resources > views > user_ubah.blade.php > body
1  <body>
2  <h1>Form Ubah Data user</h1>
3  <a href="/user">Kembali</a>
4  <br><br>
5
6  <form method="post" action="/user/ubah_simpan/{{ $data->user_id }}">
7
8      {{ csrf_field() }}
9      {{ method_field('PUT') }}
10
11     <label>Username</label>
12     <input type="text" name="username" placeholder="Masukan Username" value="{{ $data->username }}">
13     <br>
14
15     <label>Nama</label>
16     <input type="text" name="nama" placeholder="Masukan Nama" value="{{ $data->username }}">
17     <br>
18
19     <label>Password</label>
20     <input type="password" name="password" placeholder="Masukan Password" value="{{ $data->password }}">
21     <br>
22
23     <label>Level ID</label>
24     <input type="number" name="level_id" placeholder="Masukan ID Level" value="{{ $data->level_id }}">
25     <br><br>
26
27     <input type="submit" class="btn btn-success" value="Ubah">
28
29     </form>
30 </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']);
```

⇒ Source code

```
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

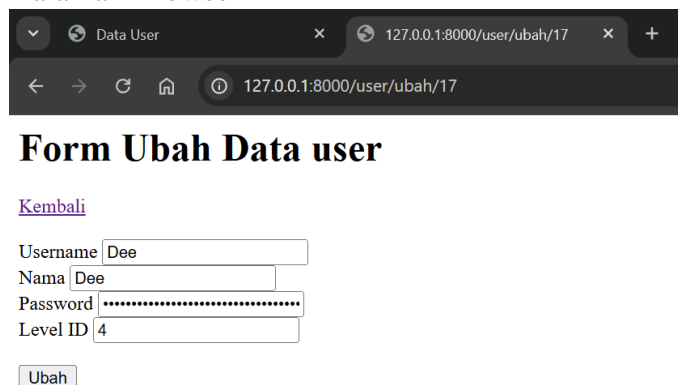
```
public function ubah($id)
{
    $user = UserModel::find($id);
    return view('user_ubah', ['data' => $user]);
}
```

⇒ Source code

```
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

⇒ Halaman Browser



Form Ubah Data user

[Kembali](#)

Username

Nama

Password

Level ID

ketika klik ubah belum bisa untuk melakukan perubahan secara langsung dikarenakan perubahan belum disimpan dan belum ada program yang melakukan penyimpanan hasil perubahan. Program diatas hanya melakukan input data saja.

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']);
```

⇒ Source code

```
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');
}
```

⇒ Source code

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

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

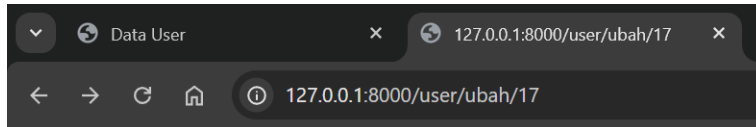
    $user->username = $request->username;
    $user->nama = $request->nama;
    $user->password = Hash::make($request->password);
    $user->level_id = $request->level_id;

    $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

⇒ Halaman Browser setelah di klik `ubah`



Form Ubah Data user

[Kembali](#)

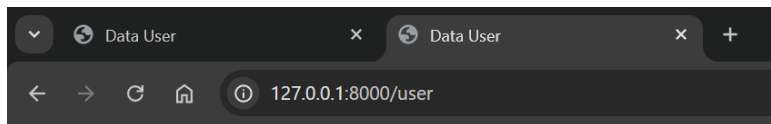
Username

Nama

Password

Level ID

Halaman browser setelah diubah



Data User

[+ Tambah User](#)

ID	Username	Nama	ID Level Pengguna	Aksi
1	admin	Administrator	1	Ubah Hapus
2	manager	Manager	3	Ubah Hapus
3	staff	Staff/Kasir	3	Ubah Hapus
10	customer-1	Pelanggan Pertama	4	Ubah Hapus
11	manager_dua	Manager 2	2	Ubah Hapus
12	manager_tiga	Manager 3	2	Ubah Hapus
13	manager22	Manager Dua Dua	2	Ubah Hapus
14	manager33	Manager Tiga Tiga	2	Ubah Hapus
15	manager56	Manager55	2	Ubah Hapus
16	manager12	Manager11	2	Ubah Hapus
17	Dee	Dee	1	Ubah Hapus

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']);
```

⇒ Source code

```
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');
}
```

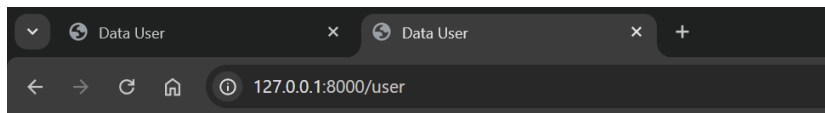
⇒ Source code

