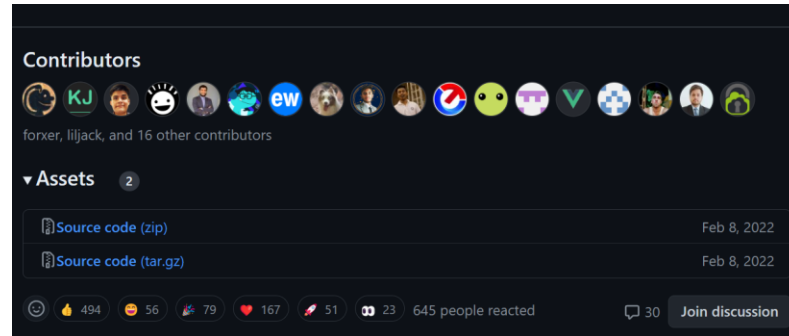


Nama : Muhammad Ridlo Febrio Putra

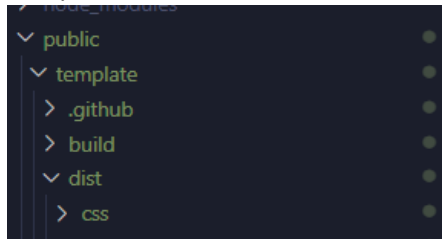
Kelas : 2H

NIM : 2241720098

1. Penggunaan template AdminLTE Bootstrap Admin Dashboard Template dengan langkah pertama akses <https://adminlte.io/> , lalu klik download pada source code (zip)



2. Lakukan extract All dan ubah nama folder AdminLTE-3.2.0 menjadi template, folder template pindahkan ke laragon/www/PWL_2024/public. Buka folder public pada VSCode maka akan tampil folder baru bernama template



3. Copy isi public/template/index2.html pada welcome.blade.php lakukan edit agar dapat diload, edit href = "{{ asset ('template/.....') }}" dan src= " {{ asset ('template/.....') }}" sesuai gambar berikut



4. Scroll ke bagian paling bawah dan sesuaikan dengan kode berikut

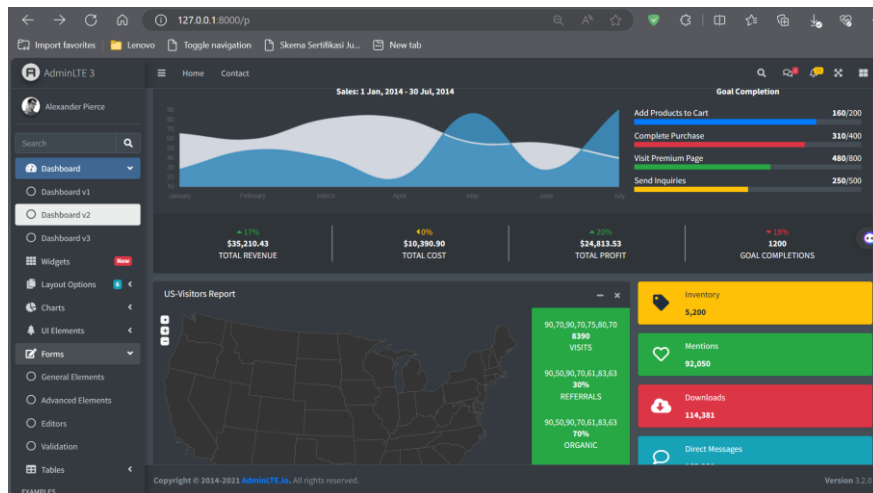


5. Jalankan pada browser

Nama : Muhammad Ridlo Febrio Putra

Kelas : 2H

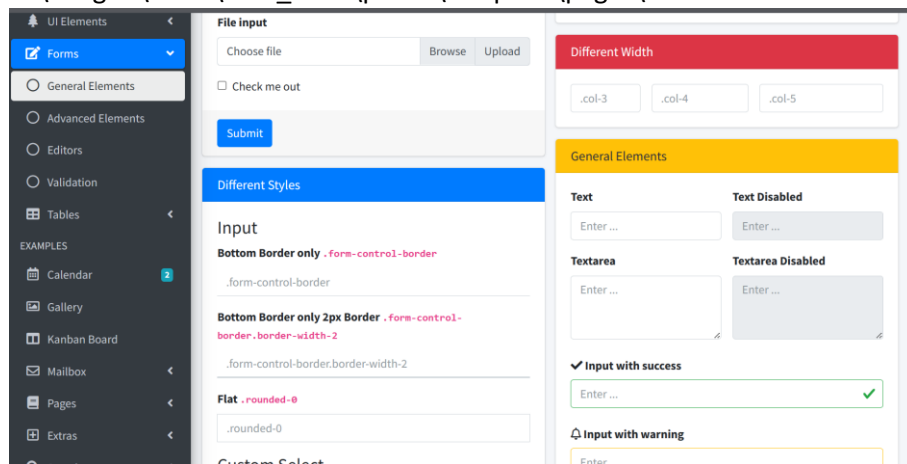
NIM : 2241720098



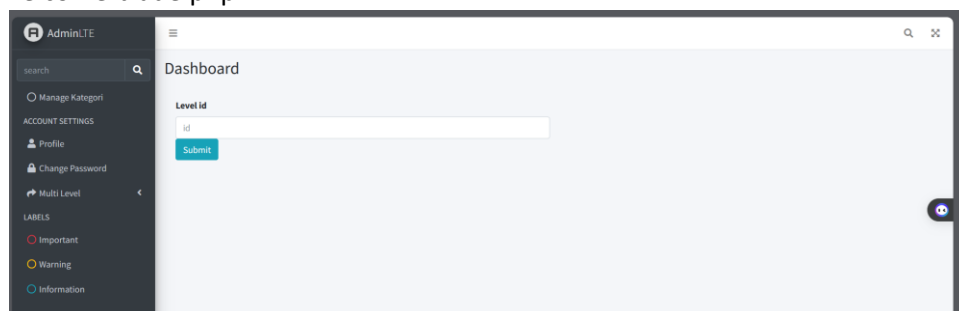
6. Pada bagian menu terdapat pilihan Form kita khususnya dulu pada Form yang disediakan oleh Admin LTE

