KEMENTERIAN PENDIDIKAN, KEBUDAYAAN, RISET, DAN TEKNOLOGI
POLITEKNIK NEGERI MALANG
JURUSAN TEKNOLOGI INFORMASI
Jl. Soekarno Hatta No. 9, Jatimulyo, Lowokwaru, Malang 65141
Telp. (0341) 404424 – 404425, Fax (0341) 404420
https://jti.polinema.ac.id

Mata Kuliah	:	Pemrograman Web Lanjut (PWL)
Program Studi	:	D4 – Sistem Informasi Bisnis
Semester	:	5

Kelas	:	SIB 3D
NIM	:	2241760030
Nama	:	Fardiyani Afro'ul Yasinta
Jobsheet Ke-	:	4
Link GitHub	:	https://github.com/Fafayasinta/PWL_POS.git

Laporan Jobsheet

Praktikum Ke-1 | $fillable

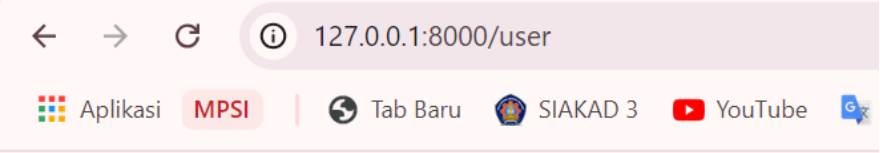
Langkah	Jawaban/Deskripsi

1.	Buka file model dengan nama UserModel.php dan tambahkan $fillable seperti gambar di bawah ini
	3 references | 0 implementations
	`class UserModel extends Model`
	`{`
	`use HasFactory;`
	0 references
	`protected $table = 'm_user'; // Mendefinisikan nama tabel yang digunak`
	0 references
	`protected $primaryKey = 'user_id'; // Mendefinisikan primary key dari`
	0 references
	`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

```
class UserController extends Controller
{
    1 reference | 0 overrides
    public function index(): Factory|View{
        // tambah data menggunakan Eloquent
        $data = [
            'username' => 'manager_dua',
            'nama' => 'Manager2',
            'password' => Hash::make(value: '12345'),
            'level_id' => 2
        ];
        UserModel::insert(values: $data);
```

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
13	customer 1	Pelanggan pertama	4
14	manager_dua	Manager2	2

Dari hasil pengamatan terdapat penambahan data dikarenakan pada proses kedua yaitu membuat data atau menginsertkan data baru ke dalam database. Lalu, data tersebut dikeluarkan atau dikembalikan agar bisa tertampilkan. Atribut tersebut diisi oleh array datayang nantinya dicreate dan dimasukkan ke dalam database.

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

	3 references | 0 implementations
	`class UserModel extends Model`
	`{`
	`use HasFactory;`
	0 references
	`protected $table = 'm_user'; // Mendefinisikan nama tabel yang digunak`
	0 references
	`protected $primaryKey = 'user_id'; // Mendefinisikan primary key dari`
	0 references
	`protected $fillable = ['level_id', 'username', 'nama'];`
	`}`

5.	Ubah kembali file controller UserController.php seperti pada gambar di bawah hanya bagian array pada $data
	2 references | 0 implementations
	`class UserController extends Controller`
	`{`
	1 reference | 0 overrides
	`public function index(): Factory|View{`
	`// tambah data menggunakan Eloquent`
	`$data = [`
	`'level_id' => 2,`
	`'username' => 'manager_tiga',`
	`'nama' => 'Manager 3',`
	`'password' => Hash::make(value: '12345'),`
	`];`
	`UserModel::insert(values: $data);`

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

<table>
<tr><td></td><td>

Data user

ID	Username	Nama	ID Level pengguna
1	admin	Administrator	1
2	manager	Manager	2
3	staff	Staff/Kasir	3
13	customer 1	Pelanggan pertama	4
14	manager_dua	Manager2	2
17	manager_tiga	Manager 3	2
</td></tr>
<tr><td>7.</td><td>Laporkan hasil Praktikum-1 ini dan commit perubahan pada git.</td></tr>
</table>

Praktikum Ke-2.1 | Retrieving Single Models

Langkah	Jawaban/Deskripsi

1.	Buka file controller dengan nama UserController.php dan ubah script seperti gambar di bawah ini
	`$user = UserModel::find(id: 1);`
	`return view(view: 'user', data: ['data' => $user]);`
	`}`
	`}`

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

resources > views > user.blade.php > html > body > table

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

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

Dari hasil pengamatan yang terjadi penggunaan find() nantinya akan mengeluarkan data berdasarkan user_id nya. Jika yang diinputkan 1 pada kasus diatasmaka yang akan keluar adalah data dengan user_id 1.

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

```
$user = UserModel::where(column: 'level_id', operator: 1)->first();
return view(view: 'user', data: ['data' => $user]);
```

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

Dari hasil pengamatan didapat bahwa pada program sebelumnya merupakan query berdasarkan pemfilteran kolom level_id lalu dimasukkan valuenya. Pada penggunaan first digunakan untuk mencari pada baris pertama setelah dimasukkan berbagai kriteria nya.