```
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

⇒ Halaman Browser



Data User

[+ Tambah User](#)

ID	Username	Nama	ID Level Pengguna	Aksi
1	admin	Administrator	1	Ubah Hapus
2	manager	Manager	3	Ubah Hapus
3	staff	Staff/Kasir	3	Ubah Hapus
10	customer-1	Pelanggan Pertama	4	Ubah Hapus
11	manager_dua	Manager 2	2	Ubah Hapus
12	manager_tiga	Manager 3	2	Ubah Hapus
13	manager22	Manager Dua Dua	2	Ubah Hapus
14	manager33	Manager Tiga Tiga	2	Ubah Hapus
15	manager56	Manager55	2	Ubah Hapus
16	manager12	Manager11	2	Ubah Hapus

proses delete data yang melewati route yang nantinya akan memanggil controller untuk langsung menghapus berdasarkan `user_id` sehingga tidak menghapus data yang salah. Setelah proses penghapusan selanjutnya adalah di direct kembali ke halaman awal Dimana menampilkan semua user nya.

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

c. Praktikum 2.7 – Relationships

1. Buka file model pada `UserModel.php` dan tambahkan scriptnya 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');
    }
}
```

⇒ Source code

```
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

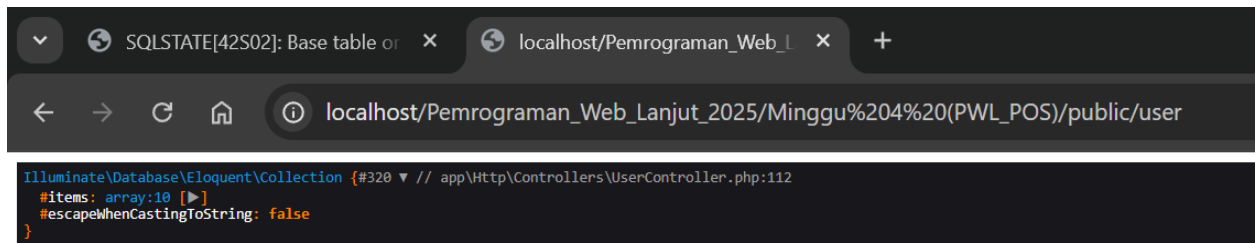
```
public function index()
{
    $user = UserModel::with('level')->get();
    dd($user);
}
```

⇒ Source code

```
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

⇒ Halaman Browser



Dari hasil diatas kita dapat mengetahui terkait jumlah data dari array items tersebut dan juga penjelasan terkait escapeCastingToString yang bernilai false yang nantinya karakter tersebut tidak akan di-encode.

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]);
}
```

⇒ Source code

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

5. 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>
```

⇒ Source code

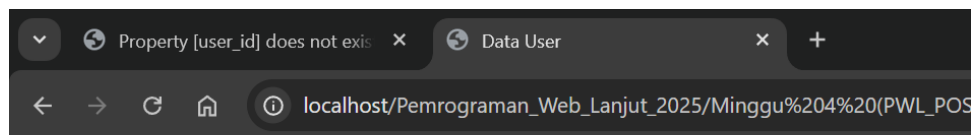
```

<head>
  <title>Data User</title>
</head>
<body>
  <h1>Data User</h1>
  <a href="{{ url('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>
      </tr>
    @endforeach
  </table>

```

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

⇒ Halaman Browser



Data User

[+ Tambah User](#)

ID	Username	Nama	ID Level Pengguna	Kode Level	Nama Level	Aksi
1	admin	Administrator	1	ADM	Administrator	Ubah Hapus
2	manager	Manager	3	STF	Staff/Kasir	Ubah Hapus
3	staff	Staff/Kasir	3	STF	Staff/Kasir	Ubah Hapus
10	customer-1	Pelanggan Pertama	4	CUS	Customer	Ubah Hapus
11	manager_dua	Manager 2	2	MNG	Manager	Ubah Hapus
12	manager_tiga	Manager 3	2	MNG	Manager	Ubah Hapus
13	manager22	Manager Dua Dua	2	MNG	Manager	Ubah Hapus
14	manager33	Manager Tiga Tiga	2	MNG	Manager	Ubah Hapus
15	manager56	Manager55	2	MNG	Manager	Ubah Hapus
16	manager12	Manager11	2	MNG	Manager	Ubah Hapus

Dari hasil pengamatan diatas merupakan proses join pada database atau menggabungkan dua tabel dan ditampilkan ke dalam halaman website. Disini yang perlu diperhatikan adalah relasi antar tabel sehingga pengimplementasian pada program bisa berjalan dan hasil nya bisa ditampilkan seperti pada gambar diatas.

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