- General Elements
- Advanced Elements
- Editor
- Validation

Cara menggunakan template bisa dilihat nama formnya contoh judulnya 'General Elements', agar dapat digunakan dapat melakukan akses pada folder yang terdapat template adminLTE contoh : C:\laragon\www\PWL_2024\public\template\pages\forms



7. Buka file pada form> general cari keyword dengan 'general elements' dan coba modifikasi pada welcome.blade.php



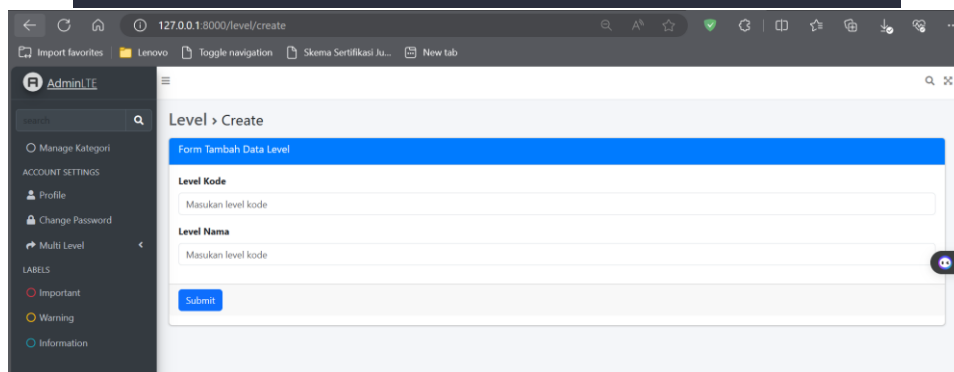
8. 9. Eksplorasi jenis form pada adminLTE dan coba terapkan yang sesuai untuk studi kasus POS PWL, buatlah form untuk tabel m_user dan m_level

Nama : Muhammad Ridlo Febrio Putra

Kelas : 2H

NIM : 2241720098

```
1 @extends('layout.app')
2 {{-- Customize layout sections --}}
3 @section('subtitle', 'Level')
4 @section('content_header_title', 'Level')
5 @section('content_header_subtitle', 'Create')
6 {{-- Content body: main page content --}}
7 @section('content')
8 <div class="card card-primary">
9   <div class="card-header">
10     <h3 class="card-title">Form Tambah Data Level</h3>
11   </div>
12   <!-- /.card-header -->
13   <!-- form start -->
14   <form>
15     <div class="card-body">
16       <div class="form-group">
17         <label for="level_kode">Level Kode</label>
18         <input type="text" name="level_kode" class="form-control" id="level_kode"
19           placeholder="Masukan level kode">
20       </div>
21       <div class="form-group">
22         <label for="level_nama">Level Nama</label>
23         <input type="text" name="level_nama" class="form-control" id="level_nama"
24           placeholder="Masukan level kode">
25       </div>
26     </div>
27     <!-- /.card-body -->
28     <div class="card-footer">
29       <button type="submit" class="btn btn-primary">Submit</button>
30     </div>
31   </form>
32 </div>
33 @endsection
```

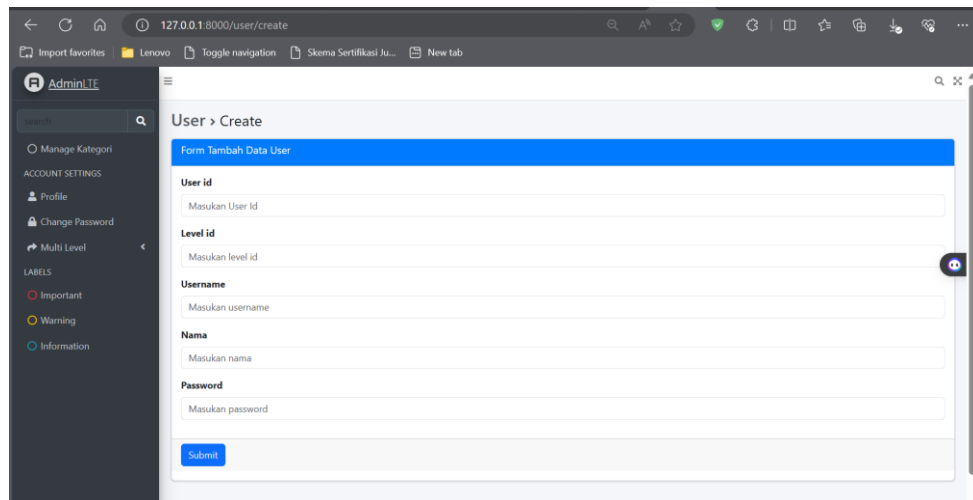


```
1 @extends('layout.app')
2 {{-- Customize layout sections --}}
3 @section('subtitle', 'Level')
4 @section('content_header_title', 'User')
5 @section('content_header_subtitle', 'Create')
6 {{-- Content body: main page content --}}
7 @section('content')
8 <div class="card card-primary">
9   <div class="card-header">
10     <h3 class="card-title">Form Tambah Data User</h3>
11   </div>
12   <!-- /.card-header -->
13   <!-- form start -->
14   <form>
15     <div class="card-body">
16       <div class="form-group">
17         <label for="level_kode">User id</label>
18         <input type="text" name="user_id" class="form-control" id="user_id"
19           placeholder="Masukan User Id">
20       </div>
21       <div class="form-group">
22         <label for="level_nama">Level id</label>
23         <input type="text" name="level_id" class="form-control" id="level_id"
24           placeholder="Masukan level id">
25       </div>
26       <div class="form-group">
27         <label for="level_nama">Username</label>
28         <input type="text" name="username" class="form-control" id="username"
29           placeholder="Masukan username">
30       </div>
31       <div class="form-group">
32         <label for="level_nama">Nama</label>
33         <input type="text" name="nama" class="form-control" id="nama"
34           placeholder="Masukan nama">
35       </div>
36       <div class="form-group">
37         <label for="level_nama">Password</label>
38         <input type="text" name="password" class="form-control" id="password"
39           placeholder="Masukan password">
40       </div>
41     </div>
42     <!-- /.card-body -->
43     <div class="card-footer">
44       <button type="submit" class="btn btn-primary">Submit</button>
45     </div>
46   </form>
47 </div>
48 @endsection
```