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

```
        $user = UserModel::firstWhere(column: 'level_id', operator: 1);
        return view(view: 'user', data: ['data' => $user]);
    }
}
```

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

<table>
<tr><td></td><td>

Data User

ID	Username	Nama	ID Level Pengguna
1	admin	Administrator	1

Dari hasil pengamatan berikut, didapatkan bahwa penggunaan program tersebut untuk mencari hanya pada baris pertama berdasarkan pemfilterannya. Pada kasus diatas adalah mencari level_id yang bernilai 1. Lalu, dikeluarkan sebuah hasil yang bernilai 1 dan berada di baris pertama.
</td></tr>
<tr><td>8.</td><td>
Ubah file controller dengan nama UserController.php dan ubah script seperti gambar di bawah ini

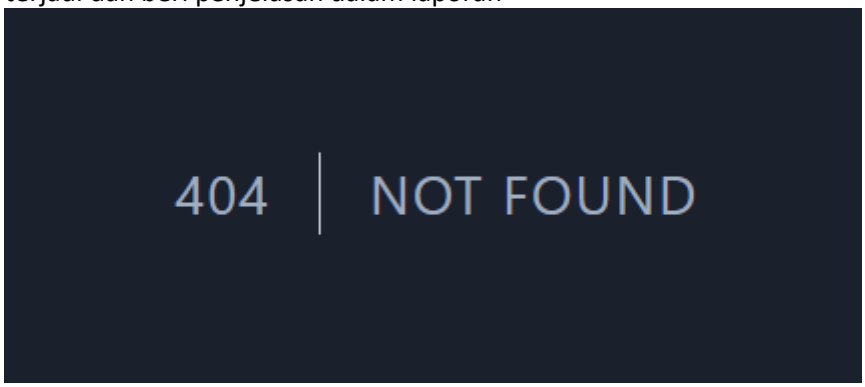
```
    $user = UserModel::findOr(id: 1, columns: ['username', 'nama'], callback: function (): never {
        abort(code: 404);
    });
    return view(view: 'user', data: ['data' => $user]);
}
```
</td></tr>
<tr><td>9.</td><td>
Simpan kode program Langkah 8. Kemudian pada browser dan amati apa yang terjadi dan beri penjelasan dalam laporan

Data User

ID	Username	Nama	ID Level Pengguna
	admin	Administrator	

Dari hasil pengamatan, program tersebut nantinya akan mencari berdasarkan id yang dimana bernilai 1. Jika tidak ditemukan nantinya akan mencari kolom lain yaitu pada username dan nama yang bernilai 1. Jika tidak ada maka nantinya akan dikeluarkan error 404not found.
</td></tr>
<tr><td>10.</td><td>
Ubah file controller dengan nama UserController.php dan ubah script seperti gambar di bawah ini
</td></tr>
</table>

	``php
	$user = UserModel::findor(id: 20, columns: ['username', 'nama'], callback: function (): never {
	abort(code: 404);
	});
	return view(view: 'user', data: ['data' => $user]);
	}
	}
	``

11.	Simpan kode program Langkah 10. Kemudian jalankan pada browser dan amati apa yang terjadi dan beri penjelasan dalam laporan  Dari penjelasan sebelumnya sudah dijelaskan terkait hal ini. Hal ini bisa terjadi dikarenakan tidak adanya value yang bernilai 20 berdasarkan id, username dan nama. Sehingga dikeluarkan respon berupa 404 not found atau data tidak ditemukan.
12.	Laporkan hasil Praktikum-2.1 ini dan commit perubahan pada git.

Praktikum Ke-2.2 | Not Found Exceptions

Langkah	Jawaban/Deskripsi

1.	Ubah file controller dengan nama UserController.php dan ubah script seperti gambar di bawah ini
	``php
	$user = UserModel::findOrFail(id: 1);
	return view(view: 'user', data: ['data' => $user]);
	``

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

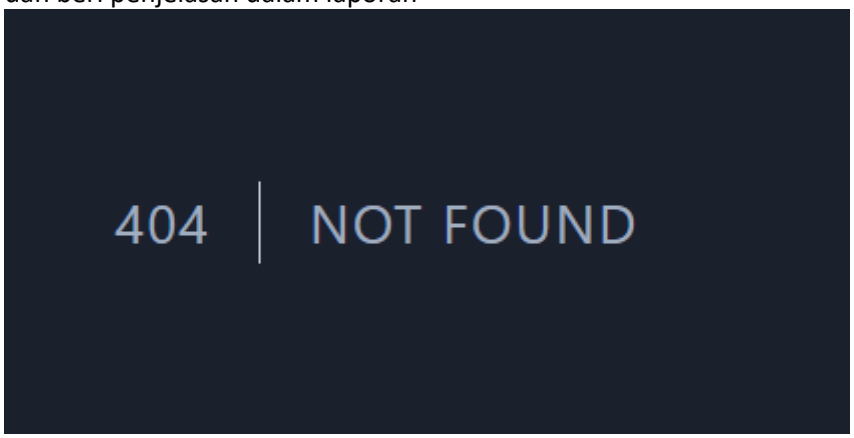
<table>
<tr><td></td><td>

Data User

ID	Username	Nama	ID Level Pengguna
1	admin	Administrator	1

Dari hasil pengamatan didapatkan bahwa program tersebut mencari id dengan nilai 1. Jika id tersebut memang tidak ada maka program akan langsung mengeluarkan status 404 not found. Dikarenakan penggunaan findOrFail.
</td></tr>
<tr><td>3.</td><td>
Ubah file controller dengan nama UserController.php dan ubah script seperti gambar di bawah ini

