

# TUTORIAL LARAVEL 4

Materi : [seputarpemograman.com](http://seputarpemograman.com)Perangkum : [drafshare.blogspot.com](http://drafshare.blogspot.com)

Email : [seputarpemrograman@gmail.com](mailto:seputarpemrograman@gmail.com) | [dirhamsistem@gmail.com](mailto:dirhamsistem@gmail.com)

**{ GRATIIIISSSSSSSSSSSSSSSSSSSSSSSSSSSS }**

# DAFTAR ISI

BAB I .....	5
1. Asyiknya, Belajar Dasar Routing Framework Laravel.....	5
<b>BELAJAR BASIC ROUTE</b> .....	5
<b>BELAJAR ROUTE PARAMETERS</b> .....	6
BAB II .....	10
2. Memahami Dasar Controller dan view di Framework Laravel .....	10
<b>Basic Controllers ( Dasar Controller )</b> .....	11
<b>Controller Dengan Route Parameters</b> .....	12
<b>Kolaborasi antara view dan Controller</b> .....	13
BAB III .....	15
3. Memanfaatkan Blade Templating Laravel Untuk Memadukan Template Bootstrap .....	15
<b>Dasar - Dasar Blade</b> .....	15
<b>Memadukan Blade Dengan Template Bootstrap</b> .....	17
BAB IV .....	24
4. Belajar Dasar Form Framework Laravel Dan Latihan Membuat Form Sederhana .....	24
<b>Dasar – dasar Form Laravel</b> .....	24
<b>Membuat Form Sederhana</b> .....	26
BAB V .....	30
5. Cara Menggunakan Migrations dan Schema Builder di Framework Laravel .....	30
<b>Dasar atau Perintah Migrations (migrasi)</b> .....	30
<b>Schema Builder (Skema Builder)</b> .....	31
<b>Menerapkan Migration (Migrasi) dan Schema Builder (Skema Builder)</b> .....	33
BAB VI .....	37
6. Cara Membangun Create, Read, Update dan Delete Menggunakan Query Builder di Laravel.....	37
BAB VII.....	46
7. Membuat Create, Read, Update dan Delete Menggunakan Eloquent ORM dan RESTful Resource Controllers Laravel .....	46
BAB VIII .....	57
8. Memanfaatkan Auth Bawaan Laravel Untuk Membuat Authentication User Sederhana .....	57
Fungsi Dasar Auth .....	57

Belajar Membuat Auth Sederhana .....	59
BAB IX.....	67
9. Cara Menggunakan package sentry di framework laravel bagian install dan register .....	67
Install Cartalyst Sentry .....	67
Membuat Form Register / Pendaftaran dengan Cartalyst Sentry. ....	70
BAB X.....	74
10. Cara Menggunakan package sentry di framework laravel bagian CRUD Group .....	74
BAB XI.....	83
11. Cara Menggunakan package sentry di framework laravel bagian CRUD User .....	83
BAB XII.....	93
12. Cara Menggunakan Package Sentry Di Framework Laravel Bagian Authentication Dan Login..	93
BAB XIII.....	99
13. Cara Menggunakan Package Sentry Di Framework Laravel Bagian Reset Password atau Forgot Password User.....	99

**Catatan :**

- Isi pada ebook ini hanya untuk pembelajaran dasar dan pengenalan Framework Laravel, adapun jika anda ingin menegetahui lebih lanjut tentang laravel sangat di sarankan untuk mengunjungi dokumentasi resmi laravel.
- Kami tidak menyertakan langkah instalasi karena langkah ini dengan mudah dapat di temukan dengan browsing #dilarang malas :p
- Ebook banyak kekurangan dan penyusunan tidak rapi ? Kan Gratissss :D

**Lisensi :**

- Isi Ebook boleh di sunting dan di bagikan dengan syarat menyertakan sumber serta tidak untuk tujuan komersil.
- Sebelum mengubah silahkan kontak kami melalui email yang tertera di awal ebook

**#MORE...**

Tutorial dan Ebook lainnya silahkan berkunjung ke :

- [Seputarpemograman.com](http://Seputarpemograman.com)
- [Drafshare.blogspot.com](http://Drafshare.blogspot.com)

## Kata Pengantar

Laravel Framework Laravel Dibuat oleh Taylor Otwell, proyek Laravel dimulai pada April 2011. Saat ini pengembangnya bekerja pada UserScape. Awal mula proyek ini dibuat karena Otwell sendiri tidak menemukan framework yang up-to-date dengan versi PHP.

Mengembangkan framewrok yang sudah ada juga bukan merupakan ide yang bagus karena keterbatasan sumber daya. Dikarenakan beberapa keterbatasan tersebut, Otwell membuat sendiri framework dengan nama Laravel.

Laravel adalah framework aplikasi web yang ekspresif, sintaks yang elegan. Kami percaya pembangunan harus menjadi menyenangkan, pengalaman kreatif yang akan benar-benar memuaskan Laravel berupaya untuk mengurangi penderitaan pembangunan web dengan mengurangi tugas-tugas umum digunakan dalam sebagian besar proyek-proyek web, seperti otentikasi, routing, sesi, dan caching.

Laravel bertujuan untuk membuat proses pembangunan web yang menyenangkan bagi pengembang tanpa mengurangi fungsionalitas aplikasi. Untuk tujuan ini, laravel telah mencoba untuk menggabungkan/mengadopsi fitur-fitur terbaik dari apa yang telah kita lihat dalam framework lain, yakni mengimplementasikan framework dalam bahasa lain, seperti Ruby on Rails, ASP.NET MVC, dan Sinatra.

# BAB I

## 1. Asyiknya, Belajar Dasar Routing Framework Laravel

Pada tutorial laravel tahap ketiga ini kita akan membahas routing di framework laravel. Dari judul artikel atau tutorial ini " Asyiknya, Belajar Dasar Routing Framework Laravel", mengapa saya menyebutkan asyik belajar routing. Dengan routing secara sederhana kita dapat menuliskan atau mensetting nama url website yang akan kita bangun. contohnya saja **www.seputarpemrograman.com/{uri maupun parameter yang kita inginkan}**.

Dalam routing kita akan mengenal yaitu **GET** dan **POST**, pastinya sudah tidak asing lagi bagi yang terbiasa belajar atau membuat website. Secara gamblang kita bisa melihat perbedaannya, GET data dan parameter akan terlihat di url sedangkan POST tidak terlihat di url. Tapi framework laravel bagian routing GET dan POST ini mempunyai cara kerja yang berbeda dan yang dimaksud bukan **\$\_GET** maupun **\$\_POST** kalau di laravel **\$\_GET** dan **\$\_POST** ini bisa kita pelajari di bagian basic input di laravel.com contohnya **Input::get('name');**. Tapi di tutorial ini atau di dalam routing laravel GET disini digunakan untuk menampilkan, parsing atau mengirimkan data. Sedangkan POST digunakan untuk menangani sebuah form untuk menerima hasil inputan data.

Di laravel maupun di dunia programming kita pasti mengenal istilah RESTful. **RESTful** mempunyai beberapa method selain **\$\_GET** dan **\$\_POST** yaitu PUT/PATCH dan DELETE. Apa itu RESTful, RESTful merupakan sebuah teknik di arsitektur software untuk sistem terdistribusi seperti World Wide Web. Contoh nyata penggunaan Restful itu sendiri seperti kita berkunjung di sebuah web contohnya saja 4shared lalu kita bisa login dengan akun facebook. Tapi di tutorial ini kita tidak akan membahas lebih dalam mengenai RESTful.

Di tutorial ini saya tidak akan membahas mengenai method POST, karena method ini akan saya bahas di tutorial form laravel. Sedangkan RESTful akan saya bahas di tutorial membuat crud. Pada Tahap ini kita hanya akan membahas penggunaan GET.

*Catatan : routing dapat didefinisikan atau dituliskan di folder **app** lalu file **routes.php**.*

**GET**, method ini dalam laravel biasanya digunakan untuk menampilkan view maupun data. Buka Project yang telah di install, Jika belum install laravel silahkan ikuti tutorial install "Belajar Install Framework Laravel Dengan Composer Maupun Tanpa Composer". Jika sudah silahkan dibuka projectnya cari file routes.php berada dalam folder app.

## BELAJAR BASIC ROUTE

Secara dasar penggunaan dasar route sebagai berikut, silahkan buka file routes.php anda akan menemukan code seperti dibawah ini.

```
1 Route::get('/', function()  
2 {  
3     return View::make('hello');  
4 });
```

Code diatas silahkan dihapus dan diganti menjadi seperti dibawah ini.

```
1 Route::get('/',function()  
2 {  
3     return'Ini Contoh Halaman Home';  
4 });
```

**Route::get** adalah method digunakan untuk route dengan method get seperti yang dijelaskan diatas.

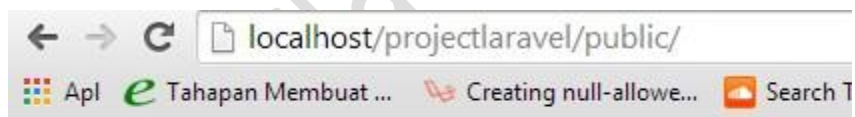
**Tanda '/'**, menunjukan route yang di definisikan atau dibuat untuk route utama atau halaman utama contoh seperti seputarpmrograman.com bukan seputarpmrograman.com/tutorial-laravel.

**return'Ini Contoh Halaman Home';** menunjukan mengembalikan suatu nilai yang berisi “Ini Contoh Halaman Home”.

Untuk cara menjalankan hasil route diatas silahkan ketik url :

```
1 <a  
  href="projectlaravel/public/">http://localhost/projectlaravel/public/</a>
```

Hasilnya menampilkan



Ini Contoh Halaman Home

## BELAJAR ROUTE PARAMETERS

Fungsi route parameters ini adalah mengirimkan sebuah nilai atau parameter ke route atau ke controller. Di route parameters kita dibagi menjadi beberapa bagian :

1. Basic Route Parameters (Dasar Route Berparameter)
2. Route Parameters (Route Berparameter Lebih dari Satu)
3. Optional Route Parameters (Opsional Route Parameter)
4. Optional Route Parameters With Defaults (Opsional Route Parameter dengan Nilai Default)

## 1. Basic Route Parameters (Dasar Route Berparameter)

Tambahkan code berikut ke file routes.php

```
1 Route::get('profile/{nama}',function($nama)
2 {
3     return'Profile Nama : '.$nama;
4 });
```

**profile/{nama}** , digunakan untuk definisi / penamaan route / urlnya sedangkan untuk {nama} adalah variable yang dikirimkan atau parameternya.

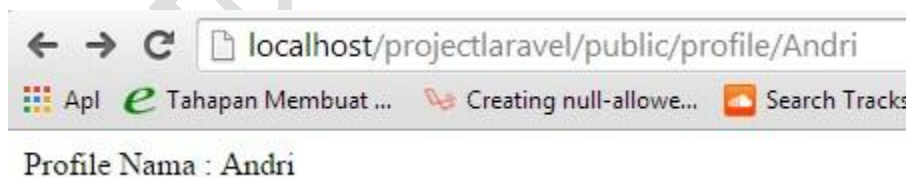
**function(\$nama)** , digunakan untuk menangkap parameter

**return'Profile Nama : '.\$nama;** , digunakan untuk Menampilkan kalimat Profile Nama : Sesuai Parameter.

Contoh cara menjalankan ketikan url seperti ini :

```
1 http://localhost/projectlaravel/public/profile/Andri
```

Maka yang dihasilkan akan seperti ini :



## 2. Route Parameters (Route Berparameter Lebih dari Satu)

Untuk mengirim parameter lebih dari satu dari contoh dasar route berparameter kita hanya perlu menambahkan /{parameter-baru}. Lebih jelasnya lihat contoh dibawah ini dan bandingkan dengan contoh sebelumnya.

