Jurusan Teknologi Informasi Politeknik Negeri Malang

Jobsheet-4: MODEL dan ELOQUENT ORM

Mata Pemrograman Web Lanjut(PWL)

Pengampu: Tim Ajar Desain dan Pemrograman Wep Lanjut

Maret 2024

Nama	:	Yanuar Rizki Aminudin
Nim	:	2341720030
Kelas	:	2A

Tulislah Jawaban Pada Kolom Yang tersedia di bawah ini;

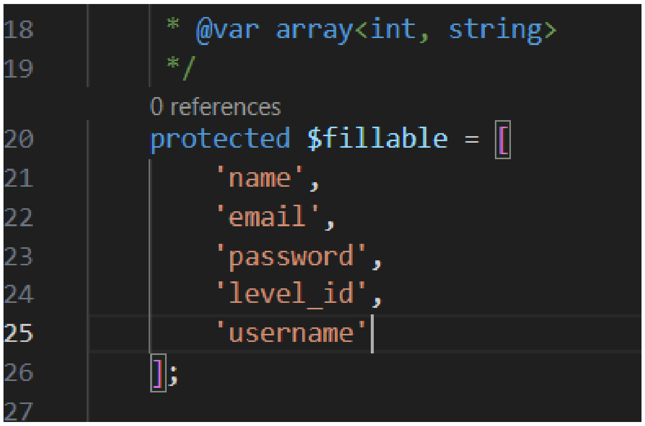
JOBSHEET-4

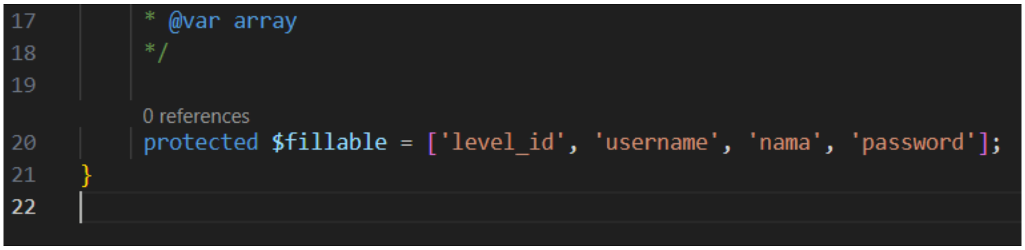
MODEL dan ELOQUENT ORM

A. PROPERTI $fillable DAN $guarded

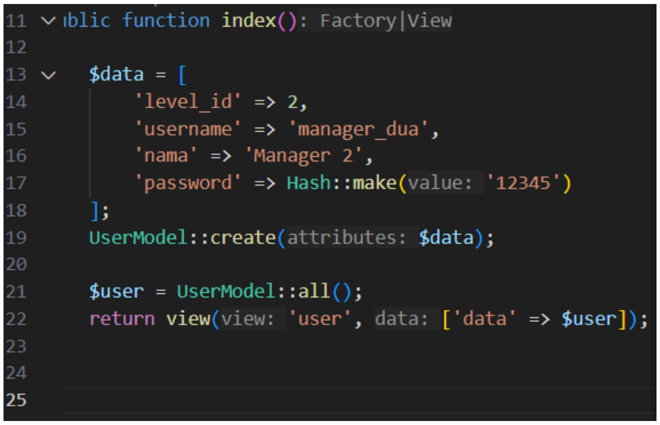
Praktikum 1 - $fillable:

Soal No	Jawaban

1	App>Models>User.php
	
	`* @var array<int, string>`
	`*/`
	`0 references`
	`protected $fillable = [`
	`'name',`
	`'email',`
	`'password',`
	`'level_id',`
	`'username'`
	`];`

2	App>Models>UserModel.php
	
	`* @var array`
	`*/`
	`0 references`
	`protected $fillable = ['level_id', 'username', 'nama', 'password'];`
	`}`

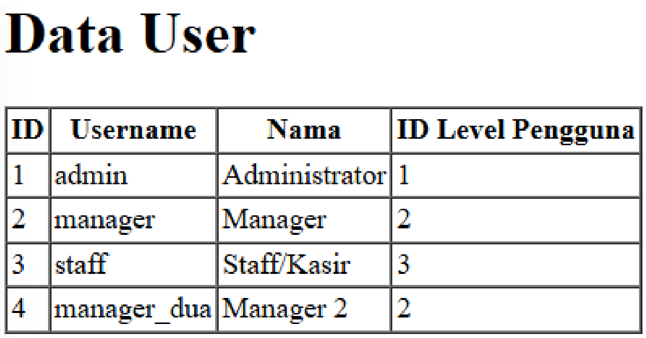
3 App>Http>Controllers>UserController.php



```
ublic function index(): Factory|View

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

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

4 http://localhost/PWL_POS/public/user

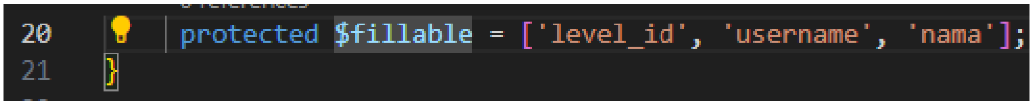
There is new data at No 4



Data User

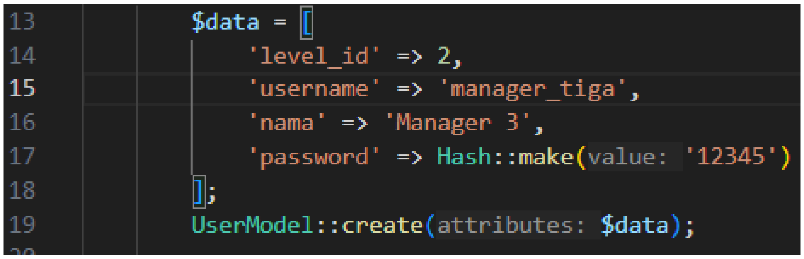
ID	Username	Nama	ID Level Pengguna
1	admin	Administrator	1
2	manager	Manager	2
3	staff	Staff/Kasir	3
4	manager_dua	Manager 2	2

5 App>Models>UserModel.php



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

6 App>Http>Controllers>UserController.php

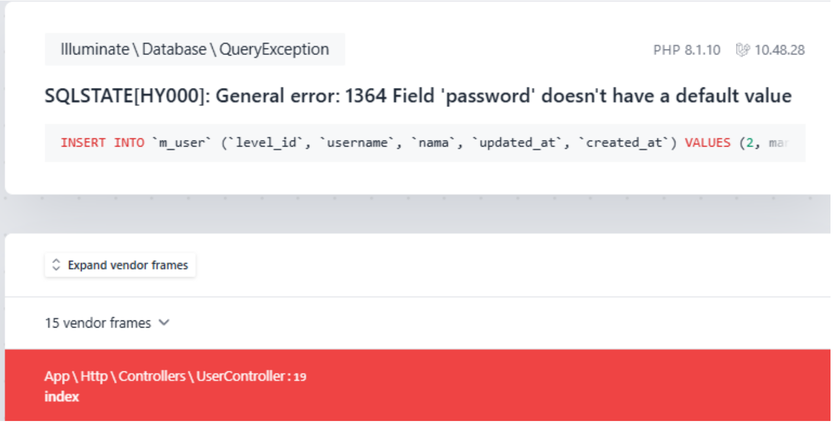


```
        $data = [
            'level_id' => 2,
            'username' => 'manager_tiga',
            'nama' => 'Manager 3',
            'password' => Hash::make(value: '12345')
        ];
        UserModel::create(attributes: $data);
```

7

App>Models>UserModel.php

http://localhost/PWL_POS/public/user



Illuminate \ Database \ QueryException

PHP 8.1.10 10.48.28

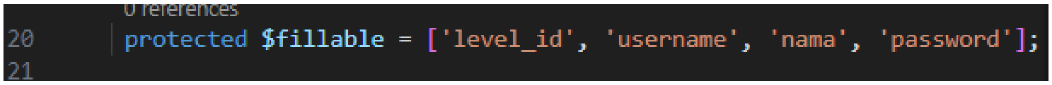
SQLSTATE[HY000]: General error: 1364 Field 'password' doesn't have a default value