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

}
```
</td></tr>
<tr><td>4.</td><td>
Simpan kode program Langkah 3. Kemudian jalankan pada browser dan amati apa yang terjadi dan beri penjelasan dalam laporan

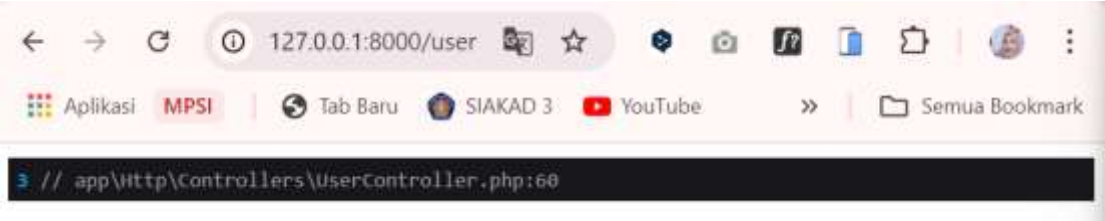


Dari hasil pengamatan diatas, program tersebut mencari username yang bernama ‘manager9’. Lalu, menggunakan firstOrFail ketika sudah menemukan username tersebut akan diambil pada baris pertama dan jika tidak ditemukan akan langsung mengeluarkan status 404 not found.
</td></tr>
<tr><td>5.</td><td>Laporkan hasil Praktikum 2.2 ini dan commit perubahan pada git.</td></tr>
</table>

Praktikum Ke-2.3 | Retreiving Aggregates

Langkah	Jawaban/Deskripsi

1.	Ubah file controller dengan nama UserController.php dan ubah script seperti gambar di bawah ini
	`$user = UserModel::where(column: 'level_id', operator: 2)->count();`
	`dd(vars: $user);`
	`return view(view: 'user', data: ['data' => $user]);`

2.	Simpan kode program Langkah 1. Kemudian jalankan pada browser dan amati apa yang terjadi dan beri penjelasan dalam laporan  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 Untuk bisa menampilkan hasilnya pada halaman website yang perlu dilakukan adalah menghapus barisan yang mengandung program dd(). Lalu, pada view dikonfigurasi seperti pada gambar dibawah **Jawab:** Untuk bisa menampilkan hasilnya pada halaman website yang perlu dilakukan adalah menghapus barisan yang mengandung program dd(). Lalu, pada view dikonfigurasi seperti pada gambar dibawah

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

Outputnya

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

Praktikum Ke-2.4 | Retreiving or Creating Models

Langkah	Jawaban/Deskripsi

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

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

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

<table>
<tr><td></td><td>

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 menginsertkan data baru dengan username dan nama yang telah didefinisikan.
</td></tr>
<tr><td>4.</td><td>
Ubah file controller dengan nama UserController.php dan ubah script seperti gambar di bawah ini

</td></tr>
<tr><td>5.</td><td>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</td></tr>
</table>

Data User

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

Tampilan Pada database

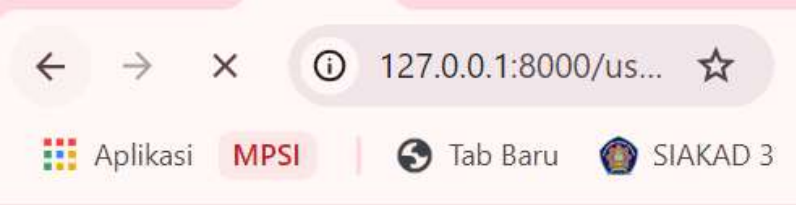
←T→				user_id	level_id	username	nama	password
Edit	Copy	Delete		1	1	admin	Administrator	$2y$12$2W0JrC0ymRtwLCzHxePbFeGu1FIE9lUr
Edit	Copy	Delete		2	2	manager	Manager	$2y$12$viwkf3t5GraM02HJQgHE/upoJHGMIXEW
Edit	Copy	Delete		3	3	staff	Staff/Kasir	$2y$12$zKNZKxqG77wJR86mBwlLNeOLpVL3aO
Edit	Copy	Delete		13	4	customer 1	Pelanggan pertama	$2y$12$AgIJ8mj8t8L4Ah9e7wboVu4RM7GsGP/Li
Edit	Copy	Delete		14	2	manager_dua	Manager2	$2y$12$8C3a5A1ZXrkEr1gJCqEAI.m3NFYBCFny
Edit	Copy	Delete		17	2	manager_tiga	Manager 3	$2y$12$eWMwMGtmMCSDID.olak7g.51bzpEV2L
Edit	Copy	Delete		18	2	manager22	Manager Dua Dua	$2y$12$t26K6c83lpFZelLUQ/ys9eJScfDHM6bsSY

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. Seperti pada gambar diatas.

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

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

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

<table>
<tr><td></td><td>


Data User

ID	Username	Nama	ID Level Pengguna
2	manager	Manager	2

Dari hasil pengamatan pada program diatas program akan mencari berdasarkan kriteria yang dimasukkan pada kasus diatas yaitu usernamedan 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.
</td></tr>
<tr><td>8.</td><td>
Ubah file controller dengan nama UserController.php dan ubah script seperti gambar di bawah ini