Tambahkan code berikut ini ke routes.php

```
1 Route::get('profile/{id}/{nama}',function($id, $nama)
2 {
3     return'Profile ID / Nama : '.$id.' / '.$nama;
4 });
```

**profile/{id}/{nama}** , digunakan untuk definisi / penamaan route / urlnya sedangkan untuk {id} adalah variable yang dikirimkan atau parameternya. {nama} adalah parameter kedua.

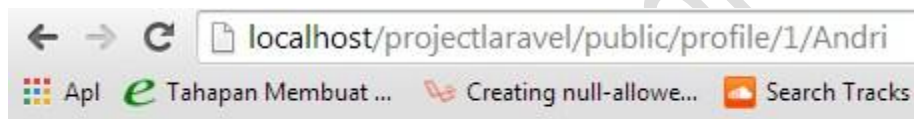
**function(\$id, \$nama)** , digunakan untuk menangkap parameter

**return'Profile ID / Nama : '.\$id.' / '.\$nama;** , digunakan untuk Menampilkan kalimat Profile Nama : Sesuai Parameter Pertama / Sesuai Parameter Kedua .

Coba jalankan dengan perintah / url seperti ini :

```
1 http://localhost/projectlaravel/public/profile/1/Andri
```

Maka hasilnya menjadi seperti ini



### 3. Optional Route Parameters (Opsional Route Parameter)

Opsional Route Parameter adalah sebuah pilihan jika parameter tidak dicantumkan maka akan diganti dengan null dengan demikian program akan tetap jalan. Contohnya saja silahkan di contoh sebelumnya anda hilangkan parameter apa yang terjadi, program akan error. Sekarang mari kita lihat contoh dari opsional route parameter ini.

```
1 Route::get('profile/{nama?}', function($nama = null)
2 {
3     return $nama;
4 });
```

**profile/{nama?}** , digunakan untuk define / penamaan route / urlnya, {nama?} artinya jika parameter tidak ditemukan akan diganti null dengan code berikut \$nama = null.

**return \$nama;** , mengembalikan nilai parameternya.

Coba jalankan dengan perintah / url seperti ini :



---

```
1 http://localhost/projectlaravel/public/profile
```

---

maka hasilnya akan blank page / kosong, jika dijalankan seperti ini

---

```
1 http://localhost/projectlaravel/public/profile/Andri
```

---

Maka menghasilkan “Profile Nama : Andri”.

#### 4. Optional Route Parameters With Defaults (Opsional Route Parameter dengan Nilai Default)

Opsional Route Parameter dengan Nilai Default ini hampir sama dengan Opsional Route Parameter Cuma perbedaannya hanya kita bisa memberikan nilai default pada suatu parameter jika parameternya tidak ditemukan, contohnya :

---

```
1 Route::get('profile/{nama?}', function($nama = 'Andri')
2 {
3     return $nama;
4 });
```

---

**profile/{nama?}** , digunakan untuk define / penamaan route / urlnya, {nama?} artinya jika parameter tidak ditemukan akan diganti null dengan code berikut \$nama = 'Andri'.

Coba jalankan dengan perintah / url seperti ini :

---

```
1 http://localhost/projectlaravel/public/profile
```

---

maka yang ditampilkan “Profile Nama : Andri” karena defaultnya Andri jika kita rubah umpama Rudi maka yang dihasilkan “Profile Nama : Rudi”.

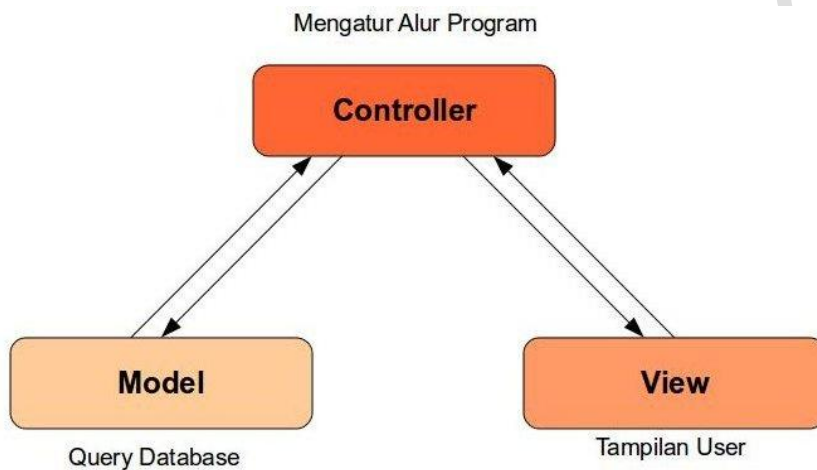
Itulah dasar route yang perlu diketahui, dan masih banyak yang lain tentang route tapi kita akan pelajari dengan berjalan, maksudnya dengan belajar yang lain. Kalau kita pelajari langsung semua bagi pemula ataupun yang belum pernah tau apa itu route maka akan menjadi bingung.

*Catatan : Untuk tes code diatas usaha kan code yang lain didalam routes.php dinonaktifkan dengan memberi komentar pada code tersebut agar tidak bentrok dan lebih tau fungsi yg mana yang sedang berjalan.*

# BAB II

## 2. Memahami Dasar Controller dan view di Framework Laravel

Setelah belajar routing laravel di tutorial [Belajar Dasar Routing Framework Laravel](#). Selanjutnya belajar memahami cara kerja controllers dan views di Framework Laravel. Bagaimana cara kerja antara routing, controller, view dan model. Sebelumnya kita harus tau konsep **MVC**, **M** adalah Model, Model digunakan untuk query atau proses mengambil data dari databas. **V** adalah View, View digunakan untuk tampilan user berupa html, css, js dan data yang bersifat client. Sedangkan **C** adalah Controller yang digunakan untuk mengatur atau menjembatani antara model dan view atau bisa digunakan untuk mengatur alur program. Berikut ini gambar dari MVC.



Di tutorial kali ini kita akan hanya membahas Controller dan View. Dari belajar routing sebelumnya kita akan menggabungkan dengan controller dan view. Jika belum silahkan lihat tutorial Asyiknya, Belajar Dasar Routing Framework Laravel. Untuk persiapan buka file **routes.php** anda dan didalam folder **controllers** buat file dengan nama **ProfileController.php**. Setelah itu ProfileController.php isi dengan code berikut :

---

```
1 < ?php
2 class ProfileController extends BaseController {
3
4 }
5 //Untuk spasi diantara tanda < dan ? silahkan dihapus
6
```

---

## Basic Controllers ( Dasar Controller )

Dasar controller ini adalah bagaimana cara secara dasar kita dapat menjalankan controller tersebut dengan bantuan route. Apakah anda masih ingat code dibawah ini terletak dimana ?

```
1 Route::get('/',function()  
2 {  
3     return'Ini Contoh Halaman Home';  
4 });
```

Yapz.. , code ini terletak di file **routes.php** dan apa hubungannya, seperti yang kita jelaskan sebelumnya dimateri routing , routing digunakan untuk menentukan uri atau format url. Lah ketika kita belajar routing sebelumnya string Ini Contoh Halaman Home berada dirouting itu sendiri, bayangkan jika didalam method / fungsi **Route::get('/',function()** Ini sampai puluhan atau sampai ratusan baris apa kita tidak pusing melihatnya belum lagi ditambah fungsi – fungsi yang lainnya. Maka dari itu controller ini berfungsi memecah code itu sendiri. Bagaimana caranya agar fungsi – fungsi puluhan bahkan ratusan itu kita pindah ke controller. Lihat code sederhana dibawah ini.

```
1 Route::get('/',function()  
2 {  
3     return'Ini Contoh Halaman Home';  
4 });
```

Code diatas silahkan di ganti dengan code dibawah ini.

```
1 Route::get('/', 'ProfileController@index');
```

Dan di **ProfileController.php** sebelumnya yang telah kita buat tambahkan code berikut ini.

```
1 public function index()  
2 {  
3     return'Ini Contoh Halaman Home';  
4 }
```

Hasilnya di ProfileController.php menjadi seperti ini

```
1 class ProfileController extends BaseController {  
2  
3     public function index()  
4     {  
5         return'Ini Contoh Halaman Home';  
6     }  
7 }  
8
```

Sekarang penjelasan dari code didalam routes.php yang telah diganti sebelumnya

```
1 Route::get('/', 'ProfileController@index');
```

**Route::get** adalah method digunakan untuk digunakan untuk menampilkan view maupun data.

**Tanda '/'**, menunjukan route yang di definisikan atau dibuat untuk route utama atau halaman utama contoh seperti seputarpmrograman.com bukan seputarpmrograman.com/tutorial-laravel.

**ProfileController** , adalah nama file ataupun class controller.

**index** , adalah method atau fungsi yang berada di controller.

Untuk mencoba menajalankan program sama seperti sebelumnya ketikan url seperti ini

```
1 http://localhost/projectlaravel/public/
```

Hasilnya pun sama seperti tutorial routing, Itu adalah Cara Dasar menggunakan Controller.

## Controller Dengan Route Parameters

Tutorial diatas adalah cara menggunakan controller secara dasar, sekarang bagaimana jika kita parsing atau mengirim parameter di controller seperti yang dilakukan di routing.

Rubah code dibawah ini yang berada di routes.php

```
1 Route::get('profile/{nama}',function($nama)
2 {
3     return'Profile Nama : '.$nama;
4 });
```

Menjadi

```
1 Route::get('profile/{nama}', 'ProfileController@profile');
```

Dan didalam controller file **ProfileController.php** tambah fungsi berikut ini dibawah fungsi index.

```
1 public function profile($nama)
2 {
3     return'Profile Nama : '.$nama;
4 }
```

Keterangan code :

```
Route::get('profile/{nama}', 'ProfileController@profile');
```

**profile/{nama}**, menunjukan route yang di definisikan. {nama} adalah parameter.

**ProfileController** , adalah nama file ataupun class controller.

**profile** , adalah method atau fungsi yang berada di controller.

**public function profile(\$nama)** . adalah sebuah fungsi bernama profile yang menangkap sebuah parameter dengan variabel nama.

**return 'Profile Nama : '.\$nama;** , mengembalikan nilai dari parameter nama.

Untuk menjalankannya juga masih sama seperti tutorial sebelumnya.

---

```
1 http://localhost/projectlaravel/public/profile/Andri
```

---

Hasilnya pun juga sama.

## Kolaborasi antara view dan Controller

Setelah membahas cara kerja controller bagaimana kita menggabungkan view dengan controller. Sebelumnya di folder **views** buatlah sebuah file bernama **profile.php** isikan code berikut :

---

```
1 Profile ID / Nama : < ?php echo $id; ?> / < ?php echo $nama; ?>
2
3 //Untuk spasi diantara tanda < dan ? silahkan dihapus
```

---

Lalu di controller, **ProfileController.php** tambah fungsi berikut dibawah fungsi profileview.