```
INSERT INTO `m_user` (`level_id`, `username`, `nama`, `updated_at`, `created_at`) VALUES (2, ma
```

App \ Http \ Controllers \ UserController : 19
index

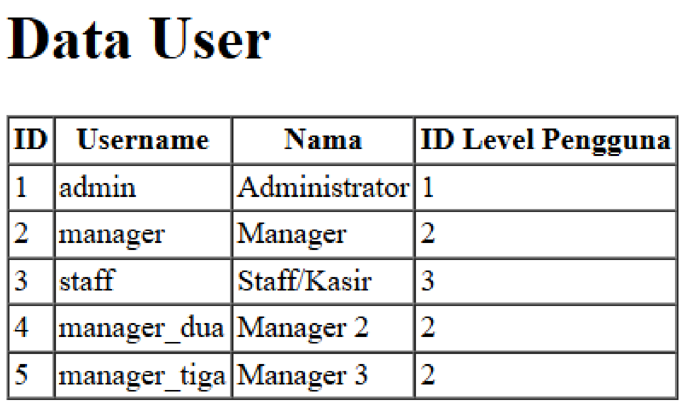
Fix for an error where passwords (model attributes) were removed

I have returned it



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

Successfully add new data no 5



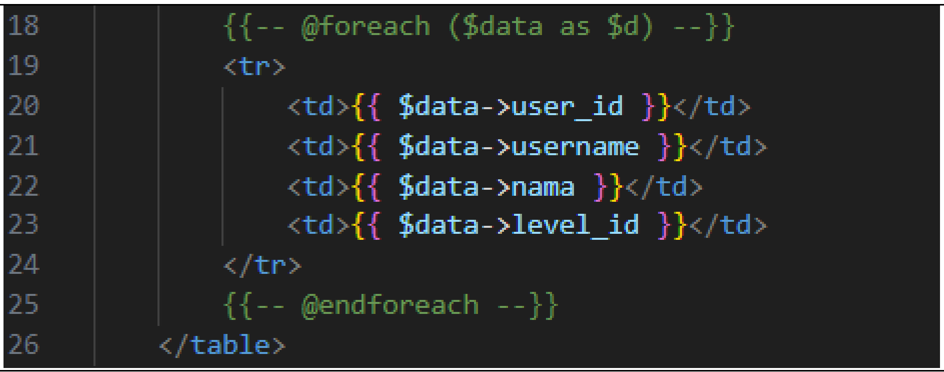
Data User

ID	Username	Nama	ID Level Pengguna
1	admin	Administrator	1
2	manager	Manager	2
3	staff	Staff/Kasir	3
4	manager_dua	Manager 2	2
5	manager_tiga	Manager 3	2

B. RETRIEVING SINGLE MODELS

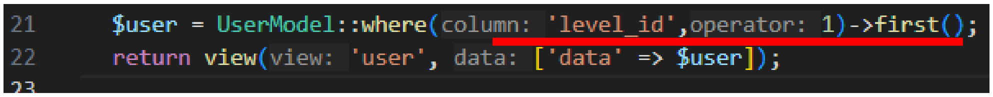
Praktikum 2.1 – Retrieving Single Models

8	resources>views>user.blade.php



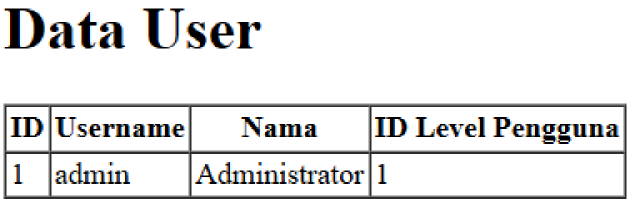
```
        {{-- @foreach ($data as $d) --}}
        <tr>
            <td>{{ $data->user_id }}</td>
            <td>{{ $data->username }}</td>
            <td>{{ $data->nama }}</td>
            <td>{{ $data->level_id }}</td>
        </tr>
        {{-- @endforeach --}}
    </table>
```

9 App>Http>Controllers>UserController.php



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

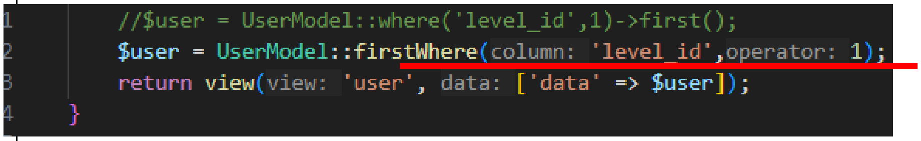
Mencari semua data dengan `level_id = 1`, lalu mengambil satu data pertama.



Data User

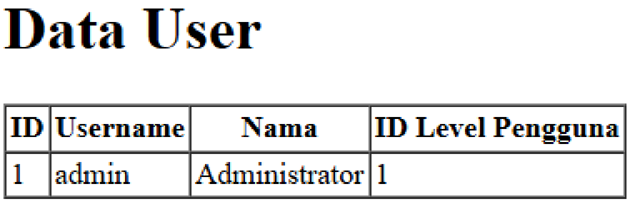
ID	Username	Nama	ID Level Pengguna
1	admin	Administrator	1

10 App>Http>Controllers>UserController.php



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

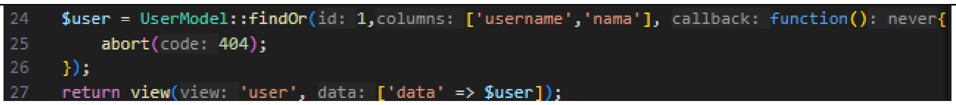
Cara singkat dari `where()->first()`, langsung mengambil satu data pertama yang cocok.



Data User

ID	Username	Nama	ID Level Pengguna
1	admin	Administrator	1

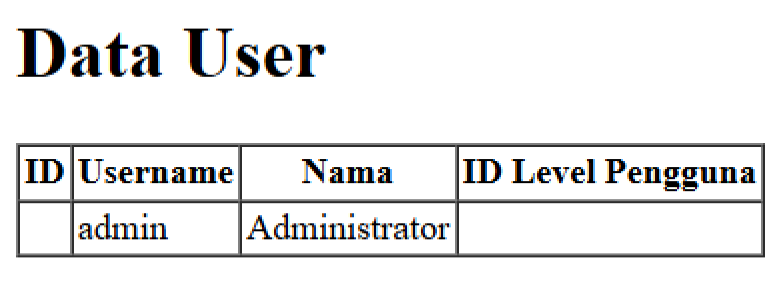
11



```
    $user = UserModel::findOr(id: 1,columns: ['username','nama'], callback: function(): never{
        abort(code: 404);
    });
    return view(view: 'user', data: ['data' => $user]);
```

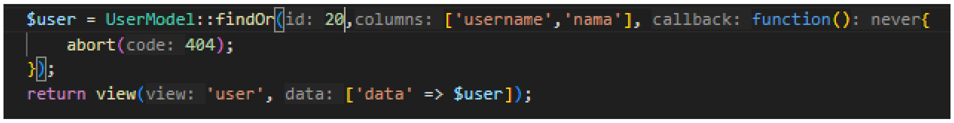
Di gunanakan untuk menutupi halaman yang error menjadi 404 | NOT FOUND

Lebih nyaman untuk user saat mengakses data yg tidak ditemukan

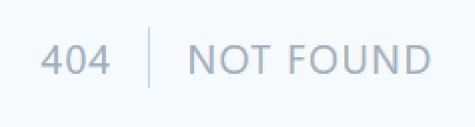


Data User

ID	Username	Nama	ID Level Pengguna
	admin	Administrator	



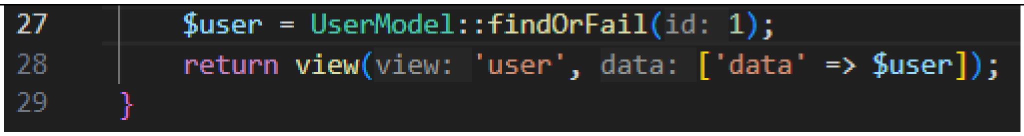
```
$user = UserModel::findOr(id: 20,columns: ['username','nama'], callback: function(): never{
    abort(code: 404);
});
return view(view: 'user', data: ['data' => $user]);
```



404 | NOT FOUND

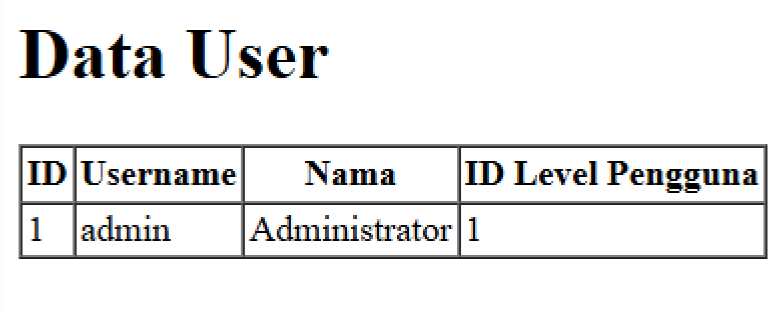
Praktikum 2.2 – Not Found Exceptions

12