```
$user = UserModel::firstOrNew(
    [
        'username' => 'manager33',
        'nama' => 'Manager Tiga Tiga',
        'password' => Hash::make('12345'),
        'level_id' => 2
    ],
);
return view('user', ['data' => $user]);
```
</td></tr>
<tr><td>9.</td><td>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</td></tr>
</table>

Data User

ID	Username	Nama	ID Level Pengguna
	manager33	Manager Tiga Tiga	2

Tampilan pada database

	user_id	level_id	username	nama	password
Edit Copy Delete	1	1	admin	Administrator	$2y$12$2W0JrC0ymRtwLCzHxePbFeGu1FIE9iUr
Edit Copy Delete	2	2	manager	Manager	$2y$12$viwkf3t5GraM02HJQgHE/upoJHGMIXEW
Edit Copy Delete	3	3	staff	Staff/Kasir	$2y$12$zKNZKxqG77wJR86mBwILNeOLpVL3aO
Edit Copy Delete	13	4	customer 1	Pelanggan pertama	$2y$12$AgIJ8mj8t8L4Ah9e7wboVu4RM7GsGP/L
Edit Copy Delete	14	2	manager_dua	Manager2	$2y$12$8C3a5A1ZXrkEr1gJCqEAI.m3NFYBCFny
Edit Copy Delete	17	2	manager_tiga	Manager 3	$2y$12$eWMwMGtmMCSDID.olak7g.51bzpEV2L
Edit Copy Delete	18	2	manager22	Manager Dua Dua	$2y$12$t26K6c83lpFZeILUQ/ys9eJScfDHM6bsS

Dari hasil pengamatan diatas, sesuai dengan penjelasan sebelumnya jika data yang dimasukkan ke dalam kriteria tidak ada maka data tersebut akan dibuat dan hanya ditampilkan pada halaman website dan tidak akan menyimpannya ke dalam database.

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

```
$user = UserModel::firstOrNew(
    attributes: [
        'username' => 'manager33',
        'nama' => 'Manager Tiga Tiga',
        'password' => Hash::make(value: '12345'),
        'level_id' => 2
    ],
);
$user->save();

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

<table>
<tr><td>11.</td><td>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
19	manager33	Manager Tiga Tiga	2

Tampilan pada database

	user_id	level_id	username	nama	password
Edit Copy Delete	1	1	admin	Administrator	$2y$12$2W0JrC0ymRtwLCzHxePbFeGu1FIE9kUr
Edit Copy Delete	2	2	manager	Manager	$2y$12$viwkf3t5GraM02HJQgHEuoJHGMIXEW
Edit Copy Delete	3	3	staff	Staff/Kasir	$2y$12$zKNZKxgG77wJR86mBwILNeOLpVL3aO
Edit Copy Delete	13	4	customer 1	Pelanggan pertama	$2y$12$AgUi8mj8tSL4Ah9e7wboVu4RM7GsGP/Lt
Edit Copy Delete	14	2	manager_dua	Manager2	$2y$12$8C3a5A1ZXrkEr1gJCqEAl.m3NFYBCFny
Edit Copy Delete	17	2	manager_tiga	Manager 3	$2y$12$eWlwMGbmMCSDID.olak7g.51bzpEV2L
Edit Copy Delete	18	2	manager22	Manager Dua Dua	$2y$12$t26K5c83lpFZelLUQ/ys9eJScfDHM6bsSY
Edit Copy Delete	19	2	manager33	Manager Tiga Tiga	$2y$12$5.9JjW28JKzLDThFURFJ.St8WEtCMpJ

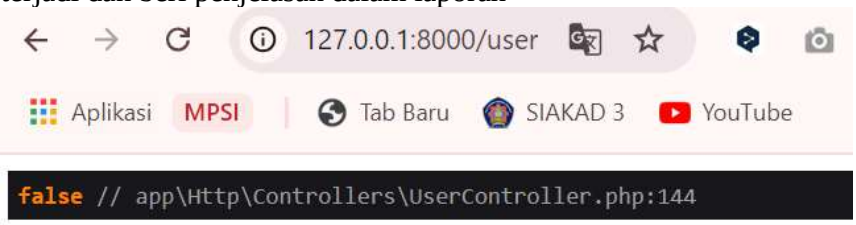
Dari hasil pengamatan diatas didapatkan bahwa data tersebut yang tadinya tidak disimpan ke dalam database dan hanya ditampilkan di halaman website kini menjadi tersimpan. Hal tersebut bisa terjadi dikarenakan pada program menggunakan function save(); yang dimana hal tersebut yang membuat data nya menjadi tersimpan di dalam database.</td></tr>
<tr><td>12.</td><td>Laporkan hasil Praktikum-2.4 ini dan commit perubahan pada git.</td></tr>
</table>

