Note :

1. Mungkin anda dapat menghapus migration table user dan password reset (database/Migration/…)

2. Jika iya, hapus Model user (app/User.php)

# Create Migration

* php artisan make:migration create\_xs\_table

Note :

1. setting migration tsb (kolom)

2. bisa plural/tidak (dengan s), jika tidak, tambahkan sintak di class Model :

* protected $table = 'crud';

# Create Migration tsb to Table di database

* php artisan migrate

# Create Controller

* php artisan make:controller XController –resource

Note :

1. Otomatis terdapat method crud

2. Tambahkan sintaks untuk router

* route::resource('xs', 'XController');

3. Jika tidak ingin semua method dapat di routing :

* Route::resource('xs', 'XController')->only([

'index', 'show'

]);

# Create Model

* php artisan make:model X

Note :

1. Jika nama tabel plural (tanpa s : X) dapat ditambahkan sintak pada model :

* $protected $table = 'x' (X adalah nama tabel)

# Read Data

* $xs = X::all();

return view('x.index',['xs' => $xs]);

Note :

1. Masukan sintaks tsb di method index() pada controller

2. Tambahkan sintak di Controller : use App\X; (X adalah nama model)

3. (X::all(), X adalah nama model)

4. (x.index, x adalah nama view)

# Menampilkan Data

* @foreach ($xs as $x)

<p> {{ $x->nama\_kolom }} </p>

<p> {{ $x->nama\_kolom2 }} </p>

@endforeach;

Note :

1. xs adalah variabel yg dikirim dari controller

# Setting Routing

* Route::group(['middleware' => ['web']], function(){

Route::resource('crud', 'CrudController');

});

Note :

1. untuk menampilkan eror di setiap view jika ada eror

# Beri inputan \_token di View Create

* <input type="hidden" name="\_token" value="{{ csrf\_token() }}">

Note :

1. untuk masalah security.

# Validasi input

* $validatedData = $request->validate([

'title' => 'required',

'subject' => 'required|min:1',

]);

Note :

1. <https://laravel.com/docs/5.6/validation#available-validation-rules>

# Menampilkan Error dari Validasi

* @if ($errors->any())

<div class="alert alert-danger">

<ul>

@foreach ($errors->all() as $error)

<li>{{ $error }}</li>

@endforeach

</ul>

</div>

@endif

atau untuk spesifik

* {{ ($errors->has('title')) ? $errors->first('title') : '' }}

# Insert data di store

* $crud = new Crud;

$crud->fill([

'title' => $request->title,

'subject' => $request->subject,

]);

$crud->save();

Note :

1. $request->title : title berasal dari view

2. Crud = nama model

3. return redirect('/crud')->with('message','Berhasil!'); :

Mengirim pesan ke view dengan variabel message

# Tampil Pesan di View setelah CRUD

* return redirect('/crud')->with('message','Berhasil tambah data!');
* {{ Session::get('message') }}

Note :

1. sintak 1 diletakan pada suatu method di controller

2. sintak 2 diletakan pada view

# Read Single Page di show()

* $crud = Crud::find($id);

if(!$crud){

abort(404);

}

return view('crud.single')->with('crud',$crud);

* <a href="/laravel\_crud/public/crud/{{$crud->id}}"> <p> {{ $crud->title }} </p> </a>

Note :

1. find(id) -> mencari id

2. buat sendiri halaman 404

3. sintak 2 untuk membuat link menampilkan data khusus

# Update Data

Note :

1. copas method store()

# Read Data Edit

Note :

1. copas method show()

2. view edit copas dari method create()

# Ganti method post dengan put/patch

* <input type="hidden" name="\_method" value="put">
* <a href="/laravel\_crud/public/crud/{{$crud->id}}/edit">Edit</a>

Note :

1. sintak 1 tambahkan sintak tersebut di view edit. Bukan mengganti!

2. sintak 2 untuk membuat link ke view edit

# Delete data

* $crud = Crud::find($id);

$crud->delete();

return redirect('/crud')->with('message','Berhasil di delete!');

* <form class="" action="/laravel\_crud/public/crud/{{$crud->id}}" method="post">

<input type="hidden" name="\_method" value="delete">

<input type="hidden" name="\_token" value="{{ csrf\_token() }}">

<input type="submit" name="name" value="delete">

</form>

Note :

1. sintak 1 letakan di methos destroy()

2. sintak 2 letakan di halaman view

# Menampilkan Data dengan Slug pada url

* $crud = Crud::where('slug',$title)->first();
* <a href="/laravel\_crud/public/crud/{{$crud->slug}}"> <p> {{ $crud->title }} </p> </a>

Note :

1. edit $crud = Crud::find($id); dengan sintak diatas di method show()

2. jangan lupa untuk edit parameter menjadi $title

2. kolom slug pada table postnya

2. edit migration pada table bersangkutan dengan : $table->string('slug');

3. edit store() dan update() dengan : 'slug' => str\_slug($request->title,'-')

4. edit halaman index pada

<a href="/laravel\_crud/public/crud/{{$crud->title}}"> <p> {{ $crud->title }} </p> </a>

Dengan sintak 2

# Menampilkan waktu dibuat dan update

* <p> Tanggal di buat : {{ date('F d,Y', strtotime($crud->created\_at)) }} </p>
* <p> Tanggal di update : {{ date('F d,Y', strtotime($crud->updated\_at)) }} </p>

Note :

1. keterangan format bisa di lihat di <http://php.net/manual/en/function.date.php>

# Pagination

* $cruds = Crud::paginate(2);
* $cruds = DB::table('crud')->paginate(1);
* {!! $cruds->links() !!}

Note :

1. tambahkan sintak use DB; di controller

2. sintak 1 jika memakai eloquent

3. sintak 2 jika memakai query builder

4. tambahkan sintak 3 pada view indexnya