Nama : Muhammad Ridlo Febrio Putra

Kelas : 2H

NIM : 2241720098



B. VALIDASI PADA SERVER

1. Buka kembali file tugas di jobsheet 05, klik dan baca documentation ini Validation - Laravel 10.x - The PHP Framework For Web Artisans
2. Pastikan sudah terdefiniskan route pada web.php :

```
Route::post('/kategori', [KategoriController::class, 'store']);
Route::get('/kategori/create', [KategoriController::class, 'create']);
```

3. Edit pada KategoriKontroller dengan kode:

```
6 references | 0 implementations
class KategoriController extends Controller
{
    0 references | 0 overrides
    public function index(KategoriDataTable $dataTable)
    {
        return $dataTable->render('kategori.index');
    }
    1 reference | 0 overrides
    public function create()
    {
        return view('kategori.create');
    }
    1 reference | 0 overrides
    public function store(Request $request) {
        $validated = $request->validate([
            'kategori_kode' => 'required',
            'kategori_nama' => 'required',
        ]);
        return redirect('/kategori');
    }
}
```

4. Tulis perbedaan penggunaan validate dengan validateWithBag!

Jawaban :

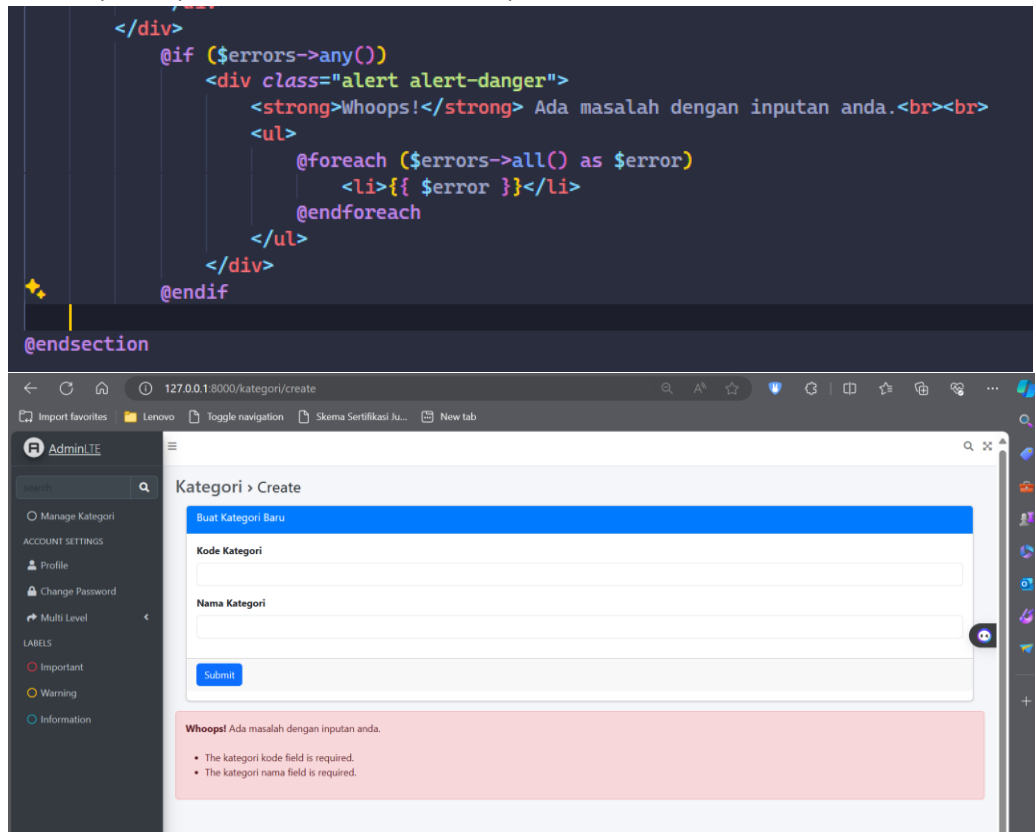
- validate: Digunakan untuk memvalidasi data input terhadap aturan yang ditentukan dalam file validation.php. Jika terdapat kesalahan validasi, maka akan dilemparkan ValidationException yang berisi pesan error.
 - validateWithBag: Mirip dengan validate, tetapi memungkinkan Anda untuk menentukan nama "bag" untuk menyimpan pesan error. Hal ini berguna untuk memisahkan pesan error dari berbagai sumber, seperti formulir yang berbeda pada halaman yang sama.
5. Terkadang Anda mungkin ingin berhenti menjalankan aturan validasi pada suatu atribut setelah kegagalan validasi pertama. Untuk melakukannya, tetapkan bail aturan ke atribut: coba sesuaikan dengan field pada m_kategori, yuk bisa.. apa yang terjadi?