```
        $user = UserModel::findOrFail(id: 1);
        return view(view: 'user', data: ['data' => $user]);
    }
```

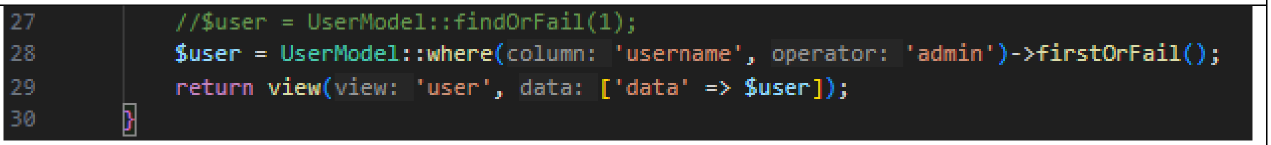
findOrFail secara otomatis menampilkan 404 | NOT FOUND tampa perinta abort(404)



Data User

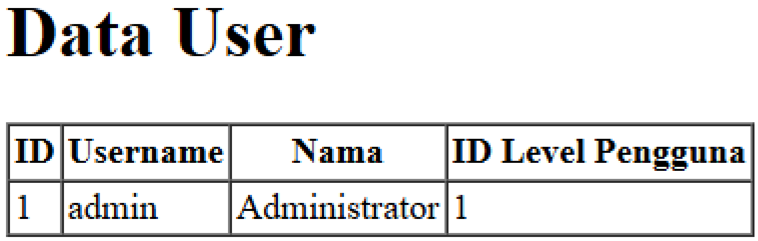
ID	Username	Nama	ID Level Pengguna
1	admin	Administrator	1

13



```
        //$user = UserModel::findOrFail(1);
        $user = UserModel::where(column: 'username', operator: 'admin')->firstOrFail();
        return view(view: 'user', data: ['data' => $user]);
    }
```

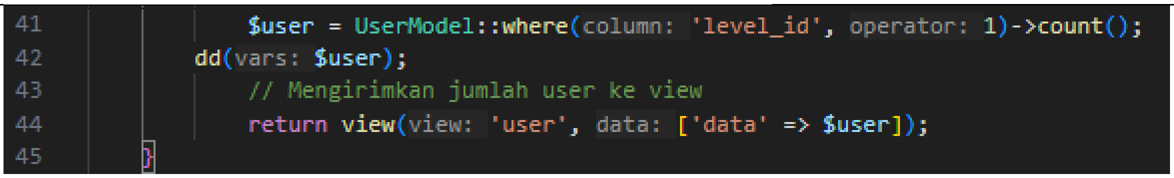
Kalau sebelumnya berdasarkan id maka yg ini melakukan pencarian melalui username jika username manager9 akan terjadi error 404 | NOT FOUND karena tidak ada di database



Data User

ID	Username	Nama	ID Level Pengguna
1	admin	Administrator	1

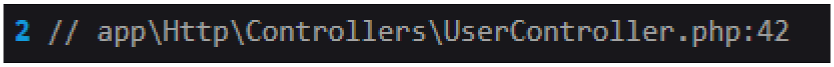
14



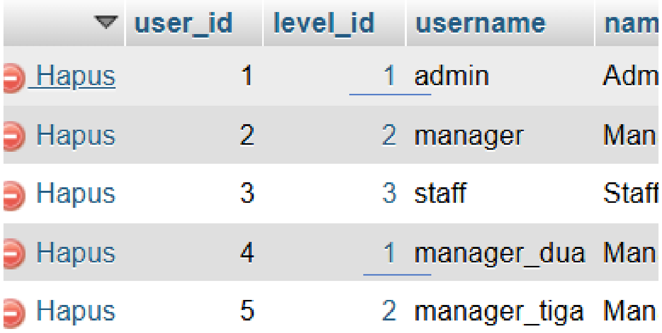
```
            $user = UserModel::where(column: 'level_id', operator: 1)->count();
        dd(vars: $user);
            // Mengirimkan jumlah user ke view
            return view(view: 'user', data: ['data' => $user]);
    }
```

terdapat dua level_id yg sama di database

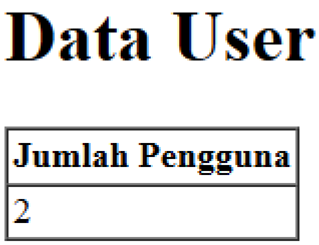
fungsi count ini hanya menhitung level id yg ada di database



```
2 // app\Http\Controllers\UserController.php:42
```



	user_id	level_id	username	nam
Hapus	1	1	admin	Adm
Hapus	2	2	manager	Man
Hapus	3	3	staff	Staff
Hapus	4	1	manager_dua	Man
Hapus	5	2	manager_tiga	Man

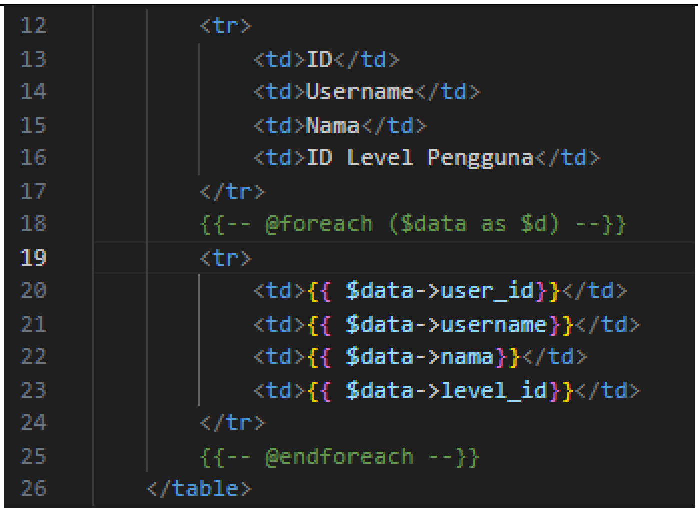


Data User

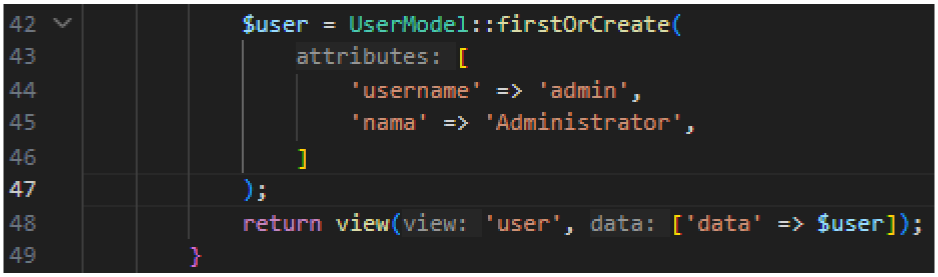
Jumlah Pengguna
2

Praktikum 2.4 – *Retreiving or Creating Models*

15

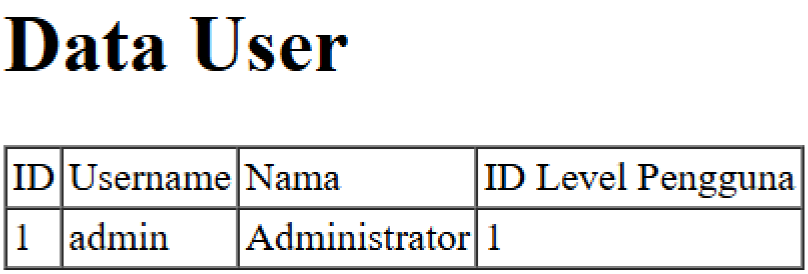


```
        <tr>
            <td>ID</td>
            <td>Username</td>
            <td>Nama</td>
            <td>ID Level Pengguna</td>
        </tr>
        {{-- @foreach ($data as $d) --}}
        <tr>
            <td>{{ $data->user_id}}</td>
            <td>{{ $data->username}}</td>
            <td>{{ $data->nama}}</td>
            <td>{{ $data->level_id}}</td>
        </tr>
        {{-- @endforeach --}}
    </table>
```