Praktikum Ke-2.5 | Attribute Changes

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

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

        $user->username = 'manager56';

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

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

        $user->save(); //simpan data ke database

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

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



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

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

Praktikum Ke-2.6 | Create, Read, Update, Delete (CRUD)

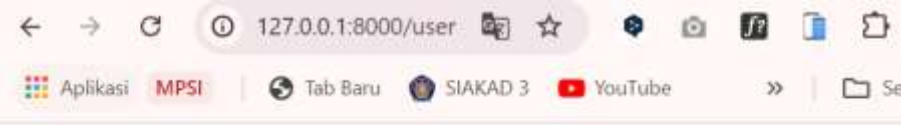
Langkah	Jawaban/Deskripsi
1.	Buka file *view* pada `user.blade.php` dan buat scritpnya menjadi seperti di bawah ini

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

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

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

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



Data User

+ Tambah User

ID	Username	Nama	ID Level Pengguna	Aksi
1	admin	Administrator	1	Ubah | Hapus
2	manager	Manager	2	Ubah | Hapus
3	staff	Staff/Kasir	3	Ubah | Hapus
13	customer 1	Pelanggan pertama	4	Ubah | Hapus
14	manager_dua	Manager2	2	Ubah | Hapus
17	manager_tiga	Manager 3	2	Ubah | Hapus
18	manager22	Manager Dua Dua	2	Ubah | Hapus
19	manager33	Manager Tiga Tiga	2	Ubah | Hapus
23	manager56	Manager55	2	Ubah | Hapus
24	manager12	Manager11	2	Ubah | Hapus

Dari hasil pengamatan diatas diatas kita menampilkan semua data pada database ke dalam halaman websiteyang nantinya akan kita lakukan proses aksi seperti ubah dan hapus.

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

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

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

```
//tambah
Route::get(uri: '/user/tambah', action: [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

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

    }

        public function tambah(): Factory|View{
            return view(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

<table>
<tr><td></td><td>

Form Tambah Data User

Username xxysnt
Nama Fardiyani Afro'ul Yasinta
Password 12345
Level ID 2

Simpan

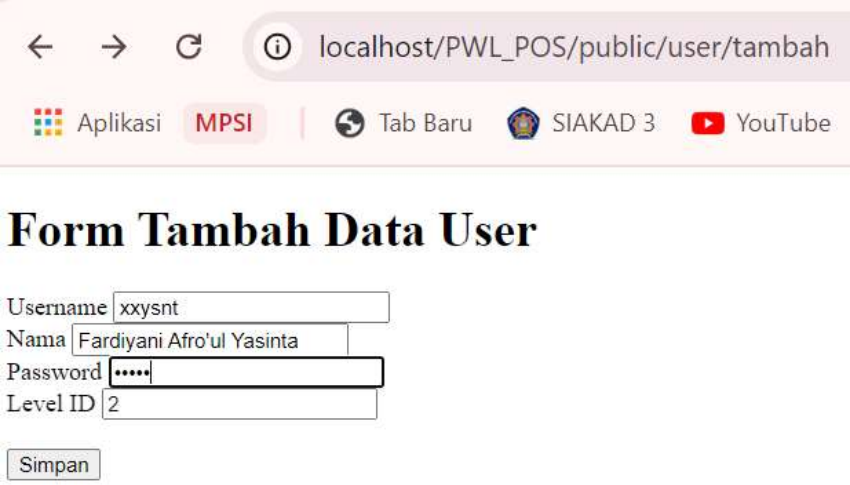
Saat melakukan simpan maka menampilkan

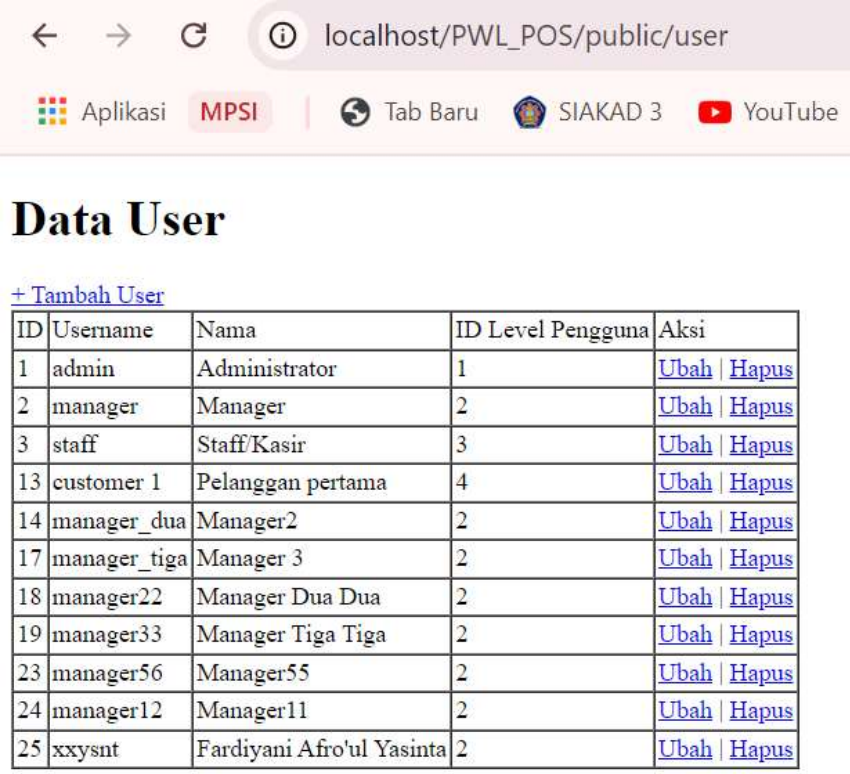
Dari hasil pengamatan diatasketika klik simpan tidak terjadi apa-apa dikarenakan belum ada konfigurasi untuk link tambah_simpan dan juga belum ada nya query untuk menyimpan ke dalam database jadi hanya sebatas form biasa.
</td></tr>
<tr><td>8.</td><td>
Tambahkan *script* pada *routes* dengan nama file `web.php`. Tambahkan seperti gambar di bawah ini