Nama : Muhammad Ridlo Febrio Putra

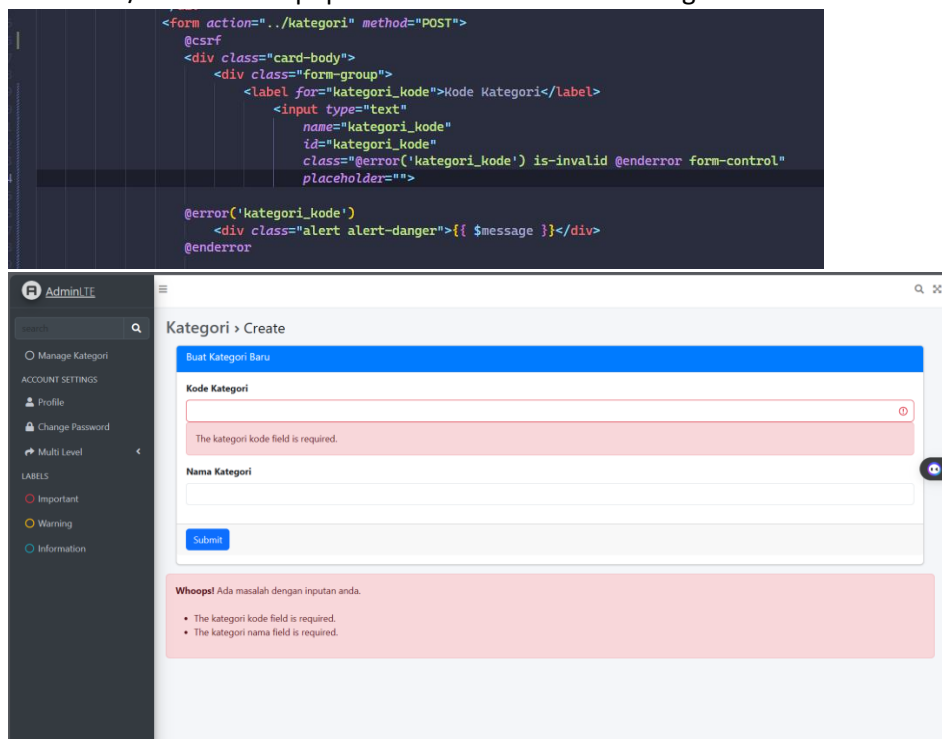
Kelas : 2H

NIM : 2241720098

6. Pada view/create.blade.php tambahkan code berikut agar ketika validasi gagal, kita dapat menampilkan pesan kesalahan dalam tampilan:



7. Pada view/create.blade.php tambahkan dan coba running code berikut :



Nama : Muhammad Ridlo Febrio Putra

Kelas : 2H

NIM : 2241720098

C. FORM REQUEST VALIDATION

1. Membuat permintaan form dengan menuliskan pada terminal

```
PS D:\Laravel KuLiah\PWL_POS> php artisan make:request StorePostRequest

[INFO] Request [D:\Laravel KuLiah\PWL_POS\app\Http\Requests\StorePostRequest.php] created successfully.

PS D:\Laravel KuLiah\PWL_POS>
```

2. Ketik kode berikut pada Http/request/StorePostRequest

```
0 references | 0 overrides
22 public function rules(): array
23 {
24     return [
25         'kategori_kode' => 'required',
26         'kategori_nama' => 'required',
27     ];
28 }
```

```
$validated = $request->validated();
$validated = $request->safe()->only(['kategori_kode', 'kategori_nama']);
$validated = $request->safe()->except(['kategori_kode', 'kategori_nama']);
```

3. Terapkan validasi juga pada tabel m_user dan m_level.

The image displays two screenshots of a web application interface. The top screenshot shows the 'User > Create' form, which includes fields for 'Level id', 'Username', 'Nama', and 'Password'. A red error message at the bottom states: 'Whoops! Ada masalah dengan inputan anda.' followed by a list of errors: 'The username field is required.', 'The nama field is required.', 'The password field is required.', and 'The level id field is required.' The bottom screenshot shows the 'Level > Create' form, which includes fields for 'Level Kode' and 'Level Nama'. A red error message at the bottom states: 'Whoops! Ada masalah dengan inputan anda.' followed by a list of errors: 'The kode level field is required.' and 'The nama level field is required.'

Nama : Muhammad Ridlo Febrio Putra

Kelas : 2H

NIM : 2241720098

D. CRUD(create, read, update, delete)

1. Buat POSController lengkap dengan resourcenya, Membuat Resource Controller dan Route yang berfungsi untuk route CRUD sehingga tidak perlu repot-repot membuat masing-masing route seperti post, get, delete dan update.

```
PS D:\Laravel Kuliah\PWL_POS> php artisan make:controller POSController --resource
```

```
INFO Controller [D:\Laravel Kuliah\PWL_POS\app\Http\Controllers\POSController.php] created successfully.
```

2. Berikut perintahnya: sekarang tambahkan kode pada route/web.php

```
Route::resource('m_user', POSController::class);
```

3. Atur pada Models/m_user

```
php

namespace App\Models;

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

0 references | 0 implementations
class m_user extends Model
{
    0 references
    protected $table = "m_user";
    0 references
    public $timestamps = false;
    0 references
    protected $primaryKey = 'user_id';
    0 references
    protected $fillable = [
        'user_id',
        'level_id',
        'username',
        'nama',
        'password',
    ];
}
```