```
        $user = UserModel::firstOrCreate(
            attributes: [
                'username' => 'admin',
                'nama' => 'Administrator',
            ]
        );
        return view(view: 'user', data: ['data' => $user]);
    }
```

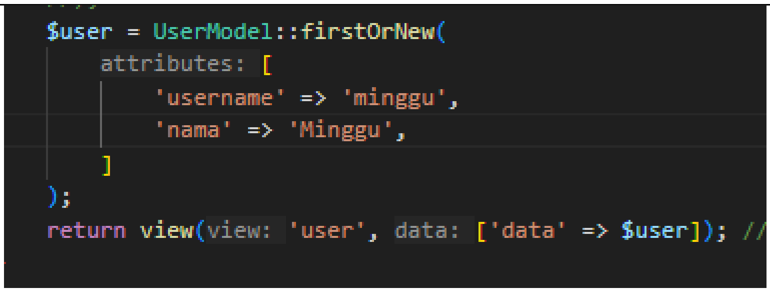
Mencari data pertama yang di minta. Jika tidak ada, data baru akan dibuat dan disimpan.



Data User

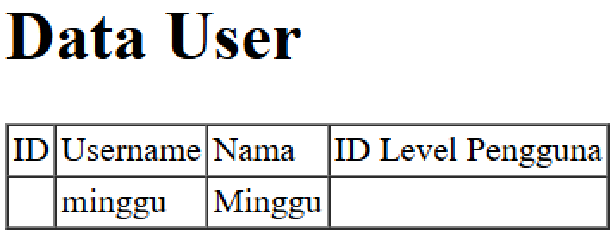
ID	Username	Nama	ID Level Pengguna
1	admin	Administrator	1

16



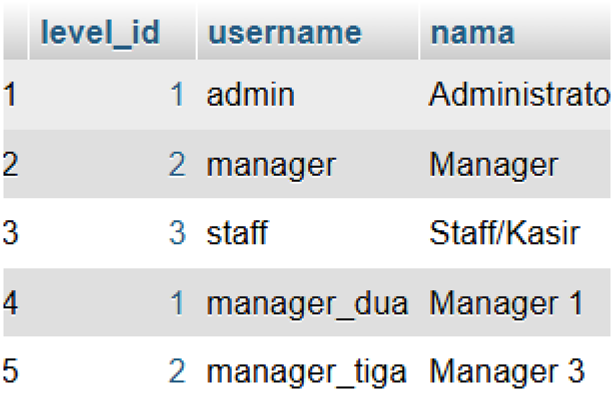
```
$user = UserModel::firstOrNew(
    attributes: [
        'username' => 'minggu',
        'nama' => 'Minggu',
    ]
);
return view(view: 'user', data: ['data' => $user]); //
```

Kalau firstOrNew membuat data baru tapi tidak menyimpan di database



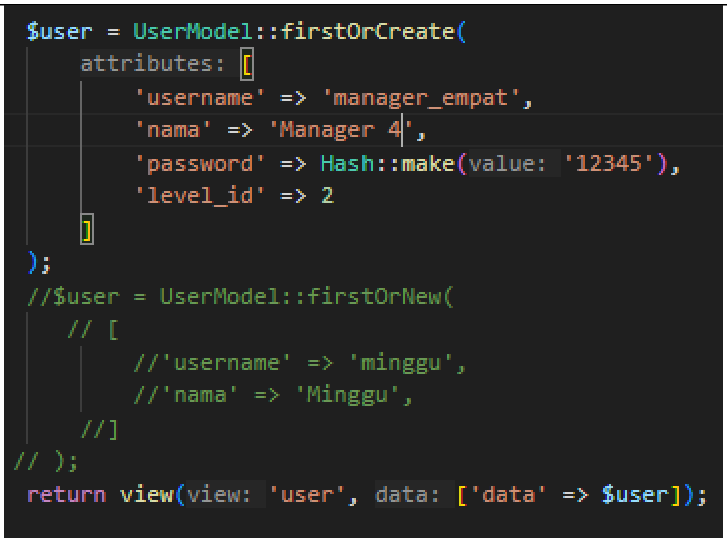
Data User

ID	Username	Nama	ID Level Pengguna
	minggu	Minggu	



	level_id	username	nama
1	1	admin	Administrato
2	2	manager	Manager
3	3	staff	Staff/Kasir
4	1	manager_dua	Manager 1
5	2	manager_tiga	Manager 3

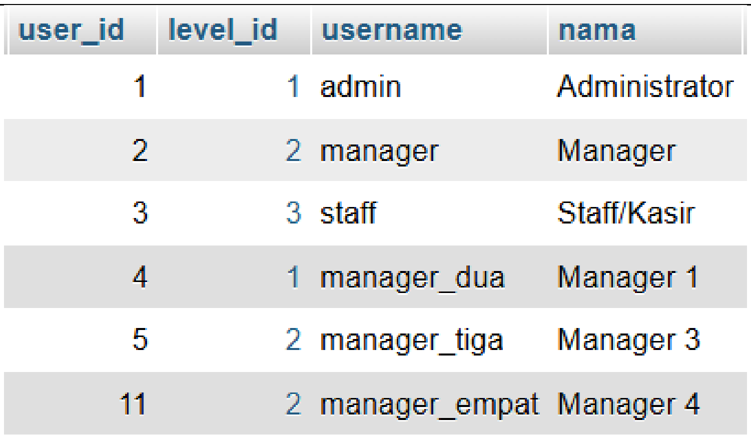
17



```
$user = UserModel::firstOrCreate(
    attributes: [
        'username' => 'manager_empat',
        'nama' => 'Manager 4',
        'password' => Hash::make(value: '12345'),
        'level_id' => 2
    ]
);
//$user = UserModel::firstOrNew(
   // [
        //'username' => 'minggu',
        //'nama' => 'Minggu',
    //]
// );
return view(view: 'user', data: ['data' => $user]);
```

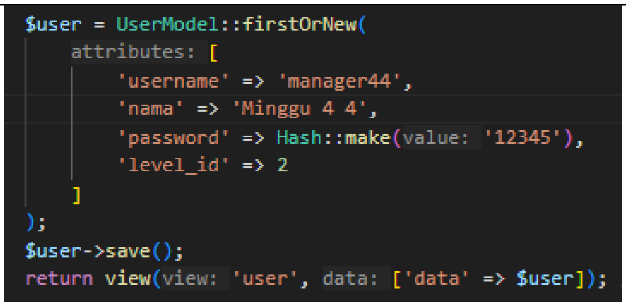
kalau ini kebalikan dari firstOrNew

dia menyimpan ke database bisa di lihat di user id 11



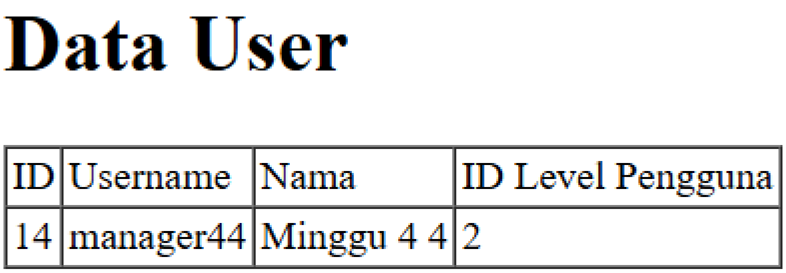
user_id	level_id	username	nama
1	1	admin	Administrator
2	2	manager	Manager
3	3	staff	Staff/Kasir
4	1	manager_dua	Manager 1
5	2	manager_tiga	Manager 3
11	2	manager_empat	Manager 4

18.



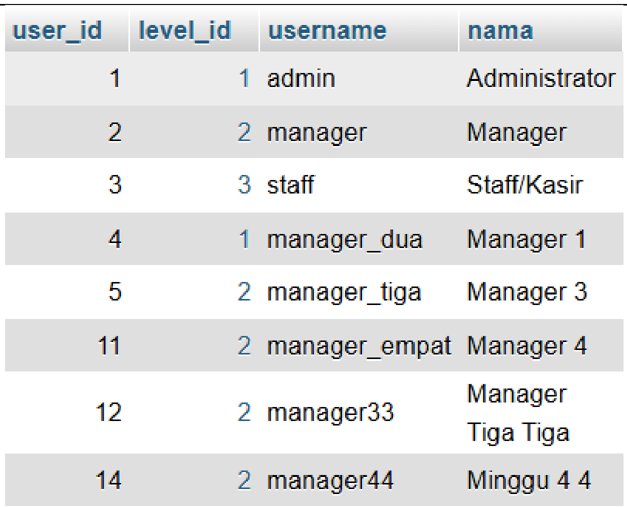
```
$user = UserModel::firstOrNew(
    attributes: [
        'username' => 'manager44',
        'nama' => 'Minggu 4 4',
        'password' => Hash::make(value: '12345'),
        'level_id' => 2
    ]
);
$user->save();
return view(view: 'user', data: ['data' => $user]);
```

Kalau yang ini setelah di buat data baru kemudian di simpan ke database karena ada method $user->save();



Data User

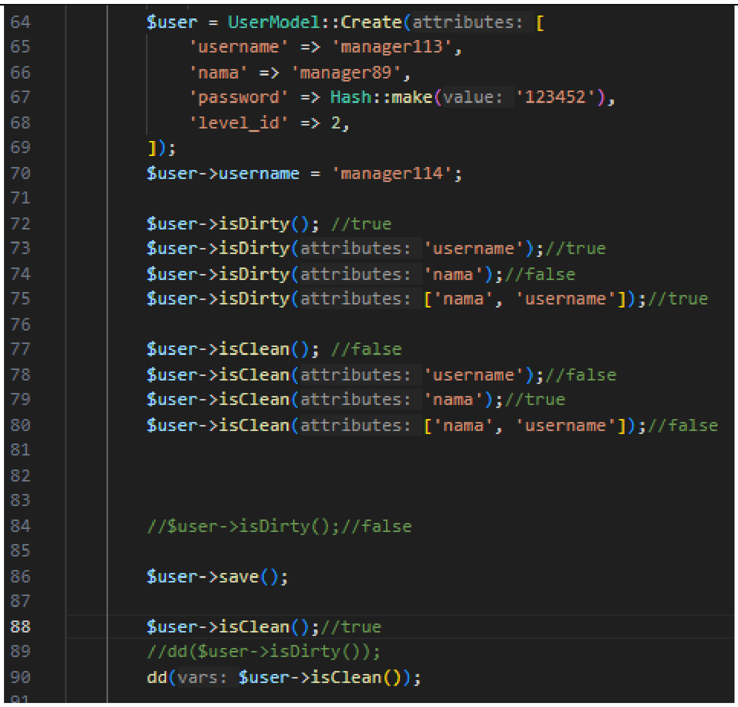
ID	Username	Nama	ID Level Pengguna
14	manager44	Minggu 4 4	2



user_id	level_id	username	nama
1	1	admin	Administrator
2	2	manager	Manager
3	3	staff	Staff/Kasir
4	1	manager_dua	Manager 1
5	2	manager_tiga	Manager 3
11	2	manager_empat	Manager 4
12	2	manager33	Manager Tiga Tiga
14	2	manager44	Minggu 4 4

Praktikum 2.5 – *Attribute Changes*

19.



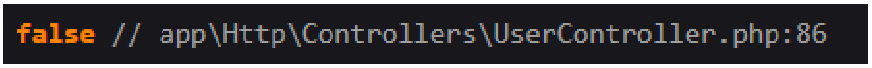
```php
        $user = UserModel::Create(attributes: [
            'username' => 'manager113',
            'nama' => 'manager89',
            'password' => Hash::make(value: '123452'),
            'level_id' => 2,
        ]);
        $user->username = 'manager114';