```
Route::get(uri: '/user/tambah_simpan', action: [UserController::class, 'tambah_simpan']);
```
</td></tr>
<tr><td>9.</td><td>
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

```
1 reference | 0 overrides
public function tambah_simpan(Request $request): Redirector|RedirectResponse {
    UserModel::create (attributes: [
        'username' => $request->username,
        'nama' => $request->nama,
        'password' => Hash::make(value: $request->password),
        'level_id' => $request->level_id
    ]);
    return redirect(to: '/user');
}
```
</td></tr>
</table>

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





ID	Username	Nama	ID Level Pengguna	Aksi
1	admin	Administrator	1	Ubah | Hapus
2	manager	Manager	2	Ubah | Hapus
3	staff	Staff/Kasir	3	Ubah | Hapus
13	customer 1	Pelanggan pertama	4	Ubah | Hapus
14	manager_dua	Manager2	2	Ubah | Hapus
17	manager_tiga	Manager 3	2	Ubah | Hapus
18	manager22	Manager Dua Dua	2	Ubah | Hapus
19	manager33	Manager Tiga Tiga	2	Ubah | Hapus
23	manager56	Manager55	2	Ubah | Hapus
24	manager12	Manager11	2	Ubah | Hapus
25	xxysnt	Fardiyani Afro'ul Yasinta	2	Ubah | Hapus

Dari hasil pengamatan diatas kita dapat mengetahui terkait alur penambahan data nya dimulai dari form yang nantinya akan menginput data nya lalu 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

```
resources > views > user_ubah.blade.php > html
<!DOCTYPE html>
<html lang="en">
<head>
</head>
<body>
    <h1>Form Ubah Data User</h1>
    <a href="/user">Kembali</a>
    <br><br>