4. Atur pada Migration/m_user_table

Nama : Muhammad Ridlo Febrio Putra

Kelas : 2H

NIM : 2241720098

```
base / migrations / 2024_05_29_134101_create_m_users_table.php / ...
<?php

use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
5 use Illuminate\Support\Facades\Schema;

return new class extends Migration
{
    /**
     * Run the migrations.
     */
    public function up(): void
    {
        Schema::create('useri', function (Blueprint $table) {
            $table->id('user_id');
            $table->unsignedBigInteger('level_id')->index;
            $table->string('username', 20)->unique();
            $table->string('nama', 100);
            $table->string('password');
            $table->timestamps();
            $table->foreign('level_id')->references('level_id')->on('m_level');
        });
    }

    /**
     * Reverse the migrations.
     */
    public function down(): void
    {
        schema::dropIfExists('useri');
    }
};
```

5. Selanjutnya bukalah app/Http/Controllers/POSController.php kemudian ketikkan, kodenya seperti berikut ini:

```
0 references | 0 overrides
32 public function store(Request $request)
33 {
34     //melakukan validasi data
35     $request->validate([
36         'user_id' => 'max 20',
37         'username' => 'required',
38         'nama' => 'required',
39     ]);
40     m_user::create($request->all());
41     return redirect()->route('m_user.index')
42     ->with('success', 'user Berhasil Ditambahkan');
43
44 }
45
46
47 /**
48  * Display the specified resource.
49  */
0 references | 0 overrides
50 public function show(string $id, m_user $useri)
51 {
52     $useri = m_user::findOrFail($id);
53     return view('m_user.show', compact('useri'));
54 }
55
56 /**
```


Nama : Muhammad Ridlo Febrio Putra

Kelas : 2H

NIM : 2241720098

```
32 public function store(Request $request)
33 {
34     //melakukan validasi data
35     $request->validate([
36         'user_id' => 'max 20',
37         'username' => 'required',
38         'nama' => 'required',
39     ]);
40     m_user::create($request->all());
41     return redirect()->route('m_user.index')
42     ->with('success', 'user Berhasil Ditambahkan');
43
44 }
45
46 /**
47  * Display the specified resource.
48  */
49 0 references | 0 overrides
50 public function show(string $id, m_user $useri)
51 {
52     $useri = m_user::findOrFail($id);
53     return view('m_user.show', compact('useri'));
54 }
55
56 public function edit(string $id)
57 {
58     $useri = m_user::find($id);
59     return view('m_user.edit', compact('useri'));
60 }
61
62 /**
63  * Update the specified resource in storage.
64  */
65 0 references | 0 overrides
66 public function update(Request $request, string $id)
67 {
68     $request->validate([
69         'username' => 'required',
70         'nama' => 'required',
71         'password' => 'required',
72     ]);
73     //fungsi eloquent untuk mengupdate data inputan kita
74     m_user::find($id)->update($request->all());
75     //jika data berhasil diupdate, akan kembali ke halaman utama
76     return redirect()->route('m_user.index')
77     ->with('success', 'Data Berhasil Diupdate');
78 }
79
80 /**
81  * Remove the specified resource from storage.
82  */
83 0 references | 0 overrides
84 public function destroy(string $id)
85 {
86     $useri = m_user::findOrFail($id)->delete();
87     return \redirect()->route('m_user.index')
88     -> with('success', 'data Berhasil Dihapus');
89 }
90
91 }
92
93 }
```

6. Buatlah folder di Resources/Views/m_user dengan beberapa blade dan isian kode berikut
- template.blade.php

Nama : Muhammad Ridlo Febrio Putra

Kelas : 2H

NIM : 2241720098

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4 <title>CRUD Laravel</title>
5 <link rel="stylesheet"
6 href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.mi
7 n.css">
8 </head>
9 <body>
10 <div class="container">
11 @yield('content')
12 </div>
13 </body>
14 </html>
```

b. index.blade.php

```
1 @extends('m_user/template')
2 @section('content')
3 <div class="row mt-5 mb-5">
4 <div class="col-lg-12 margin-tb">
5 <div class="float-left">
6 <h2>CRUD user</h2>
7 </div>
8 <div class="float-right">
9 <a class="btn btn-success" href="{{ route('m_user.create') }}"> Input
10 User</a>
11 </div>
12 </div>
13 </div>
14 @if ($message = Session::get('success'))
15 <div class="alert alert-success">
16 <p>{{ $message }}</p>
17 </div>
18 @endif
19 <table class="table table-bordered">
20 <tr>
21 <th width="20px" class="text-center">User id</th>
22 <th width="150px" class="text-center">Level id</th>
23 <th width="200px" class="text-center">username</th>
24 <th width="200px" class="text-center">nama</th>
25 <th width="150px" class="text-center">password</th>
26 </tr>
27 @foreach ($useri as $m_user)
28 <tr>
29 <td>{{ $m_user->user_id }}</td>
30 <td>{{ $m_user->level_id }}</td>
31 <td>{{ $m_user->username }}</td>
32 <td>{{ $m_user->nama }}</td>
33 <td>{{ $m_user->password }}</td>
34 <td class="text-center">
35 <form action="{{ route('m_user.destroy', $m_user->user_id) }}"
36 method="POST">
37 <a class="btn btn-info btn-sm" href="{{ route('m_user.show', $m_user -> user_id) }}">Show</a>
38 <a class="btn btn-primary btn-sm" href="{{
39 route('m_user.edit', $m_user->user_id) }}">Edit</a>
40 @csrf
41 @method('DELETE')
42 <button type="submit" class="btn btn-danger btn-sm" onclick="return
43 confirm('Apakah Anda yakin ingin menghapus data
44 ini?')">Delete</button>
45 </form>
46 </td>
47 </tr>
48 @endforeach
49 </table>
50 @endsection
```

c. create.blade.php

Nama : Muhammad Ridlo Febrio Putra

Kelas : 2H

NIM : 2241720098