---

```
1 public function profileview($id,$nama)
2 {
3     $data =
4
5     [
6
7     'id' => $id, // Menyimpan nilai dari variable id ke dalam array
8
9     'nama' => $nama // Menyimpan nilai dari variable nama ke dalam array
10
11     ];
```

---

---

```
12
13     return View::make('profile',$data); // Parsing data ke dalam view lalu
14     ditampilkan.
15
16 }
```

---

Dan yang terakhir ganti code didalam routes.php ini

---

```
1 Route::get('profile/{id}/{nama}',function($id, $nama)
2 {
3     return'Profile ID / Nama : '.$id.' / '.$nama;
4 });
```

---

Menjadi seperti ini

---

```
1 Route::get('profile/{id}/{nama}', 'ProfileController@profileview');
```

---

Keterangan code :

**\$data =**

[

**'id' => \$id, // Menyimpan nilai dari variable id ke dalam array**

**'nama' => \$nama // Menyimpan nilai dari variable nama ke dalam array**

];

Code diatas adalah menyimpan nilai dari variabel id dan nama kedalam variabel data berbentuk array.

**return View::make('profile',\$data);** , digunakan untuk memanggil atau menampilkan file di folder views dan data. **'profile'** disini adalah nama file yang berada didalam folder views. **\$data** adalah data berbentuk array yang dikirim ke dalam view atau profile.php di folder views.

*Catatan : tutorial diatas hanya dasar atau sebagian kecil dari materi controller dan view. Tapi tutorial diatas sudah lebih dari cukup untuk mengenal atau memahami laravel, setelah paham tutorial diatas anda dapat belajar lebih dalam lagi di laravel.com. Tujuan tutorial ini hanya untuk mengenalkan saja tidak semua dibahas disini. Untuk tentang model akan kita bahas di tutorial CRUD laravel.*

# BAB III

## 3. Memanfaatkan Blade Templating Laravel Untuk Memadukan Template Bootstrap

Di Tahap atau tutorial ini kita akan mempelajari blade dan cara memanfaatkannya. Blade adalah template engine yang disediakan oleh framework laravel atau bawaan dari framework laravel. Blade adalah konversi dari bahasa pemrograman PHP ke Blade itu sendiri agar lebih mudah dipahami dan lebih singkat. Kenapa kita harus menggunakan Blade ? Karena dengan menggunakan blade dalam mengatur layout web yang kita bangun akan lebih mudah. Mudah disini jika anda sudah mengetahui dasar – dasar blade itu sendiri.

### Dasar - Dasar Blade

#### 1. Menampilkan Data ( echo )

```
1 Nama Saya : {{{ $nama }}}
```

Jika Menggunakan PHP Identik dengan

```
1 Nama Saya : < ?php echo $nama; ?>
```

Anda juga bisa menggunakan fungsi dibawah ini

```
1 Nama Saya : {{ $nama }}
```

Code diatas bedanya dengan sebelumnya adalah kalau yang diatas ini menggunakan dua kurung kurawal itu tanpa adanya escape html.

#### 2. Menampilkan Data Setelah dilakukan Pengecekan

Fungsi ini bermanfaat ketika kita ingin cek variabel ditemukan atau tidak. Jika ditemukan maka menampilkan variabel itu sendiri jika tidak akan menampilkan variabel defaultnya. Lebih jelasnya lihat dibawah ini.

```
1 {{{ isset($name) ? $name : 'Default' }}}
```

Atau bisa menggunakan code dibawah ini lebih simple.

```
1 {{{ $name or 'Default' }}}
```

### 3. Menampilkan Raw Text dengan Curly Braces

Untuk menampilkan text raw dalam blade kita bisa menggunakan tanda `@{{ ... }}` seperti dibawah ini.

```
1 @{{ Kalimat yang ingin ditampilkan }}
```

### 4. Menggunakan Statemen (If Endif, If Else Endif)

Untuk menggunakan statemen if di blade caranya dibawah ini.

```
1 @if ('A'==='A')
2     @{{ Betul ini adalah A }}
3 @endif
```

Cara menggunakan If Else di blade

```
1 @if ('A'==='A')
2     @{{ Betul ini adalah A }}
3 @else
4     @{{ Ini Bukan A }}
5 @endif
```

Menggunakan If Elseif

```
1 @if ('A'==='A')
2     @{{ Betul ini adalah A }}
3 @elseif ('B'==='B')
4     @{{ Betul ini adalah B }}
5 @else
6     @{{ Ini Bukan A dan B }}
7 @endif
```

### 5. Menggunakan Perulangan (For, Foreach, While)

Fungsi Menggunakan For maupun Foreach di blade

**For**

```
1 @for ($i = 0; $i < 10; $i++)
2     {{ $i }}
3 @endfor
```

**Foreach**



```
1 @foreach ($users as $user)
2     ID user {{ $user->id }}
3 @endforeach
```

## While

```
1 @while (true)
2     I'm looping forever.
3 @endwhile
```

## 6. Include Sub View

```
1 @include('nama view')
```

Nama view tanpa .blade.php

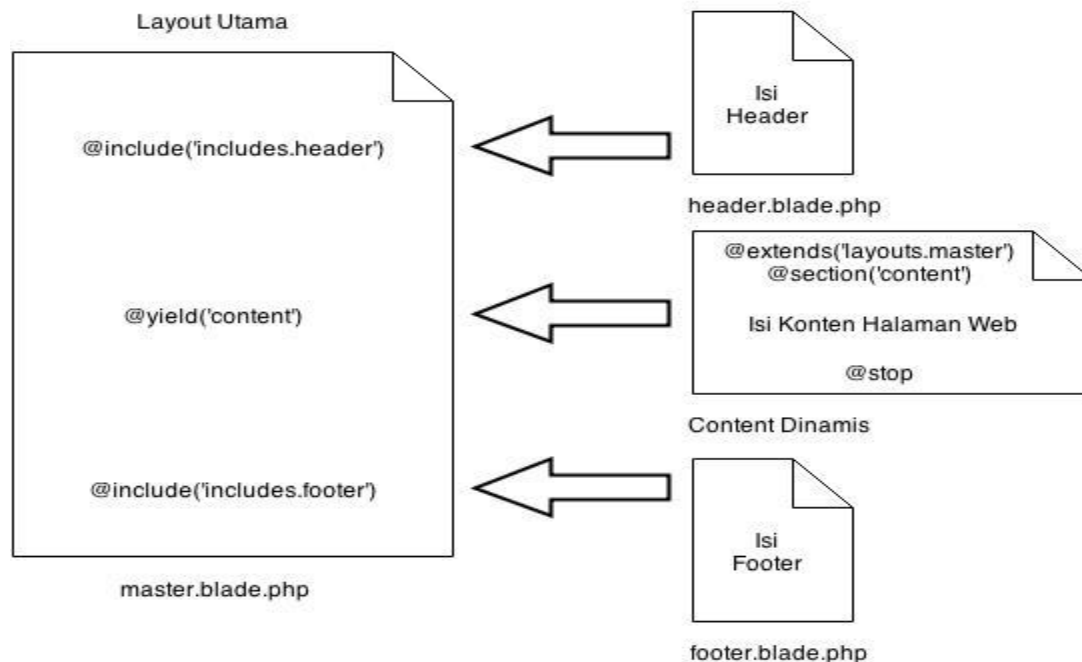
## 7. Memberikan Komentar

Untuk memberikan komentar pada blade caranya berikut ini

```
1 {{-- Dengan begini kalimat ini tidak tampil --}}
```

# Memadukan Blade Dengan Template Bootstrap

Tutorial diatas adalah dasar – dasar blade jika sudah paham kita akan belajar membuat template sederhana dari bootstrap dengan memanfaatkan blade templating. Secara sederhana untuk kerangka membuat layout dengan blade dan membagi menjadi bagian kecil seperti gambar dibawah ini.



Untuk persiapan membuat templating sederhana menggunakan laravel ini, berikut yang perlu dipersiapkan :

- Download template bootstrapnya disini [\*\*sb-admin\*\*](#)
- Buka projectlaravel yang sudah dibahas pada tutorial sebelumnya lalu tambahkan folder **layouts** dan **includes** didalam folder **views**.
- Dari hasil download template bootstrap diatas saya bagi menjadi beberapa bagian **header**, **sidebar** dan **content**. Untuk pembagian sebenarnya terserah sesuai keinginan anda, bisa anda bagi lagi menjadi bagian yang lebih kecil. Buatlah file didalam folder **layouts** dengan nama **master.blade.php** dan didalam folder **includes** buat file bernama **header.blade.php** dan **sidebar.blade.php**.

Pada tahap pertama didalam folder sb-admin hasil template download bootstrap, copy folder **css**, **js**, **font awesome** dan folder **font**, lihat gambar dibawah ini.

css	6/27/2014 2:09 AM	File folder
font-awesome-4.1.0	6/27/2014 2:09 AM	File folder
fonts	6/27/2014 2:09 AM	File folder
js	6/27/2014 2:09 AM	File folder

Folder diatas pindahkan ke folder **assets** didalam folder **public** project anda, kalau ditutorial ini **projectlaravel**. Jika belum ada folder assets buat dahulu. Jika sudah buka file bernama **blank-page.html** di folder template bootstrap, copy semua code paste ke file **master.blade.php**.

Lalu di file **master.blade.php** cari code dibawah ini :

```
1 < link href ="css/bootstrap.min.css" rel ="stylesheet">
2 < link href ="css/sb-admin.css" rel ="stylesheet">
3 < link href ="font-awesome-4.1.0/css/font-awesome.min.css" rel
4 ="stylesheet" type="text/css">
5 //Untuk spasi diantara tanda < dan link silahkan dihapus
```

Ganti menjadi seperti ini

```
1 {{ HTML::style('assets/css/bootstrap.min.css') }}
2 {{ HTML::style('assets/css/sb-admin.css') }}
3 {{ HTML::style('assets/font-awesome-4.1.0/css/font-awesome.min.css') }}
```

Dan code paling bawah sendiri :

```
1 < script src="js/jquery-1.11.0.js">
2 < script src="js/bootstrap.min.js">
3
4 //Untuk spasi diantara tanda < dan script silahkan dihapus
```

Ganti menjadi seperti ini

```
1 {{ HTML::script('assets/js/jquery-1.11.0.js') }}
2 {{ HTML::script('assets/js/bootstrap.min.js') }}
```

Selanjutnya code di master.blade.php kita bagi atau pecah beberapa bagian yaitu header, sidebar dan content. Pertamkali kita ambil code untuk header. Ambil atau cut code di **master.blade.php** diantara tanda `<!-- Brand and toggle get grouped for better mobile display -->` sampai tanda `<!-- Sidebar Menu Items - These collapse to the responsive navigation menu on small screens -->` dan paste ke header.blade.php. Jadi di file header.blade.php ada sekitar kurang lebih 113 bari code. Di dalam file master.blade.php yang codenya kita ambil atau pindahkan tadi ganti code seperti dibawah ini.

```
1 @include('includes.header')
```

Jadi akan Nampak seperti ini di master.blade.php

```
39 <!-- Navigation -->
40 <nav class="navbar navbar-inverse navbar-fixed-top" role="navigation">
41 <!-- Brand and toggle get grouped for better mobile display -->
42 @include('includes.header')
43 <!-- Sidebar Menu Items - These collapse to the responsive navigation menu on small screens -->
44 <div class="collapse navbar-collapse navbar-ex1-collapse">
```

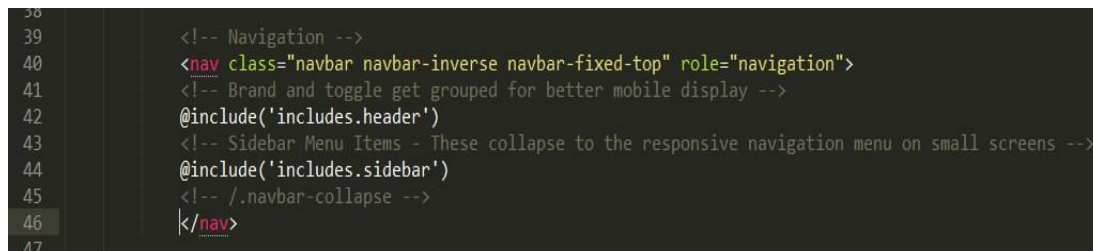
Sekarang masih di master.blade.php kali ini kita akan mengambil code untuk sidebar.blade.php. ambil code diantara `<!-- Sidebar Menu Items - These collapse to the responsive navigation menu on small screens -->` sampai dengan `<!-- /.navbar-collapse -->` paste atau taruh dalam file **sidebar.blade.php**. tempat code yang baru kita ambil silahkan ditambahkan code berikut ini.