        $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->isDirty();//false

        $user->save();

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

Pengamatan say ajika menjalan kan

dd($user->isDirty()); bernilai false

Mengecek apakah ada perubahan data **sebelum disimpan**

true jika ada perubahan dan **false** jika sudah disimpan.

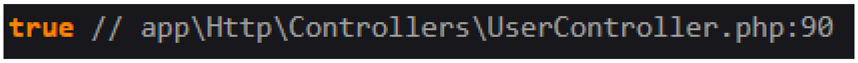


```
false // app\Http\Controllers\UserController.php:86
```

dd($user->isClean()); bernlai true

Mengecek apakah ada perubahan data **sebelum disimpan**

false jika ada perubahan dan **true** jika sudah disimpan.



```
true // app\Http\Controllers\UserController.php:90
```

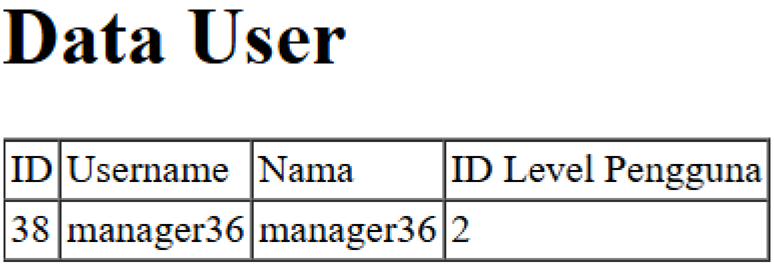
ilustrasi :

‘sername’ => ‘manager35,

Dan

$user->username = ‘manager36’;

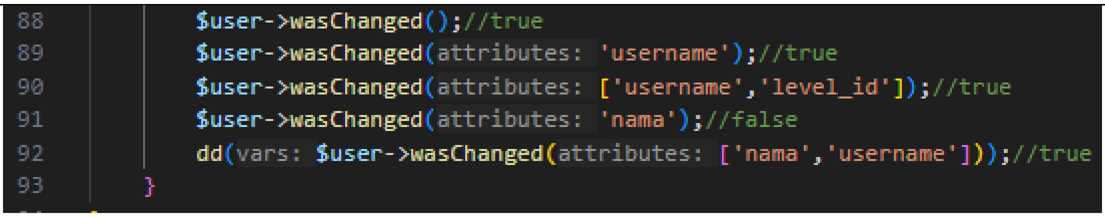
Itu akan merubah yang awalnya manager35 jadi manager36,



Data User

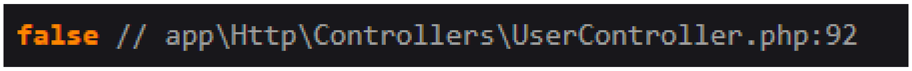
ID	Username	Nama	ID Level Pengguna
38	manager36	manager36	2

20.



```
            $user->wasChanged();//true
            $user->wasChanged(attributes: 'username');//true
            $user->wasChanged(attributes: ['username','level_id']);//true
            $user->wasChanged(attributes: 'nama');//false
            dd(vars: $user->wasChanged(attributes: ['nama','username']));//true
        }
```

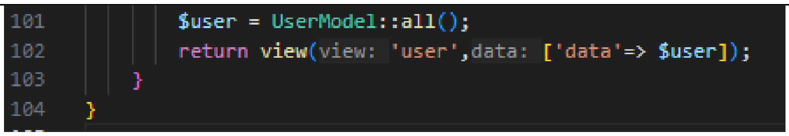
Mengecek apakah suatu atribut **telah berubah setelah disimpan**
true jika atribut mengalami perubahan saat save() dipanggil
false jika atribut tidak mengalami perubahan



```
false // app\Http\Controllers\UserController.php:92
```

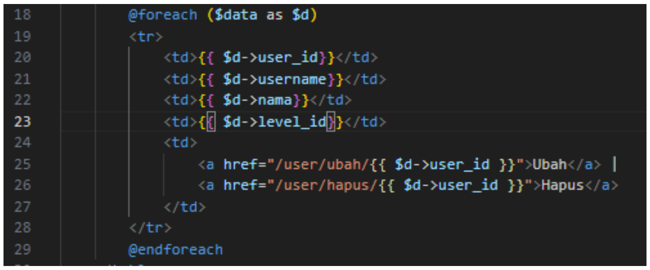
Praktikum 2.6 – *Create, Read, Update, Delete (CRUD)*

21.

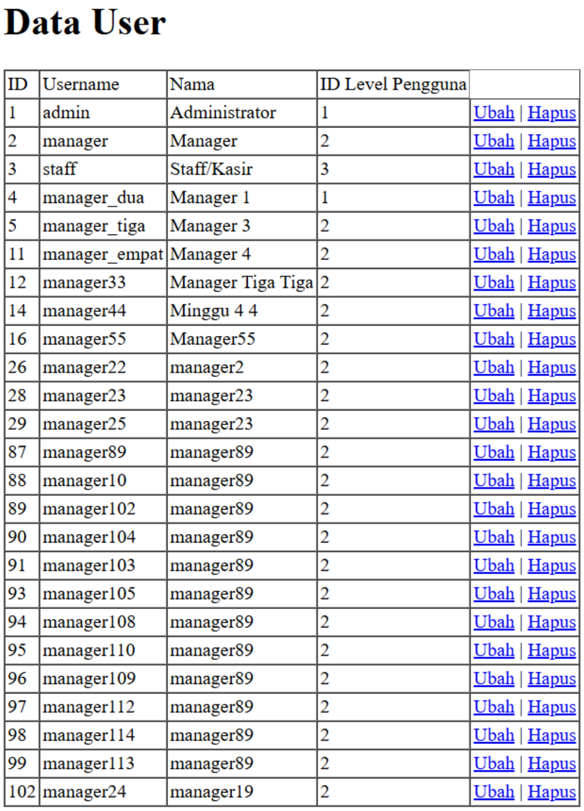


```
            $user = UserModel::all();
            return view(view: 'user',data: ['data'=> $user]);
        }
    }
```

Menambahkan tombol uabah dan hapus tapi masih belum bisa di jalankan



```
@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
```