```
15 <div class="alert alert-danger">
16 <ul>
17 </ul>
18 </div>
19 @endif
20 <form action="{{ route('m_user.store') }}" method="POST">
21 @csrf
22 <div class="col-xs-12 col-sm-12 col-md-12">
23 <div class="form-group">
24 <strong>Username:</strong>
25 <input type="text" name="username" class="form-control"
26 placeholder="Masukkan username"></input>
27 </div>
28 </div>
29 <div class="col-xs-12 col-sm-12 col-md-12">
30 <div class="form-group">
31 <strong>nama:</strong>
32 <input type="text" name="nama" class="form-control"
33 placeholder="Masukkan nama"></input>
34 </div>
35 </div>
36 <div class="col-xs-12 col-sm-12 col-md-12">
37 <div class="form-group">
38 <strong>Password:</strong>
39 <input type="password" name="password" class="form-control"
40 placeholder="Masukkan password"></input>
41 </div>
42 </div>
43 <div class="col-xs-12 col-sm-12 col-md-12 text-center">
44 <button type="submit" class="btn btn-primary">Submit</button>
45 </div>
46 </div>
47 </form>
48 @endsection
```

d. show.blade.php

```
@extends('m_user/template')
@section('content')
<div class="row mt-5 mb-5">
<div class="col-lg-12 margin-tb">
<div class="float-left">
<h2> Show User</h2>
</div>
<div class="float-right">
<a class="btn btn-secondary" href="{{ route('m_user.index') }}">
Kembali</a>
</div>
</div>
</div>
<div class="row">
<div class="col-xs-12 col-sm-12 col-md-12">
<div class="form-group">
<strong>User_id:</strong>
{{ $useri->user_id }}
</div>
</div>
<div class="col-xs-12 col-sm-12 col-md-12">
<div class="form-group">
<strong>Level_id:</strong>
{{ $useri->level_id }}
</div>
</div>
<div class="col-xs-12 col-sm-12 col-md-12">
<div class="form-group">
<strong>Username:</strong>
{{ $useri->username }}
</div>
</div>
<div class="col-xs-12 col-sm-12 col-md-12">
<div class="form-group">
<strong>Nama:</strong>
{{ $useri->nama }}
</div>
</div>
<div class="col-xs-12 col-sm-12 col-md-12">
<div class="form-group">
<strong>Password:</strong>
{{ $useri->password }}
</div>
</div>
</div>
</div>
```

Nama : Muhammad Ridlo Febrio Putra

Kelas : 2H

NIM : 2241720098

e. edit.blade.php