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

        {{ csrf_field() }}
        {{ method_field('PUT') }}

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

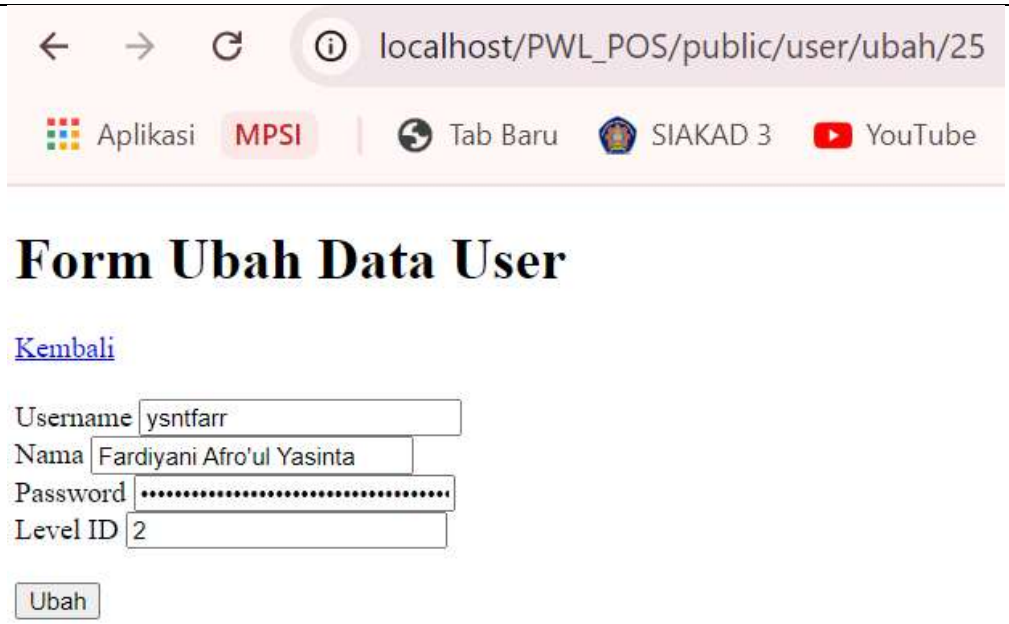
12.	Tambahkan *script* pada *routes* dengan nama file `web.php`. Tambahkan seperti gambar di bawah ini

```
//ubah
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

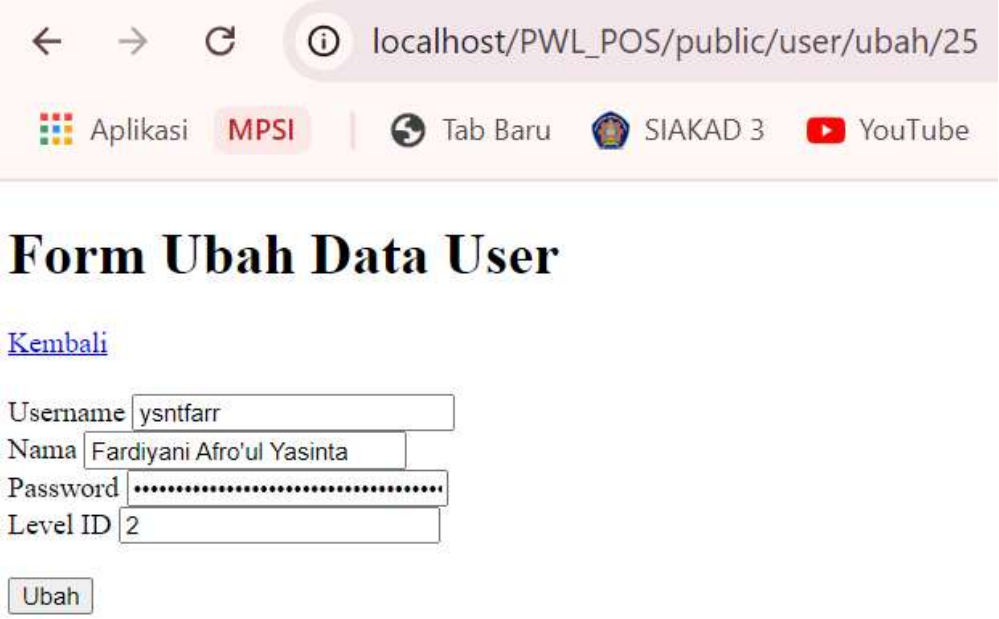
<table>
<tr>
<td></td>
<td>


Dari hasil pengamatan 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.
</td>
</tr>
<tr>
<td>15.</td>
<td>
Tambahkan script pada routes dengan nama file <code>web.php</code>. Tambahkan seperti gambar di bawah ini

</td>
</tr>
<tr>
<td>16.</td>
<td>
Tambahkan script pada controller dengan nama file <code>UserController.php</code>. Tambahkan script dalam class dan buat method baru dengan nama ubah_simpan dan diletakan di bawah method ubah seperti gambar di bawah ini
</td>
</tr>
</table>

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

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

    $user->save();

    return redirect(to: '/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



Data User

+ Tambah User

ID	Username	Nama	ID Level Pengguna	Aksi
1	admin	Administrator	1	Ubah | Hapus
2	manager	Manager	2	Ubah | Hapus
3	staff	Staff/Kasir	3	Ubah | Hapus
13	customer 1	Pelanggan pertama	4	Ubah | Hapus
14	manager_dua	Manager2	2	Ubah | Hapus
17	manager_tiga	Manager 3	2	Ubah | Hapus
18	manager22	Manager Dua Dua	2	Ubah | Hapus
19	manager33	Manager Tiga Tiga	2	Ubah | Hapus
23	manager56	Manager55	2	Ubah | Hapus
24	manager12	Manager11	2	Ubah | Hapus
25	ysntfarr	Fardiyani Afro'ul Yasinta	2	Ubah | Hapus

Dari hasil pengamatan program diatas telah selesai. Ketika melakukan perubahan data yang diinput pada form langsung di direct dan disimpan perubahan data tersebut ke dalam database sehingga bisa melakukan proses update.

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

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

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

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

        return redirect(to: '/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

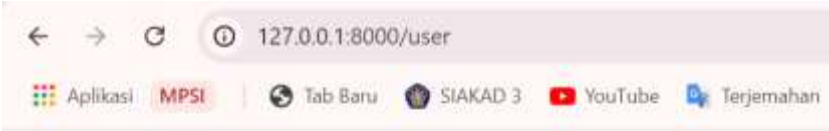
Data awal sebelum dihapus

Data User

+ Tambah User

ID	Username	Nama	ID Level Pengguna	Aksi
1	admin	Administrator	1	Ubah | Hapus
2	manager	Manager	2	Ubah | Hapus
3	staff	Staff/Kasir	3	Ubah | Hapus
13	customer 1	Pelanggan pertama	4	Ubah | Hapus
14	manager_dua	Manager2	2	Ubah | Hapus
17	manager_tiga	Manager 3	2	Ubah | Hapus
18	manager22	Manager Dua Dua	2	Ubah | Hapus
19	manager33	Manager Tiga Tiga	2	Ubah | Hapus
23	manager56	Manager55	2	Ubah | Hapus
24	manager12	Manager11	2	Ubah | Hapus
25	ysntfarr	Fardiyani Afro'ul Yasinta	2	Ubah | Hapus

Data setelah dihapus



Data User

+ Tambah User

ID	Username	Nama	ID Level Pengguna	Aksi
1	admin	Administrator	1	Ubah | Hapus
2	manager	Manager	2	Ubah | Hapus
3	staff	Staff/Kasir	3	Ubah | Hapus
13	customer 1	Pelanggan pertama	4	Ubah | Hapus
14	manager_dua	Manager2	2	Ubah | Hapus
17	manager_tiga	Manager 3	2	Ubah | Hapus
18	manager22	Manager Dua Dua	2	Ubah | Hapus
19	manager33	Manager Tiga Tiga	2	Ubah | Hapus
23	manager56	Manager55	2	Ubah | Hapus
24	manager12	Manager11	2	Ubah | Hapus

	Dari hasil pengamatan diatas kita dapat mengetahui bahwa 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*.

Praktikum Ke-2.7 | Relationships

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