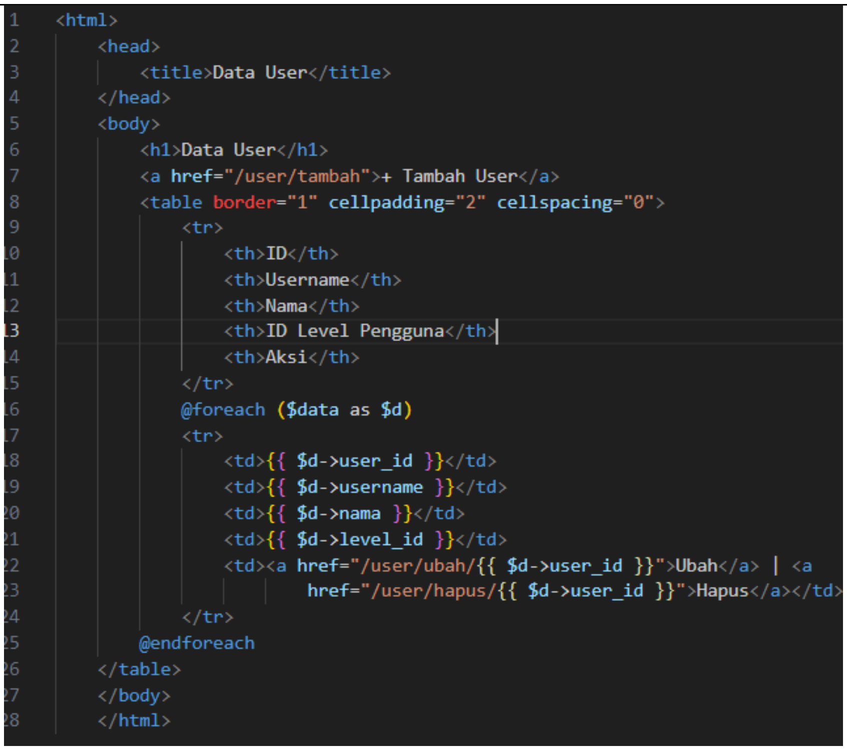


Data User

ID	Username	Nama	ID Level Pengguna	
1	admin	Administrator	1	Ubah | Hapus
2	manager	Manager	2	Ubah | Hapus
3	staff	Staff/Kasir	3	Ubah | Hapus
4	manager_dua	Manager 1	1	Ubah | Hapus
5	manager_tiga	Manager 3	2	Ubah | Hapus
11	manager_empat	Manager 4	2	Ubah | Hapus
12	manager33	Manager Tiga Tiga	2	Ubah | Hapus
14	manager44	Minggu 4 4	2	Ubah | Hapus
16	manager55	Manager55	2	Ubah | Hapus
26	manager22	manager2	2	Ubah | Hapus
28	manager23	manager23	2	Ubah | Hapus
29	manager25	manager23	2	Ubah | Hapus
87	manager89	manager89	2	Ubah | Hapus
88	manager10	manager89	2	Ubah | Hapus
89	manager102	manager89	2	Ubah | Hapus
90	manager104	manager89	2	Ubah | Hapus
91	manager103	manager89	2	Ubah | Hapus
93	manager105	manager89	2	Ubah | Hapus
94	manager108	manager89	2	Ubah | Hapus
95	manager110	manager89	2	Ubah | Hapus
96	manager109	manager89	2	Ubah | Hapus
97	manager112	manager89	2	Ubah | Hapus
98	manager114	manager89	2	Ubah | Hapus
99	manager113	manager89	2	Ubah | Hapus
102	manager24	manager19	2	Ubah | Hapus

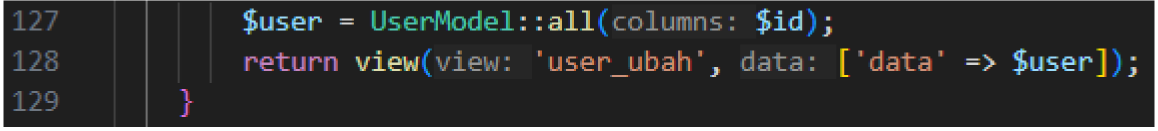
Praktikum 2.6 – *Create, Read, Update, Delete (CRUD)*

22. User.blade.php

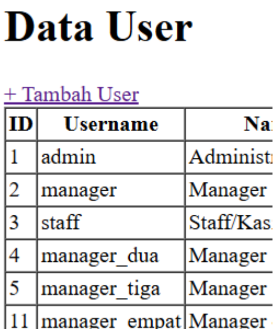


```
<html>
    <head>
        <title>Data User</title>
    </head>
    <body>
        <h1>Data User</h1>
        <a href="/user/tambah">+ Tambah User</a>
        <table border="1" cellpadding="2" cellspacing="0">
            <tr>
                <th>ID</th>
                <th>Username</th>
                <th>Nama</th>
                <th>ID Level Pengguna</th>
                <th>Aksi</th>
            </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>
```

UserController.php



```
127            $user = UserModel::all(columns: $id);
128            return view(view: 'user_ubah', data: ['data' => $user]);
129        }
```

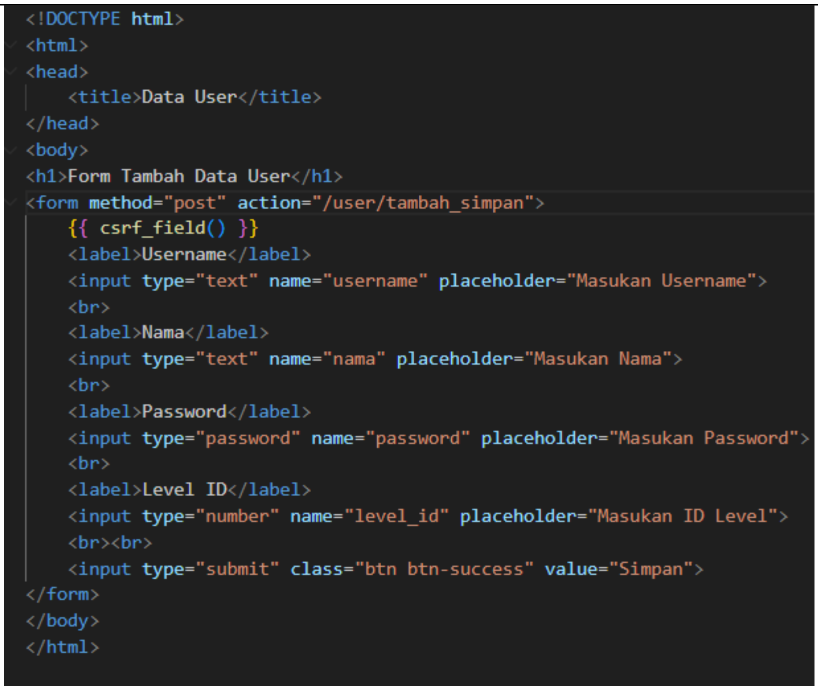


Data User

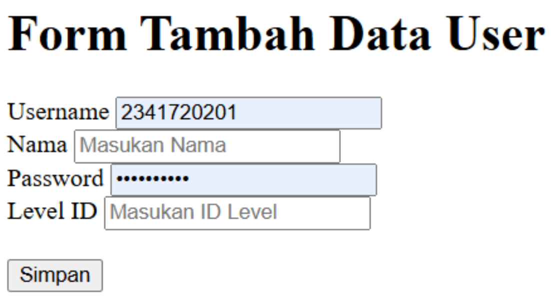
+ Tambah User

ID	Username	Na
1	admin	Administ
2	manager	Manager
3	staff	Staff/Kas
4	manager_dua	Manager
5	manager_tiga	Manager
11	manager_empat	Manager

23. user_tambah.blade.php



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

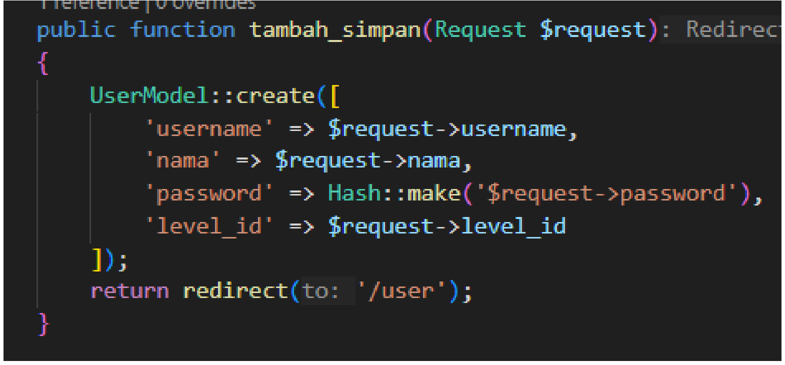


Form Tambah Data User

Username 2341720201
Nama Masukan Nama
Password ••••••••••
Level ID Masukan ID Level