```
@extends('m_user/template')
@section('content')
<div class="row mt-5 mb-5">
<div class="col-lg-12 margin-tb">
<div class="float-left">
<h2>Edit User</h2>
</div>
<div class="float-right">
<a class="btn btn-secondary" href="{{ route('m_user.index') }}">
Kembali</a>
</div>
</div>
</div>
</div>
@if ($errors->any())
<div class="alert alert-danger">
<strong>Ops!</strong> Error <br><br>
<ul>
@foreach ($errors->all() as $error)
<li>{{ $error }}</li>
@endforeach
</ul>
</div>
@endif
<form action="{{ route('m_user.update', $user->user_id) }}"
method="POST">
@csrf
@method('PUT')
<div class="row">
<div class="col-xs-12 col-sm-12 col-md-12">
<div class="form-group">
<strong>User_id:</strong>
<input type="text" name="userid" value="{{ $user->user_id }}"
class="form-control" placeholder="Masukkan user id">
</div>
</div>
<div class="col-xs-12 col-sm-12 col-md-12">
<div class="form-group">
<strong>Level_id:</strong>
<input type="text" name="levelid" value="{{ $user->level_id }}"
class="form-control" placeholder="Masukkan level">
</div>
</div>
<div class="col-xs-12 col-sm-12 col-md-12">
<div class="form-group">
<strong>Username:</strong>
<input type="text" value="{{ $user->username }}" class="form-control" name="username" placeholder="Masukkan Nomor username">
</div>
</div>
<div class="col-xs-12 col-sm-12 col-md-12">
<div class="form-group">
<strong>nama:</strong>
<input type="text" value="{{ $user->nama }}" name="nama"
class="form-control" placeholder="Masukkan nama">
</div>
</div>
<div class="col-xs-12 col-sm-12 col-md-12">
<div class="form-group">
<strong>Password:</strong>
<input type="password" value="{{ $user->password }}"
name="password" class="form-control" placeholder="Masukkan
password">
</div>
</div>
<div class="col-xs-12 col-sm-12 col-md-12 text-center">
<button type="submit" class="btn btn-primary">Update</button>
</div>
</div>
</form>
@endsection
```

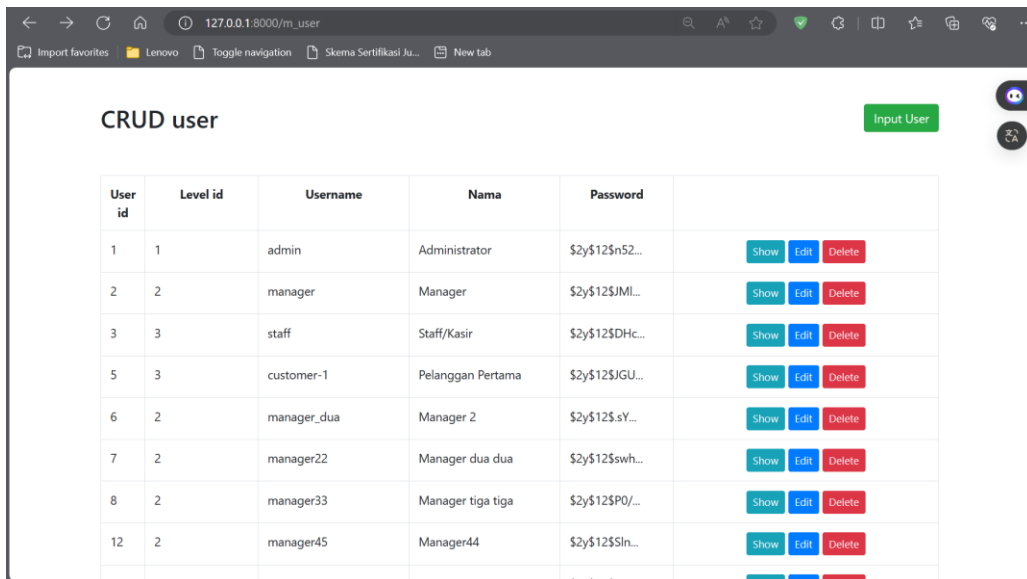
Tugas:

1. Coba tampilkan level_id pada halaman web tersebut dimana field ini merupakan foreign key