```php
0 references | 0 overrides
public function level() : BelongsTo
{
    return $this->belongsTo(related: LevelModel::class, foreignKey: 'level_id', ownerKey: 'level_id');
}
}
```

```php
<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;
use Illuminate\Database\Eloquent\Relations\BelongsTo;
use Illuminate\Database\Eloquent\Relations\HasMany;


1 reference | 0 implementations
class LevelModel extends Model{
    use HasFactory;
    0 references
    protected $table = "m_level";
    0 references
    protected $primaryKey ="level_id";
    0 references | 0 overrides
    public function user(): HasMany
    {
        return $this->hasMany(related: User::class);
    }

}
```

2.	Buka file controller pada `UserController.php` dan ubah method *script* menjadi seperti di bawah ini
	`1 reference | 0 overrides`
	`public function index(): never {`
	`  $user = UserModel::with(relations: 'level')->get();`
	`  dd(vars: $user);`
	`}`

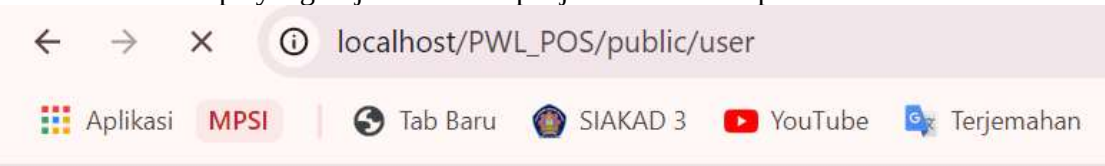
3.	Simpan kode program Langkah 2. Kemudian jalankan link pada *browser*, kemudian amati apa yang terjadi dan beri penjelasan dalam laporan
	localhost/PWL_POS/public/user
	Aplikasi MPSI Tab Baru SIAKAD 3 YouTube
	`Illuminate\Database\Eloquent\Collection {#321 ▼ // app\Http\Controllers\UserContr`
	`  #items: array:11 [▶]`
	`  #escapeWhenCastingToString: false`
	`}`
	
	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-enkode.

4.	Buka file controller pada `UserController.php` dan ubah method *script* menjadi seperti di bawah ini
	`1 reference | 0 overrides`
	`public function index(): Factory|View {`
	`  $user = UserModel::with(relations: 'level') -> get();`
	`  return view (view: 'user', data: ['data' => $user]);`
	`}`

5.	Buka file view pada `user.blade.php` dan ubah *script* menjadi seperti di bawah ini

<table>
<tr><td></td><td>

</td></tr>
<tr><td>6.</td><td>
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	admin	Administrator	1	ADM	Administrator	Ubah | Hapus
2	manager	Manager	2	MNG	Manager	Ubah | Hapus
3	staff	Staff/Kasir	3	STF	Staff/kasir	Ubah | Hapus
13	customer 1	Pelanggan pertama	4	PLG	Pelanggan	Ubah | Hapus
14	manager_dua	Manager2	2	MNG	Manager	Ubah | Hapus
17	manager_tiga	Manager 3	2	MNG	Manager	Ubah | Hapus
18	manager22	Manager Dua Dua	2	MNG	Manager	Ubah | Hapus
19	manager33	Manager Tiga Tiga	2	MNG	Manager	Ubah | Hapus
23	manager56	Manager55	2	MNG	Manager	Ubah | Hapus
24	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
</td></tr>
<tr><td>7.</td><td>Laporkan hasil Praktikum-2.7 ini dan *commit* perubahan pada *git*.</td></tr>
</table>