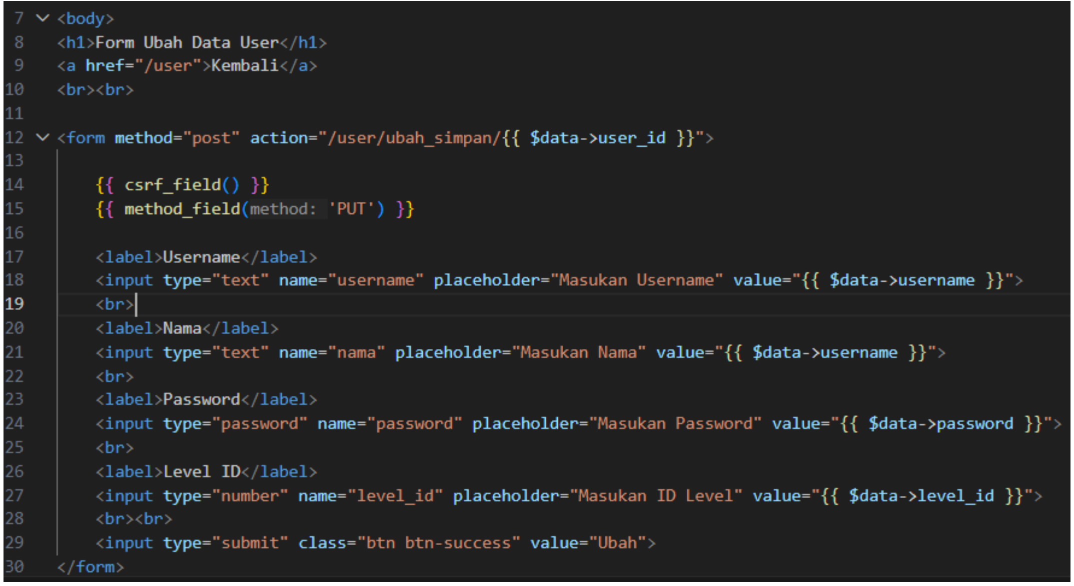
Simpan

24. UserController.php.



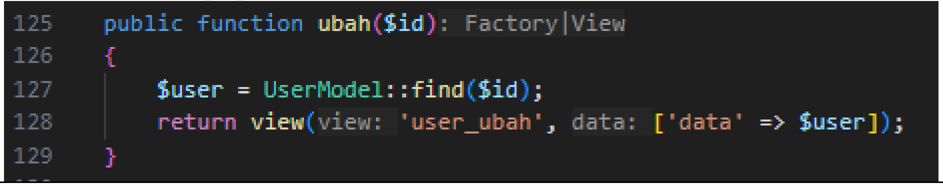
```
1 reference | 0 overrides
public function tambah_simpan(Request $request): Redirec
{
    UserModel::create([
        'username' => $request->username,
        'nama' => $request->nama,
        'password' => Hash::make('$request->password'),
        'level_id' => $request->level_id
    ]);
    return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

25. user_ubah.blade.php



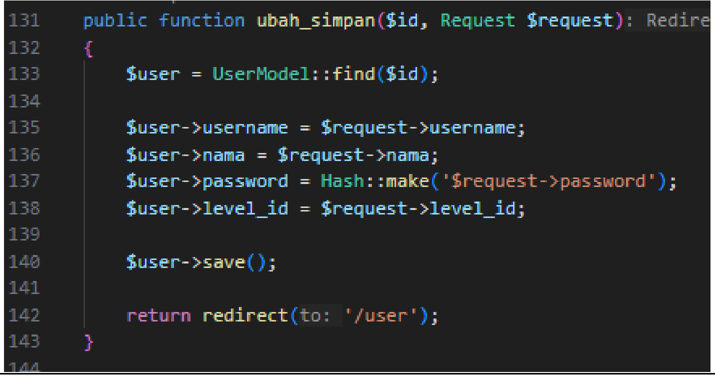
```
<body>
<h1>Form Ubah Data User</h1>
<a href="/user">Kembali</a>
<br><br>

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

    {{ csrf_field() }}
    {{ method_field(method: '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>
```

26. UserController.php.



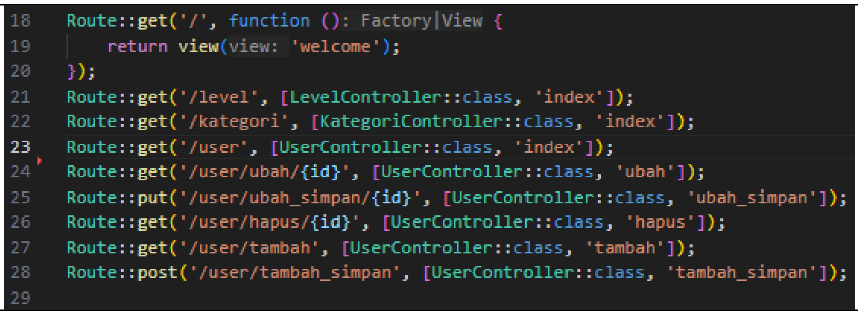
```
public function ubah($id): Factory|View
{
    $user = UserModel::find($id);
    return view(view: 'user_ubah', data: ['data' => $user]);
}
```

27. UserController.php.



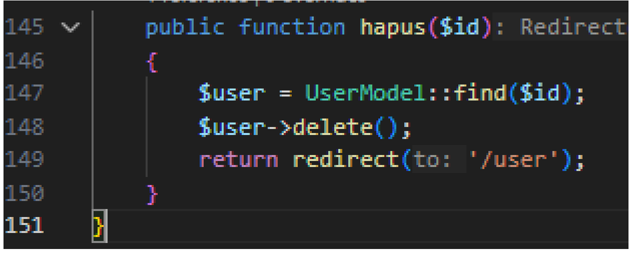
```
public function ubah_simpan($id, Request $request): Redire
{
    $user = UserModel::find($id);

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

    $user->save();

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

28. web.php.



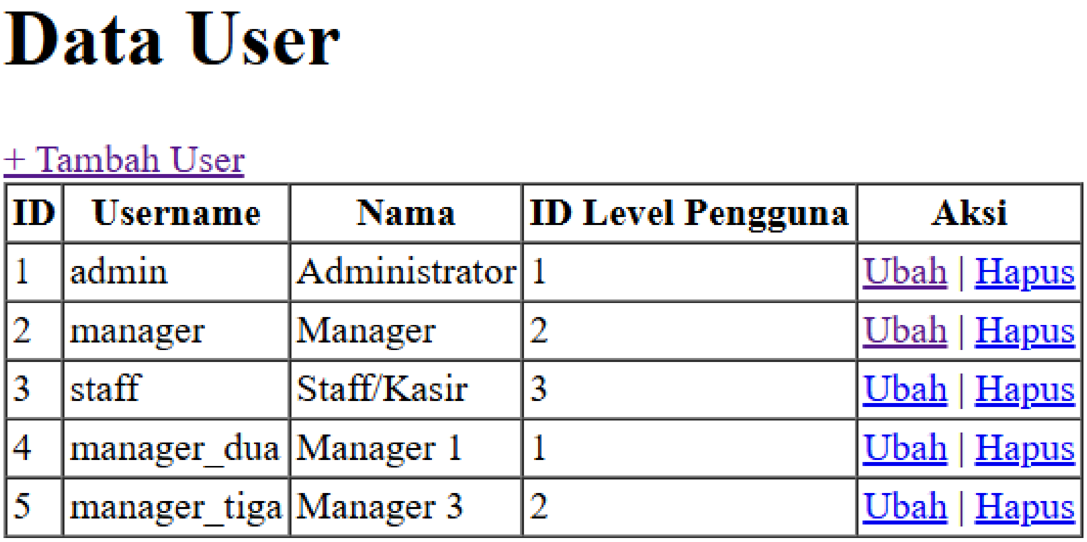
```
Route::get('/', function (): Factory|View {
    return view(view: 'welcome');
});
Route::get('/level', [LevelController::class, 'index']);
Route::get('/kategori', [KategoriController::class, 'index']);
Route::get('/user', [UserController::class, 'index']);
Route::get('/user/ubah/{id}', [UserController::class, 'ubah']);
Route::put('/user/ubah_simpan/{id}', [UserController::class, 'ubah_simpan']);
Route::get('/user/hapus/{id}', [UserController::class, 'hapus']);
Route::get('/user/tambah', [UserController::class, 'tambah']);
Route::post('/user/tambah_simpan', [UserController::class, 'tambah_simpan']);
```

29. UserController.php



```
    public function hapus($id): Redirect
    {
        $user = UserModel::find($id);
        $user->delete();
        return redirect(to: '/user');
    }
}
```