---

```
1 @include('includes.sidebar')
```

---

Harusnya tampilan code di master.blade.php anda nampak seperti gambar dibawah ini.



```
38
39 <!-- Navigation -->
40 <nav class="navbar navbar-inverse navbar-fixed-top" role="navigation">
41 <!-- Brand and toggle get grouped for better mobile display -->
42 @include('includes.header')
43 <!-- Sidebar Menu Items - These collapse to the responsive navigation menu on small screens -->
44 @include('includes.sidebar')
45 <!-- /.navbar-collapse -->
46 </nav>
47
```

Untuk selanjutnya yaitu cari di folder **views** file yang bernama **profile.php** lalu rubah namanya / rename menjadi **profile.blade.php**. Kemudian isi file itu ganti dengan code dibawah ini.

---

```
1 @extends('layouts.master')
2 @section('content')
3 < div class="row">
4 < div class="col-lg-12">
5 < h1 class="page-header">
6 Halaman
7 < small>Profile< /small>
8 < /h1>
9 Profile ID / Nama : {{{ $id }}} / {{{ $nama }}}
10 < /div>
11 @stop
12 //Untuk spasi diantara tanda < dan div, /div, h1 ,/h1 ,small, /small
13 silahkan dihapus
14
```

---

Kembali buka master.blade.php diantara `<!-- Page Heading -->` sampai dengan `<!-- /.row -->` code didalamnya silahkan dihapus dan diganti dengan code berikut.

---

```
1 @yield('content')
```

---

Sehingga di master.blade.php menjadi seperti gambar ini.

```

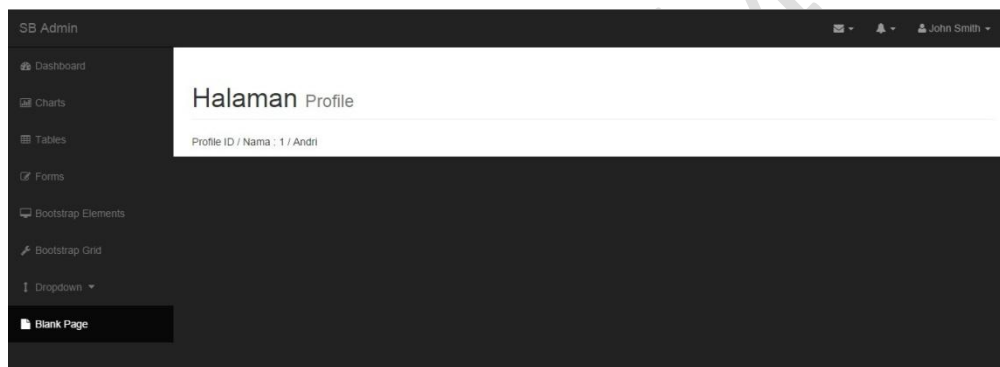
48     <div id="page-wrapper">
49
50         <div class="container-fluid">
51
52             <!-- Page Heading -->
53             @yield('content')
54             <!-- /.row -->
55
56         </div>
57         <!-- /.container-fluid -->
58
59     </div>
60     <!-- /#page-wrapper -->

```

Dari sini kita sudah bisa melihat hasil kerja kita. Ketika url di browser anda seperti ini.

1 <http://localhost/projectlaravel/public/profile/1/Andri>

Maka hasilnya seperti ini



Sekarang kita rubah sedikit didalam file controller yang bernama **ProfileController.php**. Silahkan dirubah atau diganti menjadi seperti ini.

```

1  < ?php
2  class ProfileController extends BaseController {
3
4      public function index()
5      {
6          return View::make('home');
7      }
8
9      public function profile($nama)
10     {
11         $data =
12         [
13             'nama' => $nama
14         ];
15         return View::make('profile', $data);
16     }
17 }

```

---

```
14
15 }
16
17 //Untuk spasi diantara tanda < dan ?php silahkan dihapus
18
19
20
```

---

Kemudian buka file **profile.blade.php** yang berada dalam folder views. Rubah atau ganti code dibawah ini.

---

```
1 Profile ID / Nama : {{{ $id }}} / {{{ $nama }}}
```

---

Menjadi seperti ini

---

```
1 Nama : {{{ $nama }}}
```

---

Jika sudah selesai buat file bernama **home.blade.php** dan tempatkan di dalam folder **views**, lalu isi dengan code berikut ini.

---

```
1 @extends('layouts.master')
2 @section('content')
3 < div class="row">
4     < div class="col-lg-12">
5         < h1 class="page-header">
6             Halaman
7             < small>Beranda< /small>
8         < /h1>
9         Ini Halaman Home / Beranda
10    < /div>
11 @stop
12 //Untuk spasi diantara tanda < dan div, /div, h1, /h1, small, /small
13 silahkan dihapus
14
```

---

Untuk selanjutnya buka file **sidebar.blade.php** didalam folder includes. Dan ganti code menjadi seperti dibawah ini.

---

```
1 < div class="collapse navbar-collapse navbar-ex1-collapse">
2     < ul class="nav navbar-nav side-nav">
3         < li>
4             < a href="{{ URL::to('/') }}">< i class="fa fa-fw
5             fa-dashboard">< /i> Home< /a>
6             < /li>
7             < li>
8                 < a href="{{ URL::to('profile/Andri') }}">< i
9                 class="fa fa-fw fa-user">< /i> Profile< /a>
10
```

---

---

```
9         < /li>
10     < /ul>
11 < /div>
12 //Untuk spasi diantara tanda < dan div, /div, ul, /ul, li, /li, a, /a, i,
    i/ silahkan dihapus
```

---

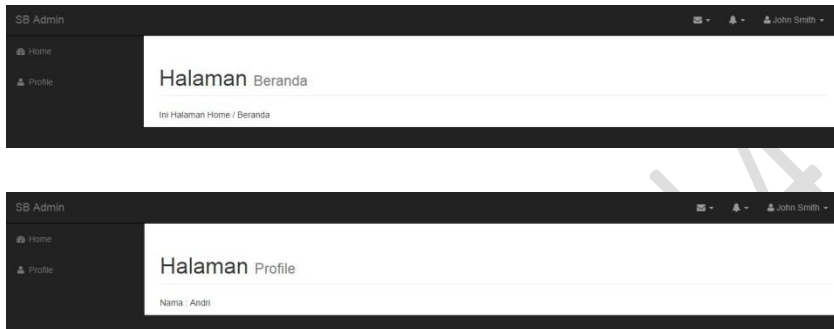
Silahkan dicoba dijalankan dengan mengetikan url seperti ini

---

```
1 http://localhost/projectlaravel/public/
2
3 http://localhost/projectlaravel/public/profile/Andri
```

---

Sekarang menu sidebar sudah berfungsi silahkan dicoba



*Catatan : untuk tutorial blade templating ini masih sangat sederhana jadi silahkan dikembangkan lagi sesuai keinginan anda. Jika anda tidak mengikuti tutorial sebelumnya dari install, routing, controller dan view jika anda masih bingung saya sarankan untuk mempelajarinya juga atau mempelajari ulang.*

---

# BAB IV

## 4. Belajar Dasar Form Framework Laravel Dan Latihan Membuat Form Sederhana

Form biasanya digunakan untuk menginputkan, menampung, memasukan data. Jika yang sudah belajar programming pasti lebih tau apa itu form. Di laravel sudah ada class form yang akan mempermudah dalam pembuatan form. Dengan menggunakan form bawaan laravel untuk pemula memang agak ribet, bagi yang sudah terbiasa maka akan tau keuntungan dari menggunakan form bawaan laravel daripada membuat form manual. Untuk itu kita harus tau terlebih dahulu tentang dasar – dasar yang ada di form laravel.

### Dasar – dasar Form Laravel

#### 1. Pembuka Form

Biasanya untuk membuat form kita memerlukan seperti code dibawah ini.

```
1 < form method="POST" action="" >
2 ...
3 < /form>
```

Didalam laravel kita dapat menulisnya seperti ini.

```
1 {{ Form::open(array('url' => '...')) }}
2 ...
3 {{ Form::close() }}
```

Kita sebelumnya juga sudah belajar tentang blade, dengan code form seperti diatas akan pasti lebih enak. Form diatas method default nya adalah POST, bagaimana kalau mau mengganti method menjadi GET, Lihat code dibawah ini.

```
1 Form::open(array('url' => 'user', 'method' => 'GET'))
```

Tidak hanya **POST** dan **GET** yang bisa kita gunakan untuk selain itu seperti **PUT** dan **DELETE** pun juga bisa.

Untuk agar bisa digunakan upload file kita biasanya menggunakan **enctype='multipart/form-data'** di form laravel kita bisa menggunakan **'files' => true** code seperti ini.



---

```
1 Form::open(array('url' => 'profile', 'files' => true))
```

---

## 2. Label

Dalam membuat label dengan form bawaan laravel code seperti dibawah ini.

---

```
1 Form::label('nama', 'Nama')
```

---

## 3. Text, Text Area Password, Hidden Field dan Button.

Jika kita biasanya untuk membuat form input text, textarea, password dan hidden file seperti ini.

---

```
1 < input name="username" type="text">
2
3 < textarea name="alamat" cols="50" rows="10"> < /textarea>
4
5 < input name="password" type="password" >
6
7 < input name="id" type="hidden">
8
9 < input type="submit" value="Submit">
```

---

Di Framework Laravel anda bisa menuliskan seperti dibawah ini.

---

```
1 Form::text('username')
2
3 Form::textarea('alamat')
4
5 Form::password('password')
6
7 Form::hidden('id')
8
9 Form::submit('Submit')
```

---

Untuk memberi nilai default contohnya seperti ini.

---

```
1 Form::text('nama', 'SeputarPemrograman')
```

---

## 4. Checkbox dan Radio

Biasanya untuk buat Checkbox dan Radio seperti ini.