Nama : Muhammad Ridlo Febrio Putra

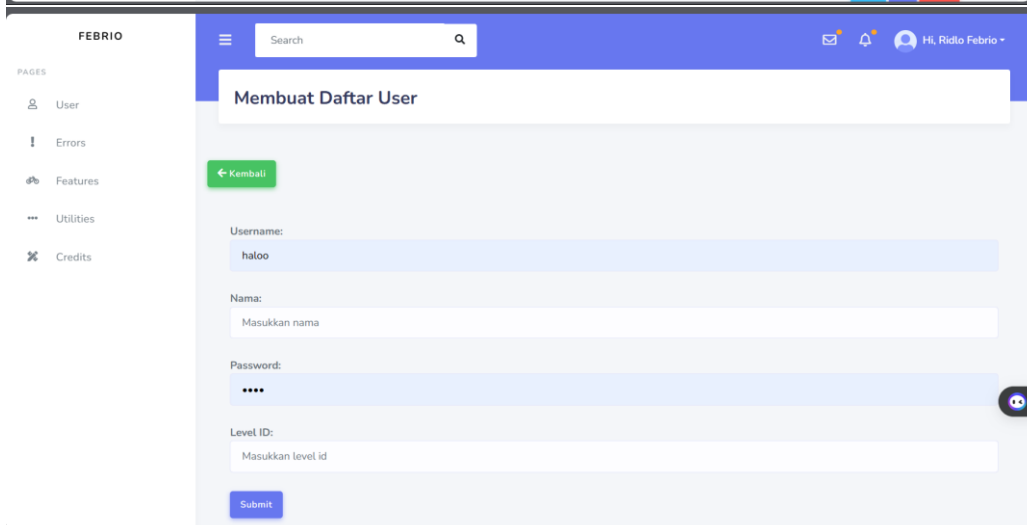
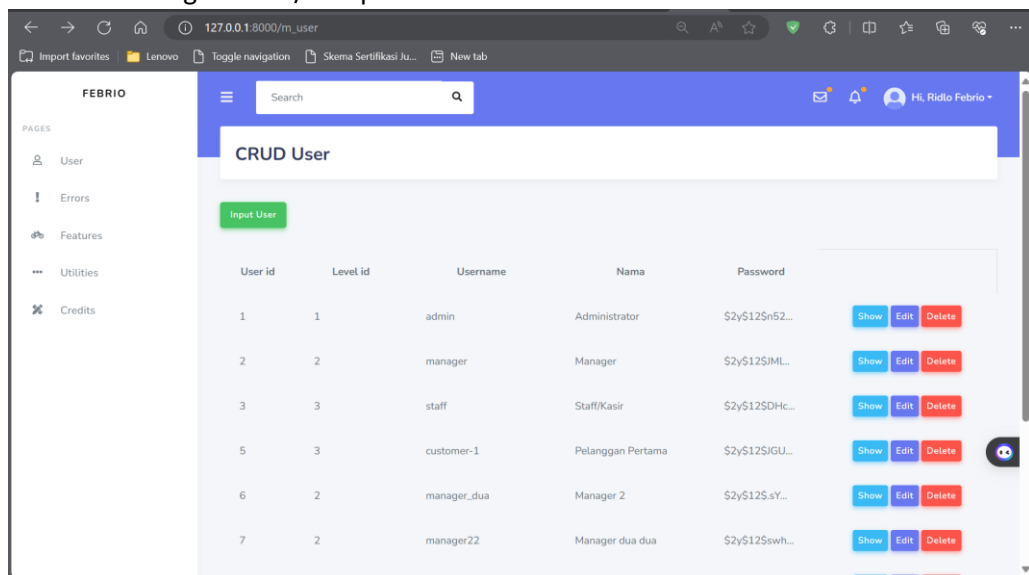
Kelas : 2H

NIM : 2241720098



User id	Level id	Username	Nama	Password	
1	1	admin	Administrator	\$2y\$12\$52...	Show Edit Delete
2	2	manager	Manager	\$2y\$12\$JML...	Show Edit Delete
3	3	staff	Staff/Kasir	\$2y\$12\$DHC...	Show Edit Delete
5	3	customer-1	Pelanggan Pertama	\$2y\$12\$JGU...	Show Edit Delete
6	2	manager_dua	Manager 2	\$2y\$12\$.sY...	Show Edit Delete
7	2	manager22	Manager dua dua	\$2y\$12\$swH...	Show Edit Delete
8	2	manager33	Manager tiga tiga	\$2y\$12\$P0/...	Show Edit Delete
12	2	manager45	Manager44	\$2y\$12\$Sln...	Show Edit Delete

2. Modifikasi dengan tema/ template kesukaan Anda



FEBRIO

Search

PAGES

- User
- Errors
- Features
- Utilities
- Credits

Membuat Daftar User

Kembali

Username:

haloo

Nama:

Masukkan nama

Password:

....

Level ID:

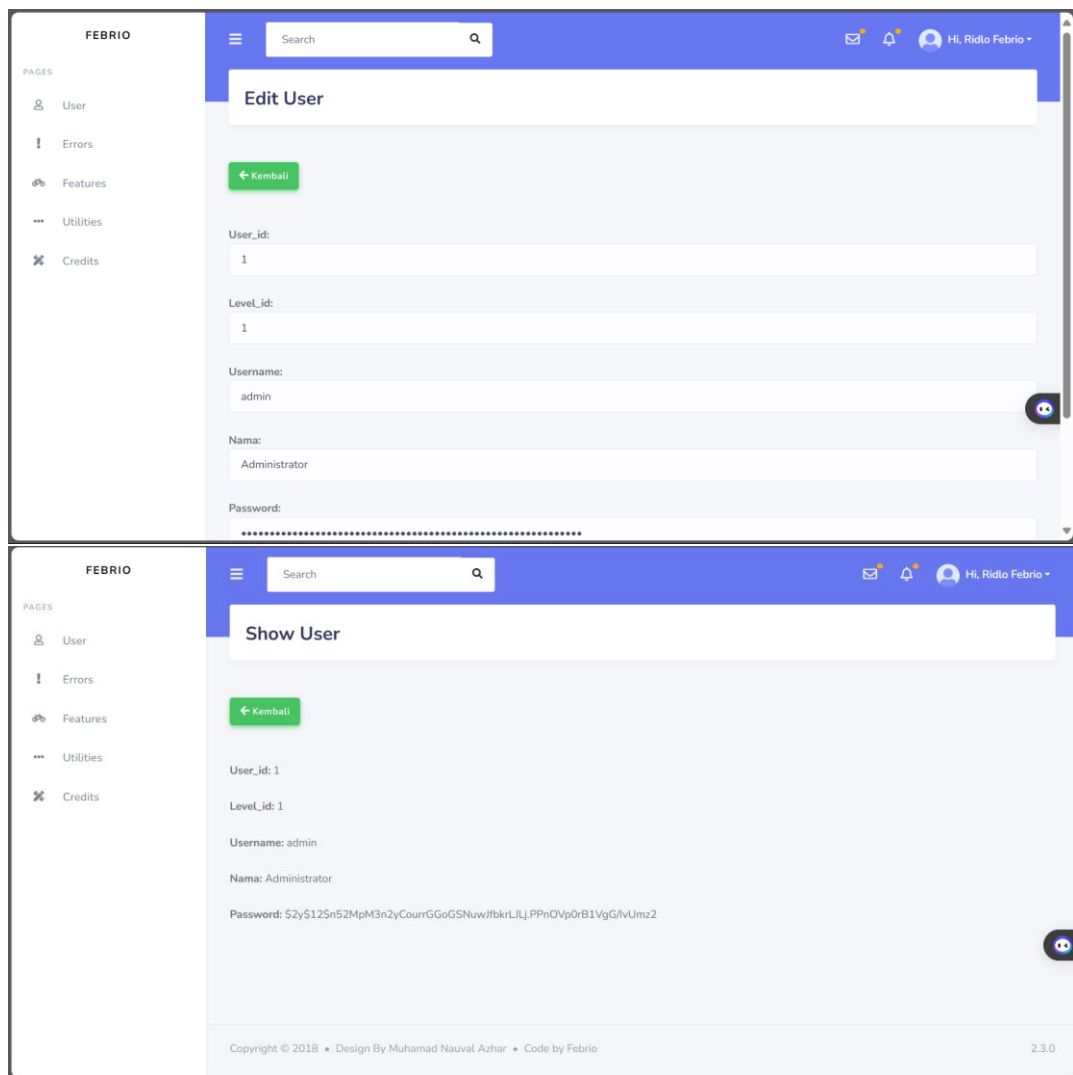
Masukkan level id

Submit

Nama : Muhammad Ridlo Febrio Putra

Kelas : 2H

NIM : 2241720098



3. Apa fungsi \$request->validate, \$error dan alert yang ada pada halaman CRUD tersebut

➤ \$request->validate

Fungsi \$request->validate digunakan untuk memvalidasi data yang dikirimkan dari formulir. Fungsi ini menerima array aturan validasi sebagai parameter. Aturan validasi ini menentukan format dan nilai yang diizinkan untuk setiap field input. Jika terdapat data yang tidak lolos validasi, fungsi \$request->validate akan menghasilkan exception dan menampilkan pesan error kepada pengguna.

➤ \$errors

Variabel \$errors berisi objek yang menyimpan semua pesan error yang dihasilkan dari proses validasi. Objek ini dapat diakses dalam blade view untuk menampilkan pesan error kepada pengguna.

➤ Alert

Fungsi alert digunakan untuk menampilkan pesan kepada pengguna. Fungsi ini menerima dua parameter: jenis pesan dan teks pesan. Jenis pesan dapat berupa success, info, warning, atau danger.