Data Setelah di hapus



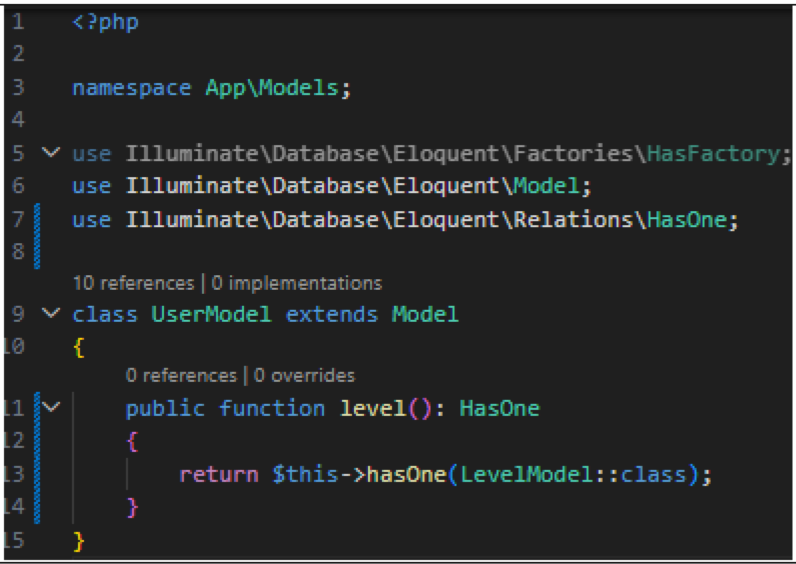
Data User

+ 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
4	manager_dua	Manager 1	1	Ubah | Hapus
5	manager_tiga	Manager 3	2	Ubah | Hapus

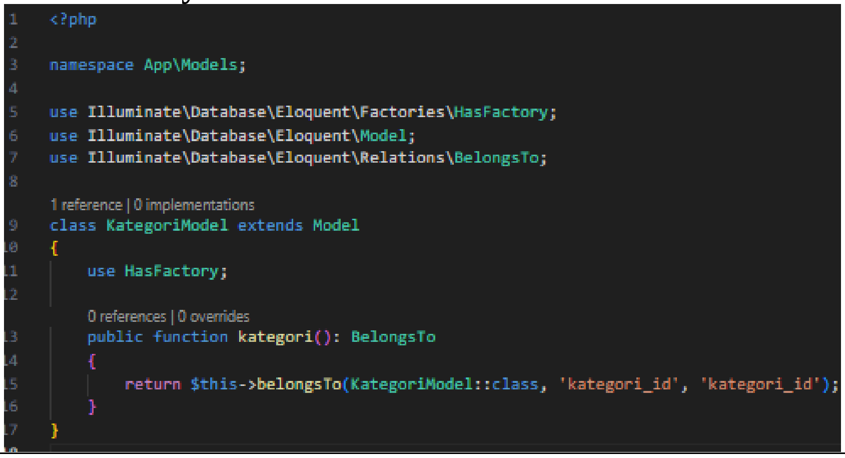
Praktikum 2.7 – *Relationships*

30. One-to-one



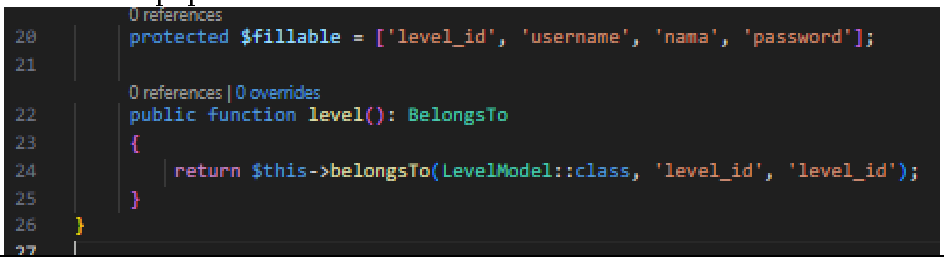
```php
<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;
use Illuminate\Database\Eloquent\Relations\HasOne;

10 references | 0 implementations
class UserModel extends Model
{
    0 references | 0 overrides
    public function level(): HasOne
    {
        return $this->hasOne(LevelModel::class);
    }
}
```

31. **One to Many**



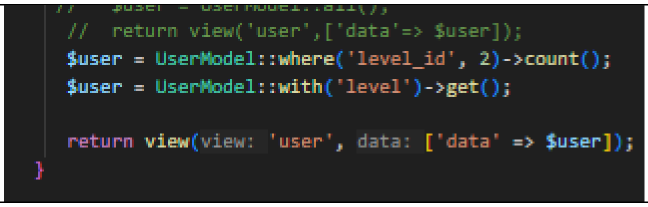
```php
<?php

namespace App\Models;

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

1 reference | 0 implementations
class KategoriModel extends Model
{
    use HasFactory;

    0 references | 0 overrides
    public function kategori(): BelongsTo
    {
        return $this->belongsTo(KategoriModel::class, 'kategori_id', 'kategori_id');
    }
}
```

32. UserModel.php



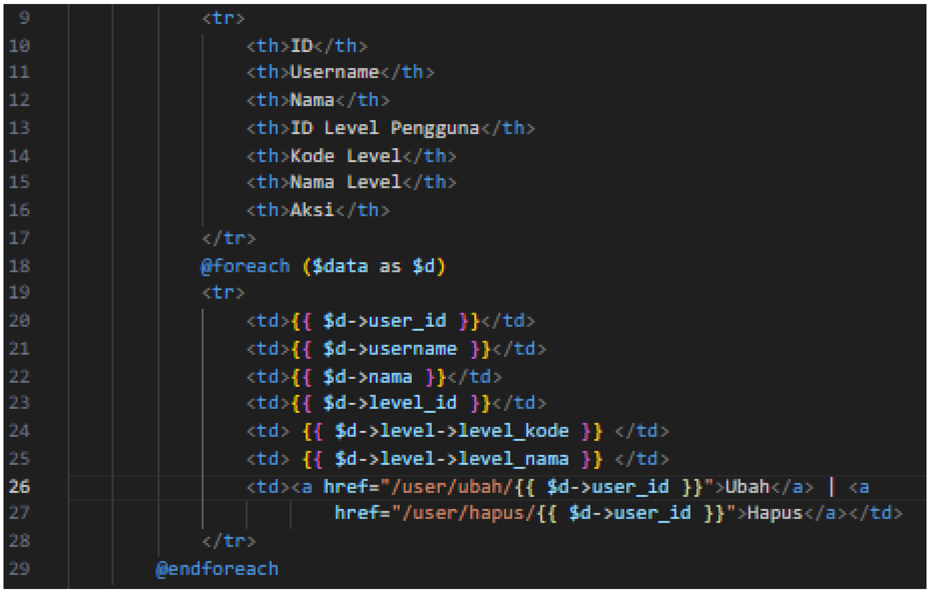
```php
        0 references
        protected $fillable = ['level_id', 'username', 'nama', 'password'];

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

33.



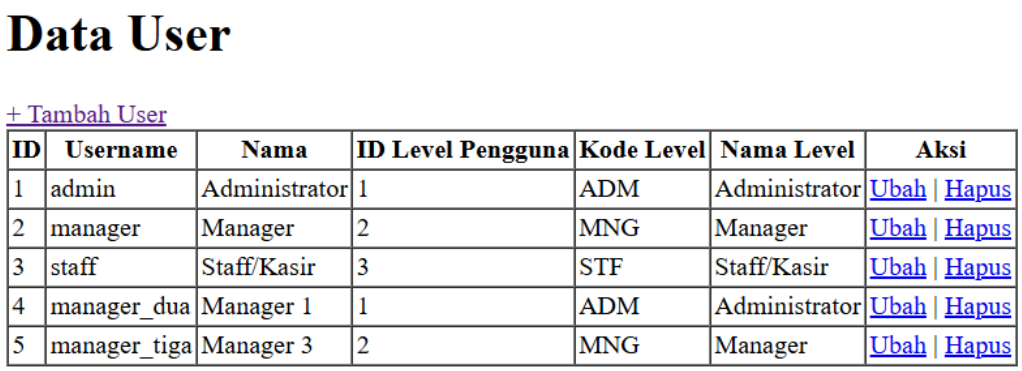
```
    //   $user = UserModel::all();
    //  return view('user',['data'=> $user]);
    $user = UserModel::where('level_id', 2)->count();
    $user = UserModel::with('level')->get();

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

34.



```
            <tr>
                <th>ID</th>
                <th>Username</th>
                <th>Nama</th>
                <th>ID Level Pengguna</th>
                <th>Kode Level</th>
                <th>Nama Level</th>
                <th>Aksi</th>
            </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
```

Menambahakan beberapa variabel di langkah 5 level_kode dan level_nama



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
4	manager_dua	Manager 1	1	ADM	Administrator	Ubah | Hapus
5	manager_tiga	Manager 3	2	MNG	Manager	Ubah | Hapus