---

```
1 < input name="nama" type="checkbox" value="SeputarPemrograman">
2
3 < input name="nama" type="radio" value=" SeputarPemrograman ">
```

---

Jika menggunakan Form bawaan Laravel maka menjadi seperti ini.

```
1 {{Form::checkbox('nama', ' SeputarPemrograman ')}}  
2  
3 {{Form::radio('nama', ' SeputarPemrograman ')}}
```

Agar secara default tercentang atau Checked anda dapat menggunakan **true** seperti ini.

```
1 Form::checkbox('nama', ' SeputarPemrograman ', true);  
2 Form::radio('nama', ' SeputarPemrograman ', true);
```

## 5. File Input

Anda dapat menggunakan seperti dibawah ini.

```
1 Form::file('image')
```

## 6. Drop-Down List

Jika biasanya anda membuat drop-down list dengan code sepanjang ini.

```
1 < select name="jeniskelamin">  
2 < option value="L">Laki - Laki< /option>  
3 < option value="P">Perempuan< /option>  
4 < /select>
```

Dengan menggunakan Form bawaan laravel akan menjadi lebih pendek.

```
1 Form::select('jeniskelamin', array('L' => 'Laki - Laki', 'P' =>  
    'Perempuan'))
```

Untuk membuat list terseleksi secara default anda dapat menggunakan seperti ini.

```
1 Form::select('jeniskelamin', array('L' => 'Laki - Laki', 'P' =>  
    'Perempuan'), 'L')
```

Itu adalah dasar – dasar form dari bawaan framework laravel dan masih banyak yang lain silahkan ke website laravel.com

# Membuat Form Sederhana

Tutorial diatas telah menjelaskan dasar – dasar dari penggunaan form laravel. Selanjutnya kita akan membuat form laravel seperti gambar dibawah ini.

Pertama buat dahulu file didalam folder **views** bernama **formsederhana.blade.php** dan isikan dengan code berikut ini.

```

1  @extends('layouts.master')
2
3  @section('content')
4
5      < div class="row">
6
7          < div class="col-lg-12">
8
9              < h1 class="page-header">
10
11                  Halaman
12
13                  Form Sederhana
14
15              < /h1>
16
17              {{ Form::open(array('url' => 'formsederhana')) }}
18
19              < div class="form-group">
20
21                  {{ Form::label('nama', 'Nama') }}
22
23                  {{ Form::text('nama', null,
24                      array('class' => 'form-control', 'placeholder'=>'masukkan nama')) }}
25
26              < /div>
27
28              < div class="form-group">
29
30                  {{ Form::label('jeniskelamin', 'Jenis Kelamin') }}

```

---

```

28
29         {{ Form::select('jeniskelamin', array('L' => 'Laki -
30 Laki', 'P' => 'Perempuan'), null, array('class' => 'form-
31 control', 'placeholder' => 'Pili Jenis Kelamin')) }}
32
33         < /div>
34
35         < div class="form-group">
36
37             {{ Form::label('alamat', 'Alamat') }}
38
39             {{ Form::textarea('alamat', null, array('class' => 'form-
40 control', 'placeholder' => 'masukkan alamat')) }}
41
42             < /div>
43
44             {{ Form::submit('Kirim', array('class' => 'form-control')) }}
45
46             {{ Form::close() }}
47
48         < /div>
49
50     @stop
51
52     // Hilangkan spasi antara < dan div, /div, h1, /h1
53

```

---

Setelah itu buka controller **ProfileController.php** dan tambahkan code / fungsi berikut ini, dibawah fungsi profile.

---

```

1  public function formsederhana()
2  {
3      return View::make('formsederhana');
4  }
5
6  public function postformsederhana()
7  {
8      $nama = Input::get('nama');
9      $jeniskelamin = Input::get('jeniskelamin');
10     $alamat = Input::get('alamat');
11
12     return 'Nama : '.$nama.< br/> Jenis Kelamin : '.$jeniskelamin.<
13     br/> Alamat '.$alamat;
14 }
15
16 // Hilangkan spasi antara < dan br/

```

---

Lalu buka **routes.php** dan tambahkan code dibawah ini.

---

```
1 Route::get('formsederhana', 'ProfileController@formsederhana');
2
3 Route::post('formsederhana', 'ProfileController@postformsederhana');
```

---

Dan yang terakhir buka file **sidebar.blade.php** dan tambahkan code ini.

---

```
1 < li>
2     < a href="{{ URL::to('formsederhana') }}">< i class="fa fa-fw fa-
3 edit">< /i> Form Sederhana< /a>
4 < /li>
5 // Hilangkan spasi antara < dan li, /li, a, /a, i, i/
```

---

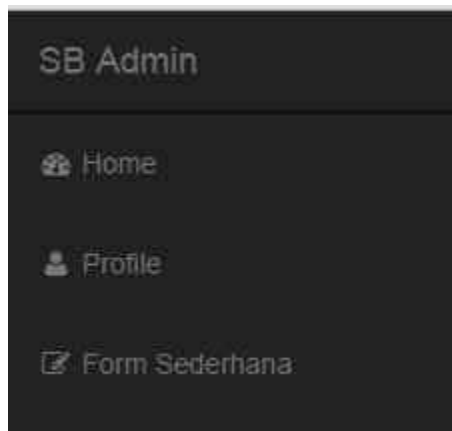
Dibawahnya code.

---

```
1 < li>
2     < a href="{{ URL::to('profile/Andri') }}">< i class="fa fa-fw fa-
3 user">< /i> Profile< /a>
4 < /li>
5 // Hilangkan spasi antara < dan li, /li, a, /a, i, i/
```

---

Maka di sidebar akan muncul menu baru seperti gambar berikut.



Selesai, sekarang coba klik menu form sederhana maka akan muncul seperti gambar form diatas.

*Catatan : tutorial ini bisa diikuti jika sudah mengikuti tutorial sebelumnya yaitu "Memfaatkan Blade Templating Laravel Untuk Memadukan Template Bootstrap".*

---

# BAB V

## 5. Cara Menggunakan Migrations dan Schema Builder di Framework Laravel

Migrations (migrasi) adalah sebuah fasilitas di Laravel digunakan untuk mempermudah kita ketika ada perubahan dalam database. Schema Builder digunakan untuk membuat sebuah skema database. Dengan menggunakan migrations dan schema builder kita untuk membuat database tidak perlu repot – repot membuka phpmyadmin, sqlyog, ataupun aplikasi lain untuk membuat database. Dengan migrations dan schema builder juga akan lebih mudah ketika kita membuat project besar dan banyak developer.

### Dasar atau Perintah Migrations (migrasi)

#### 1. Creating Migrations (Membuat Migrasi)

Untuk migrasi kita bisa menggunakan **migrate:make**. Untuk membuat migrasi baru kita dapat menggunakan perintah dibawah ini.

```
1 php artisan migrate:make create_users_table
```

Perintah diatas akan membuat file didalam folder php **app/database/migrations**. Nanti di file itulah kita bisa membuat schema databasenya. **create\_users\_table** anda ganti sesuai dengan kebutuhan.

#### 2. Running Migrations (Menjalankan Migrasi)

Untuk menjalankan atau memigrasi schema yang telah dibuat sebelumnya anda dapat menggunakan perintah dibawah ini.

```
1 php artisan migrate
```

Perintah diatas digunakan untuk menjalankan semua migrasi yang ada. Jika anda hanya ingin menjalankan path atau vendor tertentu bisa menggunakan perintah ini.

```
1 php artisan migrate --path=app/foo/migrations
2
3 php artisan migrate --package=vendor/package
```

#### 3. Rolling Back Migrations

Kita juga bisa merollback database yang telah dibuat sebelumnya dengan perintah berikut ini.

```
1 php artisan migrate:rollback
```

Perintah diatas hanya merollback data terakhir saja jika ingin merollback keseluruhan dapat menggunakan perintah berikut ini.

```
1 php artisan migrate:reset
```

## Schema Builder (Skema Builder)

Skema Builder seperti yang sudah katakan sejak awal, Skema Builder digunakan untuk membuat skema database kita berupa table dan field.

### 1. Creating & Dropping Tables (Membuat dan Menghapus tabel)

Berikut ini adalah script atau perintah dasar untuk membuat tabel.

```
1 Schema::create('profiles', function($table)  
2 {  
3     });  
4
```

Untuk rename tabel bisa menggunakan perintah berikut.

```
1 Schema::rename($from, $to);
```

Jika anda memiliki lebih dari satu koneksi database bisa menggunakan perintah ini dan dipilih menurut koneksi yang diinginkan.

```
1 Schema::connection('koneksi1')->create(' profiles ', function($table)  
2 {  
3     });  
4
```

Untuk menghapus tabel yang telah dibuat bisa menggunakan perintah dibawah ini.

```
1 Schema::drop(' profiles ');
```

atau

```
1 Schema::dropIfExists(' profiles ');
```

## 2. Adding, Renaming & Dropping Columns (Menambah, Mengubah & Menghapus Kolom)

Adakalanya ketika kita bikin program ingin menambahkan kolom pada suatu tabel, didalam skema builder ini bisa menggunakan perintah ini.

```
1 Schema::table('profiles', function($table)
2 {
3     $table->string('notlep');
4 });
```

Untuk mengubah kolom bisa menggunakan perintah **renameColumn** seperti dibawah ini.

```
1 Schema::table('profiles', function($table)
2 {
3     $table->renameColumn('from', 'to');
4 });
```

Sedangkan untuk menghapus kolom bisa menggunakan perintah berikut ini.

```
1 Schema::table('profiles', function($table)
2 {
3     $table->dropColumn('notlep');
4 });
```

Untuk menghapus kolom lebih dari satu bisa seperti ini.

```
1 $table->dropColumn('notlep');
```

Menjadi

```
1 $table->dropColumn(array('notlep', 'email'));
```

## 3. Adding Indexes & Foreign Keys

Menambahkan index pada tabel tertentu dan kolom tertentu bisa menggunakan perintah dibawah ini.

```
1 $table->string('email')->unique();
```

Sedangkan untuk menambahkan foreign key atau kunci tamu bisa menggunakan perintah ini.

```
1 $table->integer('user_id')->unsigned();
2 $table->foreign('user_id')->references('id')->on('users');
```



Itu adalah dasar dasar dari migrasi dan skema builder untuk lebih lengkapnya bisa langsung ke <http://laravel.com/docs/4.2/migrations> dan <http://laravel.com/docs/4.2/schema>.

## Menerapkan Migration (Migrasi) dan Schema Builder (Skema Builder)

Setelah kita belajar dasar dasar dari migrasi dan skema builder, saatnya untuk mencoba secara langsung agar kita lebih tahu cara penggunaannya dari fungsi yang dijelaskan diatas. Kita buka dahulu project laravel yang telah dibuat di tutorial sebelumnya atau jika belum bisa mengikuti tutorial “Belajar Install Framework Laravel Dengan Composer Maupun Tanpa Composer”. Dan buat database bernama **belajarlaravel**. Buka folder app/config dan cari file bernama database.php. silahkan cari code seperti dibawah ini.

```
1  'mysql' => array(
2
3      'driver' => 'mysql',
4
5      'host' => 'localhost',
6
7      'database' => 'database',
8
9      'username' => 'root',
10
11     'password' => '',
12
13     'charset' => 'utf8',
14
15     'collation' => 'utf8_unicode_ci',
16
17     'prefix' => '',
18
19 ),
```

Rubah menjadi

```
1  'mysql' => array(
2
3      'driver' => 'mysql',
4
5      'host' => 'localhost',
6
7      'database' => 'belajarlaravel', //
8  nama database yang telah dibuat
```

---

```
9
10         'username' => 'root', // user
11
12         'password' => '', //password untuk
13 xampp default kosong.
14
15         'charset' => 'utf8',
16
17         'collation' => 'utf8_unicode_ci',
18
19         'prefix' => '',
20
21     ),
```

---

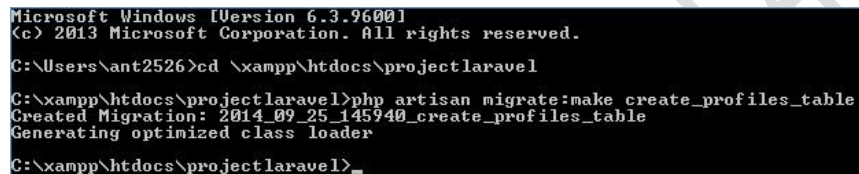
Selanjutnya ketikan perintah ini ke command prompt / cmd

---

```
1 php artisan migrate:make create_profiles_table
```

---

lebih jelasnya lihat gambar dibawah ini.



```
Microsoft Windows [Version 6.3.9600]
(c) 2013 Microsoft Corporation. All rights reserved.

C:\Users\ant2526>cd \xampp\htdocs\projectlaravel

C:\xampp\htdocs\projectlaravel>php artisan migrate:make create_profiles_table
Created Migration: 2014_09_25_145940_create_profiles_table
Generating optimized class loader

C:\xampp\htdocs\projectlaravel>_
```

**cd \xampp\htdocs\projectlaravel**, digunakan untuk mengalihkan ke folder projectlaravel

**php artisan migrate:make create\_profiles\_table**, digunakan untuk perintah membuat migrasi.

Jika sudah berhasil seperti gambar diatas lihat di folder project anda klo disini projectlaravel di folder **app/database/migrations** dan akan terdapat file seperti ini 2014\_09\_25\_145940\_create\_profiles\_table.php dan berisi seperti ini.

---

```
1 < ?php
2 use Illuminate\Database\Schema\Blueprint;
3 use Illuminate\Database\Migrations\Migration;
4 class CreateProfilesTable extends Migration {
5
6     /**
7      * Run the migrations.
8      *
9      * @return void
10
11
12
```

---

---

```

13         */
14
15     public function up()
16     {
17         //
18     }
19
20     /**
21      * Reverse the migrations.
22      *
23      * @return void
24      */
25     public function down()
26     {
27         //
28     }
29 }
30
31
32
33
34
35
36
37
38
39

```

---

Selanjutnya didalam code diatas rubah menjadi seperti dibawah ini.

---

```

1  < ?php
2  use Illuminate\Database\Schema\Blueprint;
3  use Illuminate\Database\Migrations\Migration;
4
5  class CreateProfilesTable extends Migration {
6
7      /**
8       * Run the migrations.
9       *
10      * @return void
11      */
12
13     public function up()
14
15
16

```

---

---

```

17         {
18
19             Schema::create('profiles', function($table)
20             {
21                 $table->increments('id');
22
23                 $table->string('nama')->length(100);
24
25                 $table->enum('jeniskelamin', array('L', 'P'));
26
27                 $table->string('alamat')->nullable();
28
29                 $table->timestamps();
30             });
31         }
32
33         /**
34          * Reverse the migrations.
35          *
36          * @return void
37          */
38
39         public function down()
40         {
41             Schema::drop('profiles');
42         }
43     }
44 }
45
46 }
47
48
49
50
51
52

```

---

Selesai silahkan disimpan dan jalankan perintah berikut ini di command prompt / cmd.

---

**1 php artisan migrate**

---

Lebih jelasnya lihat gambar dibawah ini.

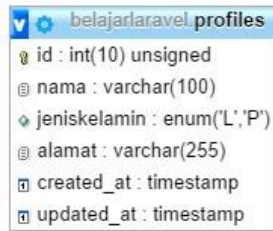
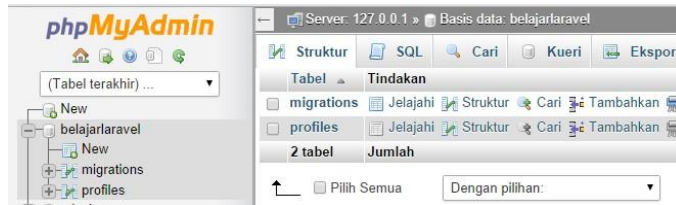


```

C:\xampp\htdocs\projectlaravel>php artisan migrate
Migration table created successfully.
Migrated: 2014_09_25_145940_create_profiles_table
C:\xampp\htdocs\projectlaravel>

```

Sampai sini silahkan cek apakah tabel **profiles** yang anda buat masuk kedalam database **belajarlaravel** seperti ini.



Sampai disini anda sudah sukses membuat migration sederhana menggunakan laravel, jika anda terlalu keberatan dalam membuat database dengan cara seperti ini maka anda bisa membuat database secara manual melalui phpmyadmin atau menurut cara anda sendiri.

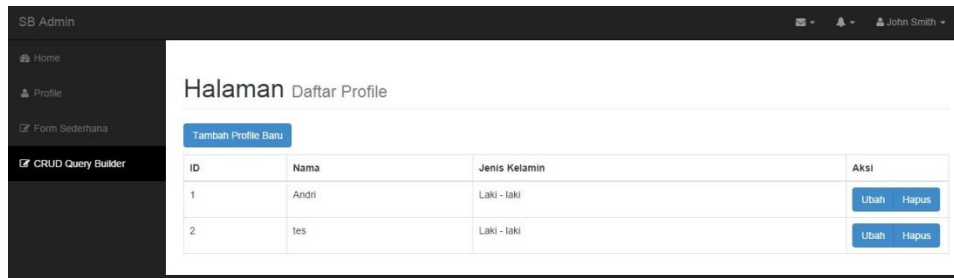
## BAB VI

### 6. Cara Membangun Create, Read, Update dan Delete Menggunakan Query Builder di Laravel

Create, read, update dan delete biasanya disingkat **CRUD**. Operasi CRUD ini digunakan untuk membuat, membaca, mengubah dan menghapus data dalam database. Operasi CRUD adalah operasi umum untuk mengolah data dan biasanya digunakan juga untuk belajar pertama kali untuk mengolah data sederhana bagi pemula.

Apa itu **query builder**, jika anda pernah belajar framework codeigniter pasti sudah mengenal, di framework codeigniter biasa disebut active record. Sedangkan di framework laravel disebut sebagai query builder. Dengan query builder code yang kita bikin akan mudah atau enak dibaca. Disamping itu ketika ingin migrasi ke database lain anda tidak perlu mengubah query (catatan hanya database yang didukung oleh laravel saja). Jika anda belum pernah atau belum mendengar sama sekali. Query builder sama dengan sql / query yang biasa kita pakai contoh `select * from`

tabel; Cuma di query builder lebih dipersingkat lagi dan dibikin agar support dengan beberapa database. Untuk lebih jelasnya silahkan lihat tutorial query builder di laravel.com



Kali ini kita akan membuat operasi crud sederhana seperti tampilan gambar diatas dengan memanfaatkan database yang kita buat pada tutorial “Cara Menggunakan Migrations dan Schema Builder di Framework Laravel”. Jika anda belum terbiasa dengan cara ini, anda bisa langsung membuatnya di phpmyadmin dengan struktur tabel dan nama database sebagai berikut.

Nama Database : belajarlaravel

Nama Tabel : profiles

Struktur tabel silahkan dilihat gambar dibawah ini atau SQL yang saya sertakan.



```
1 CREATE TABLE IF NOT EXISTS `profiles` (  
2  
3   `id` int(10) unsigned NOT NULL AUTO_INCREMENT,  
4   `nama` varchar(100) COLLATE utf8_unicode_ci NOT NULL,  
5   `jeniskelamin` enum('L','P') COLLATE utf8_unicode_ci NOT NULL,  
6   `alamat` varchar(255) COLLATE utf8_unicode_ci DEFAULT NULL,  
7   `created_at` timestamp NOT NULL DEFAULT '0000-00-00 00:00:00',  
8   `updated_at` timestamp NOT NULL DEFAULT '0000-00-00 00:00:00',  
9  
10  PRIMARY KEY (`id`)  
11  
12  
13  
14  
15
```

---

```
16 ) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci
17 AUTO_INCREMENT=4 ;
```

---

Jika telah selesai membuat database diatas silahkan di konfigurasi dulu, **database.php** didalam folder **app/config**.

---

```
1  'mysql' => array(
2
3      'driver' => 'mysql',
4
5      'host'   => 'localhost',
6
7      'database' => 'belajarlaravel',
8
9      'username' => 'root',
10
11     'password' => '',
12
13     'charset' => 'utf8',
14
15     'collation' => 'utf8_unicode_ci',
16
17     'prefix' => '',
18 ),
19
```

---

Kemudian kita lanjutkan dengan membuat controller lebih dahulu bernama **CrudController.php** letakkan dalam folder **app/controllers**. Jika sudah silahkan isi dengan code dibawah ini.

---

```
1  < ?php
2  class CrudController extends \BaseController {
3      public function index()
4      {
5          $profiles = DB::table('profiles')->paginate(5);
6          $profiles =
7          [
8              'profiles' => $profiles
9          ];
10         return View::make('crud.index', $profiles);
11     }
12 }
13
```

---

Sekarang buat folder bernama **crud** didalam folder **app/views**. Setelah itu didalam folder **crud** buat file bernama **index.blade.php** dan isikan dengan code dibawah ini.

---

```
1  @extends('layouts.master')
```

---

```

2  @section('content')
3  < div class="row">
4      < div class="col-lg-12">
5          < h1 class="page-header">
6              Halaman
7          < /h1>
8          @if (Session::has('message'))
9              {{ Session::get('message') }}
10         @endif
11         < p>< a href="{{ URL::to('crud/create') }}" class="btn btn-
12         primary" role="button">Tambah Profile Baru< /a>< /p>
13         < div class="table-responsive">
14             < table class="table table-bordered table-
15             hover">
16                 < thead>
17                     < tr>
18                         < th>ID< /th>
19                         < th>Nama< /th>
20                         < th>Jenis Kelamin< /th>
21                         < th width="146">Aksi< /th>
22                     < /tr>
23                 < /thead>
24                 < tbody>
25                     @foreach($profiles as $value)
26                         < tr>
27                             < td>{{{ $value->id }}}< /td>
28                             < td>{{{ $value->nama }}}< /td>
29                             < td>{{{ $value->jeniskelamin ==
30                             'L' ? 'Laki - laki' : 'Perempuan' }}}< /td>
31                             < td>
32                                 < div class="btn-group">
33                                     < a href="{{ URL::to('crud/edit/' . $value->id) }}"
34                                     class="btn btn-primary">Ubah< /a>
35                                     < a href="{{ URL::to('crud/destroy/' . $value->id)
36                                     }}" class="btn btn-primary">Hapus< /a>
37                                 < /div>
38                             < /td>
39                         < /tr>
40                     @endforeach
41                 < /tbody>
42             < /table>
43         < /div>
44         {{ $profiles->links() }}
45     < /div>
46 @stop

```

// agar berjalan dengan baik hilangkan spasi sesudah tandan <

Selanjutnya buka file **routes.php** didalam folder **app** tambahkan code dibawah ini.



---

```
1 Route::get('crud', 'CrudController@index');
```

---

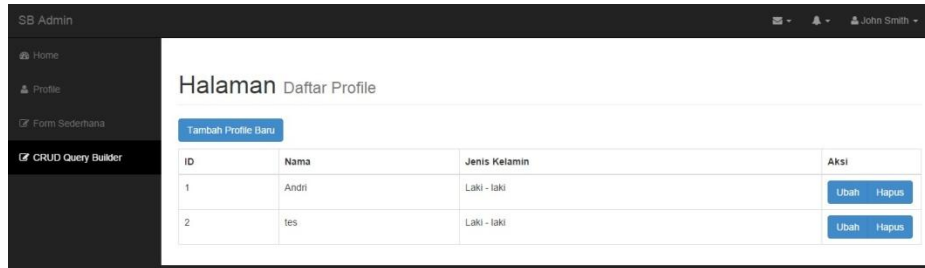
Sekarang silahkan diisi dulu data secara manual di phpmyadmin, terserah anda diisi data berapapun dan untuk menjalankan hasil code diatas ketikan url seperti ini.

---

```
1 http://localhost/projectlaravel/public/crud
```

---

maka akan tampil halaman seperti gambar dibawah ini.



Darisini anda sudah berhasil menampilkan data dari database. Selanjutnya kita akan membuat form tambah data profile. Tambahkan code berikut ini di controller bernama **CrudController.php** dibawah function **index**.

---

```
1 public function create()
2 {
3     return View::make('crud.create');
4 }
5 public function store()
6 {
7     $rules = array(
8         'nama' => 'required',
9         'jeniskelamin' => 'required',
10    );
11    $validator = Validator::make(Input::all(), $rules);
12    if ($validator->fails()) {
13        return Redirect::to('crud/create')->withErrors($validator)-
14    >withInput();
15    } else {
16        DB::table('profiles')->insert(
17            array(
18                'nama' => Input::get('nama'),
19                'jeniskelamin' => Input::get('jeniskelamin'),
20                'alamat' => Input::get('alamat')
21            )
22        );
23        Session::flash('message', 'Data Berhasil Ditambahkan');
24        return Redirect::to('crud');
25    }
```

---

---

```
26 }
27
28
29
```

---

Kemudian buat file bernama **create.blade.php** didalam folder **crud** yang sebelumnya sudah dibuat dan isikan code dibawah ini.

---

```
1  @extends('layouts.master')
2  @section('content')
3      < div class="row">
4          < div class="col-lg-12">
5              < h1 class="page-header">
6                  Halaman
7              < small>Tambah Profile< /small>
8              < /h1>
9              {{ Form::open(array('url' => 'crud/create')) }}
10
11              < div class="form-group">
12                  {{ Form::label('nama', 'Nama') }}
13                  {{ Form::text('nama', null, array('class' => 'form-
14 control','placeholder'=>'masukkan nama')) }}
15                  {{ '< div>'. $errors->first('nama'). '< /div>' }}
16              < /div>
17
18              < div class="form-group">
19                  {{ Form::label('jeniskelamin', 'Jenis Kelamin') }}
20                  {{ Form::select('jeniskelamin', array('L' => 'Laki -
21 laki', 'P' => 'Perempuan'), null, array('class' => 'form-
22 control','placeholder'=>'Pili Jenis Kelamin')) }}
23                  {{ '< div>'. $errors->first('jeniskelamin'). '< /div>' }}
24              < /div>
25
26              < div class="form-group">
27                  {{ Form::label('alamat', 'Alamat') }}
28                  {{ Form::textarea('alamat', null, array('class' => 'form-
29 control','placeholder'=>'masukkan alamat')) }}
30              < /div>
31              {{ Form::submit('SIMPAN', array('class' => 'form-control')) }}
32          {{ Form::close() }}
33      < /div>
34  < /div>
35  @stop
36  // agar berjalan dengan baik hilangkan spasi sesudah tandan <
```

---

Kemudian di **routes.php** tambahkan code dibawah ini.

---

```
1 Route::get('crud/create', 'CrudController@create');
2 Route::post('crud/create', 'CrudController@store');
```

---

Selesai jalankan dengan menekan tombol tambah profile baru atau ketikan url seperti ini.

---

```
1 http://localhost/projectlaravel/public/crud/create
```

---

Sehingga menampilkan form sebagai berikut.

---

Halaman Tambah Profile

Nama

Jenis Kelamin

Alamat

---

Silahkan dicoba input data dan lihat hasilnya.

Setelah membuat tambah data kita akan membuat update atau ubah data. Maka dari itu tambahkan code dibawah ini didalam controller **CrudController.php** dibawah function **store**.

---

```
1 public function edit($id)
2 {
3     $profilesbyid = DB::table('profiles')->where('id',$id)->first();
4     $profilesbyid =
5     [
6         'profilesbyid' => $profilesbyid
7     ];
8     return View::make('crud.edit', $profilesbyid);
9 }
10
11 public function update($id)
12 {
13     $rules = array(
14         'nama' => 'required',
15         'jeniskelamin' => 'required',
16     );
17     $validator = Validator::make(Input::all(), $rules);
18     if ($validator->fails()) {
19         echo "string";
20         return Redirect::to('crud/edit/'. $id)->withErrors($validator)-
21         >withInput();
22     }
```

---

---

```

22     } else {
23     DB::table('profiles')
24     ->where('id', $id)
25     ->update(array(
26         'nama' => Input::get('nama'),
27         'jeniskelamin' => Input::get('jeniskelamin'),
28         'alamat' => Input::get('alamat')
29     ));
30
31     Session::flash('message', 'Data Berhasil Diubah');
32     return Redirect::to('crud');
33 }
34
35
36

```

---

Jika sudah silahkan dibuat file bernama **edit.blade.php** didalam folder **crud** dan isikan code dibawah ini.

---

```

1  @extends('layouts.master')
2  @section('content')
3  < div class="row">
4      < div class="col-lg-12">
5          < h1 class="page-header">
6              Halaman
7              < small>Ubah Profile< /small>
8          < /h1>
9
10         {{ Form::model($profilesbyid, array('route' =>
11         array('crud.update', $profilesbyid->id))) }}
12         < div class="form-group">
13             {{ Form::label('nama', 'Nama') }}
14             {{ Form::text('nama', null, array('class' => 'form-
15             control', 'placeholder' => 'masukkan nama')) }}
16             {{ '< div>'. $errors->first('nama'). '< /div>' }}
17         < /div>
18
19         < div class="form-group">
20             {{ Form::label('jeniskelamin', 'Jenis Kelamin') }}
21             {{ Form::select('jeniskelamin', array('L' => 'Laki -
22             Laki', 'P' => 'Perempuan'), null, array('class' => 'form-
23             control', 'placeholder' => 'Pili Jenis Kelamin')) }}
24             {{ '< div>'. $errors->first('jeniskelamin'). '< /div>' }}
25         < /div>
26
27         < div class="form-group">
28             {{ Form::label('alamat', 'Alamat') }}
29             {{ Form::textarea('alamat', null, array('class' => 'form-
30             control', 'placeholder' => 'masukkan alamat')) }}
31         < /div>
32         {{ Form::submit('SIMPAN', array('class' => 'form-control')) }}
33     < /div>
34 < /div>

```

---

---

```

28
29         {{ Form::close() }}
30
31     < /div>
32 < /div>
33 @stop
34 // agar berjalan dengan baik hilangkan spasi sesudah tandan <
35

```

---

Tambahkan juga code dibawah ini didalam **routes.php**

---

```

1 Route::get('crud/edit/{id}', 'CrudController@edit');
2 Route::post('crud/update/{id}', array('as' => 'crud.update', 'uses' =>
  'CrudController@update'));

```

---

Sekarang coba jalankan untuk mengubah data dengan menekan tombol ubah maka akan tampil form seperti gambar ini.

Ubah data apapun dan klik tombol simpan dan silahkan amati apa yang terjadi.

Yang terakhir adalah membuat fungsi untuk menghapus data, silahkan tambahkan code berikut di controller CrudController.php dibawah function update.

---

```

1 public function destroy($id)
2 {
3     DB::table('profiles')->where('id', '=', $id)->delete();
4     Session::flash('message', 'Data Berhasil Dihapus');
5     return Redirect::to('crud');
6 }

```

---

Dan dialam routes.php tambahkan juga code dibawah ini.

---

```

1 Route::get('crud/destroy/{id}', 'CrudController@destroy');

```

---

Selesai silahkan dicoba untuk menghapus data dengan menekan tombol hapus.

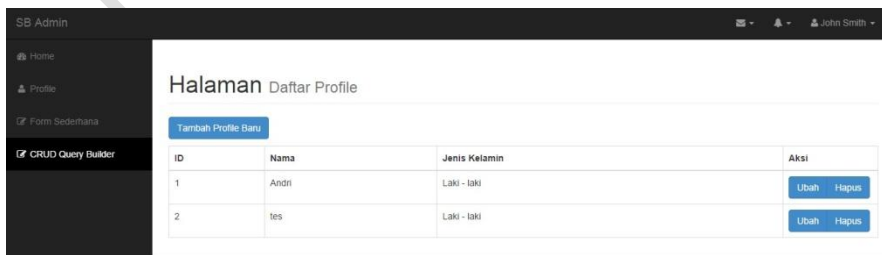
Itu adalah contoh dasar operasi CRUD menggunakan Query Builder bawaan Framework Laravel.

---

## BAB VII

### 7. Membuat Create, Read, Update dan Delete Menggunakan Eloquent ORM dan RESTful Resource Controllers Laravel

Pada tutorial sebelumnya membuat CRUD dengan Query Builder sudah saya jelaskan apa itu yang dimaksud CRUD. Ditutorial ini akan sedikit saya bahas lagi apa itu CRUD. Create, read, update dan delete biasanya disingkat CRUD. Operasi CRUD ini digunakan untuk membuat, membaca, mengubah dan menghapus data dalam database. Untuk CRUD Ditutorial ini kita akan memanfaatkan Eloquent ORM dan RESTful Resource Controllers. Eloquent ORM adalah ORM (Object Relational Mapping) yang dibundling bersama Laravel Framework. Dengan ORM kita akan lebih mudah dalam membuat proses CRUD. Sedangkan RESTful Resource Controllers adalah untuk mempermudah dalam membuat RESTful controllers. RESTful resource adalah semua hal yang bisa diakses dan ditransfer melalui web antara client dan server. Contohnya saja pada aplikasi facebook atau google plus. Pada suatu website mereka untuk login ke website mereka bisa menggunakan data pihak ketiga yaitu facebook dan google plus.



The screenshot shows a web application interface with a sidebar on the left containing links for 'Home', 'Profile', 'Form Sederhana', and 'CRUD Query Builder'. The main content area is titled 'Halaman Daftar Profile' and features a 'Tambah Profile Baru' button above a table. The table has four columns: 'ID', 'Nama', 'Jenis Kelamin', and 'Aksi'. It contains two rows of data. Each row has 'Ubah' and 'Hapus' buttons in the 'Aksi' column.

ID	Nama	Jenis Kelamin	Aksi
1	Andri	Laki - laki	Ubah Hapus
2	tes	Laki - laki	Ubah Hapus

Sebelum lanjut untuk membuat CRUD dengan Eloquent ORM dan RESTful Resource Controllers silahkan membuat database terlebih dahulu jika tidak mengikuti tutorial sebelumnya.

Nama Database : belajarlaravel

Nama Tabel : profiles

Untuk setting database silahkan disetting pada file **database.php** didalam folder **app/config**. Silahkan untuk isi disesuaikan saja dengan pengaturan anda sendiri, jika belum paham betul silahkan melihat tutorial sebelumnya “ Cara Membangun Create, Read, Update dan Delete Menggunakan Query Builder di Laravel”.

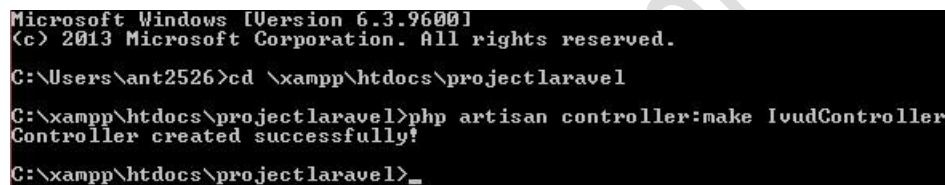
Setelah selesai buka project anda yang telah diinstall sebelumnya jika belum bisa install project silahkan mengikuti tutorial sebelumnya install laravel. membuat dan konfigurasi database selanjutnya buat file bernama **IvudController.php** didalam folder **app/controllers**. Jika anda ingin cara cepat bisa menggunakan perintah berikut ini pada artisan.

---

**1 php artisan controller:make IvudController**

---

Lebih jelasnya silahkan lihat gambar dibawah ini.



```
Microsoft Windows [Version 6.3.9600]
(c) 2013 Microsoft Corporation. All rights reserved.

C:\Users\ant2526>cd \xampp\htdocs\projectlaravel

C:\xampp\htdocs\projectlaravel>php artisan controller:make IvudController
Controller created successfully!

C:\xampp\htdocs\projectlaravel>_
```

**cd \xampp\htdocs\projectlaravel** , digunakan untuk mengarahkan kedalam folder project kita.

**php artisan controller:make IvudController** , perintah untuk membuat file controller tanpa membuat secara manual dan langsung terisi perintah dasar. Jika menggunakan ini dan berhasil menjalankan perintah diatas langsung lihat folder **app/controllers** maka disitu akan ada file baru.

Jika anda membuat secara manual file controller **IvudController.php**nya silahkan isi dengan code berikut ini, jika anda menggunakan perintah **controller:make** diatas silahkan diperhatikan saja code dibawah ini.

---

```
1  < ?php
2
3  class IvudController extends BaseController {
4
5      /**
6          * Display a listing of the resource.
7
8          *
```

---

---

```
9
10     * @return Response
11
12     */
13
14     public function index()
15     {
16         //
17     }
18
19     /**
20
21     * Show the form for creating a new resource.
22
23     *
24     * @return Response
25
26     */
27
28     public function create()
29     {
30         //
31     }
32
33     /**
34
35     * Store a newly created resource in storage.
36
37     *
38     * @return Response
39
40     */
41
42     public function store()
43     {
44         //
45     }
46
47
48     }
49
50
51     /**
52
53     * Display the specified resource.
54
55     *
56
```

---



---

```
57      * @param int $id
58
59      * @return Response
60
61      */
62      public function show($id)
63      {
64          //
65      }
66
67      /**
68       * Show the form for editing the specified resource.
69       *
70       * @param int $id
71       * @return Response
72       */
73      public function edit($id)
74      {
75          //
76      }
77
78      /**
79       * Update the specified resource in storage.
80       *
81       * @param int $id
82       * @return Response
83       */
84      public function update($id)
85      {
86          //
87      }
88
89      /**
90       * Remove the specified resource from storage.
91       */
92
93
94
95
96
97
98
99
100
101
102
103
104
```

---

---

```

105         *
106
107         * @param int $id
108
109         * @return Response
110
111         */
112     public function destroy($id)
113     {
114         //
115     }
116 }
117
118 // agar program berjalan lancar hapus spasi setelah tanda <
119
120
121
122
123
124
125
126
127
128
129
130

```

---

Masih dalam file **IvudController.php** sekarang silahkan diganti function **index** diatas menjadi seperti dibawah ini.

---

```

1 public function index()
2 {
3     $profiles = Profile::paginate(5);
4     $profiles =
5     [
6         'profiles' => $profiles
7     ];
8     return View::make('ivud.index', $profiles);
9 }

```

---

Kemudian buat file bernama **Profile.php** didalam folder **app/models** dan isikan dengan code dibawah ini.

---

```

1 < ?php class Profile extends Eloquent {
2
3 }
4

```

---

---

```
5 // agar program berjalan lancar hapus spasi setelah tanda <
```

---

Jika selesai langkah selanjutnya adalah membuat folder bernama **ivud** didalam folder **app/views**. Setelah itu didalam folder **ivud** buat file bernama **index.blade.php** dan isikan dengan code dibawah ini.

---

```
1 @extends('layouts.master')
2 @section('content')
3 < div class="row">
4     < div class="col-lg-12">
5         < h1 class="page-header">
6             Halaman
7             < small>Daftar Profile< /small>
8         < /h1>
9         @if (Session::has('message'))
10             {{ Session::get('message') }}
11         @endif
12         < p>< a href="{{ URL::to('ivud/create') }}" class="btn btn-
13 primary" role="button">Tambah Profile Baru< /a>< /p>
14         < div class="table-responsive">
15             < table class="table table-bordered table-
16 hover">
17                 < thead>
18                     < tr>
19                         < th>ID< /th>
20                         < th>Nama< /th>
21                         < th>Jenis Kelamin< /th>
22                         < th width="146">Aksi< /th>
23                     < /tr>
24                 < /thead>
25                 < tbody>
26                     @foreach($profiles as $value)
27                         < tr>
28                             < td>{{{ $value->id }}}< /td>
29                             < td>{{{ $value->nama }}}< /td>
30                             < td>{{{ $value->jeniskelamin ==
31 'L' ? 'Laki - laki' : 'Perempuan' }}}< /td>
32                             < td>
33                                 {{ Form::open(array('url' =>
34 'ivud/' . $value->id)) }}
35                                 < div class="btn-group">
36                                     < a href="{{ URL::to('ivud/' .
37 $value->id . '/edit') }}" class="btn btn-primary">Ubah< /a>
38                                     {{ Form::hidden('_method',
39 'DELETE') }}
40                                     {{Form::button('Hapus',
41 array('type' => 'submit', 'class' => 'btn btn-primary'))}}
42                                 < /div>
43                                 {{ Form::close() }}
44                             < /td>
45                         < /tr>
46                     @endforeach
47                 < /tbody>
48             < /table>
49         < /div>
```

---

---

```

42         {{$profiles->links()}}
43
44     < /div>
45 < /div>
46 @stop
47 // agar program berjalan lancar hapus spasi setelah tanda <
48
49

```

---

Selanjutnya buka file **routes.php** pada folder **app** dan tambahkan code dibawah ini.

---

```

1 Route::resource('ivud', 'IvudController');

```

---

Dengan code route diatas kita tidak perlu seperti tutorial kemarin membuat satu – satu routenya cukup satu baris sudah mewakili satu CRUD. Jika kita jalankan dengan mengetikkan url seperti ini dibrowser

---

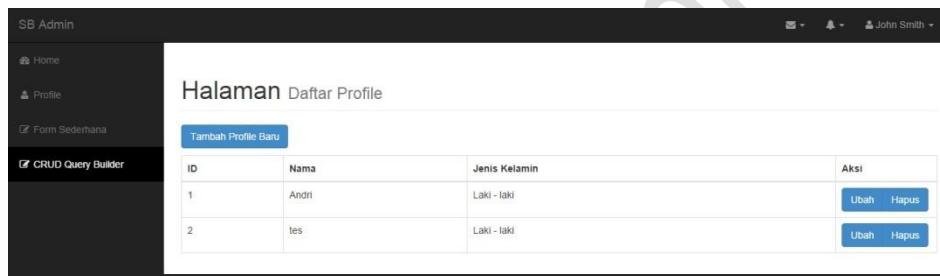
```

1 http://localhost/projectlaravel/public/ivud

```

---

hasilnya pun sama seperti tutorial CRUD dengan Query Builder.



Selanjutnya ganti code function **create** dan **store** di dalam file **IvudController.php**

---

```

1 public function create()
2 {
3     return View::make('ivud.create');
4 }
5
6 public function store()
7 {
8     $rules = array(
9         'nama' => 'required',
10        'jeniskelamin' => 'required',
11    );
12
13    $validator = Validator::make(Input::all(), $rules);
14
15    if ($validator->fails()) {
16        return Redirect::to('ivud/create')-

```

---

---

```

16 >withErrors($validator)->withInput();
17     } else {
18         $profile = new Profile;
19         $profile->nama = Input::get('nama');
20         $profile->jeniskelamin = Input::get('jeniskelamin');
21         $profile->alamat = Input::get('alamat');
22         $profile->save();
23     Session::flash('message', 'Data Berhasil Ditambahkan');
24     return Redirect::to('ivud');
25     }
26 }

```

---

Sekarang buat file didalam folder **app/views** bernama **create.blade.php** dan isikan code dibawah ini.

---

```

1  @extends('layouts.master')
2  @section('content')
3      < div class="row">
4          < div class="col-lg-12">
5              < h1 class="page-header">
6                  Halaman
7              < small>Tambah Profile< /small>
8              < /h1>
9              {{ Form::open(array('url' => 'ivud')) }}
10
11              < div class="form-group">
12                  {{ Form::label('nama', 'Nama') }}
13                  {{ Form::text('nama', null, array('class' => 'form-
14 control', 'placeholder' => 'masukkan nama')) }}
15                  {{ '< div>'. $errors->first('nama'). '< /div>' }}
16              < /div>
17
18              < div class="form-group">
19                  {{ Form::label('jeniskelamin', 'Jenis Kelamin') }}
20                  {{ Form::select('jeniskelamin', array('L' => 'Laki -
21 Laki', 'P' => 'Perempuan'), null, array('class' => 'form-
22 control', 'placeholder' => 'Pili Jenis Kelamin')) }}
23                  {{ '< div>'. $errors->first('jeniskelamin'). '< /div>' }}
24              < /div>
25
26              < div class="form-group">
27                  {{ Form::label('alamat', 'Alamat') }}
28                  {{ Form::textarea('alamat', null, array('class' => 'form-
29 control', 'placeholder' => 'masukkan alamat')) }}
30                  < /div>
31                  {{ Form::submit('SIMPAN', array('class' => 'form-control')) }}
32              {{ Form::close() }}
33          < /div>
34      < /div>

```

---

---

```
33 @stop
34
35 // agar program berjalan lancar hapus spasi setelah tanda <
36
```

---

Untuk menjalankan coba ketikan url seperti ini.

---

```
1 http://localhost/projectlaravel/public/ivud/create
```

---

Hasilnya akan seperti gambar dibawah ini.

---



---

Isi data lalu klik tombol simpan dan amati apa yang terjadi.

Selanjutnya silahkan ganti code function **edit** dan **update** pada file **IvudController.php** seperti dibawah ini.

---

```
1 public function edit($id)
2 {
3     $profilesbyid = Profile::findOrFail($id);
4     $profilesbyid =
5     [
6         'profilesbyid' => $profilesbyid
7     ];
8     return View::make('ivud.edit', $profilesbyid);
9 }
10 public function update($id)
11 {
12     $rules = array(
13         'nama' => 'required',
14         'jeniskelamin' => 'required',
15     );
16     $validator = Validator::make(Input::all(), $rules);
17     if ($validator->fails()) {
18         return Redirect::to('ivud/'.$id.'/edit')->withErrors($validator)-
```

---

---

```

19 >withInput();
20     } else {
21         $profile = Profile::findOrFail($id);
22         $profile->nama = Input::get('nama');
23         $profile->jeniskelamin = Input::get('jeniskelamin');
24         $profile->alamat = Input::get('alamat');
25         $profile->save();
26
27         Session::flash('message', 'Data Berhasil Diubah');
28         return Redirect::to('ivud');
29     }
30 }
31
32

```

---

Kemudian buat file bernama **edit.blade.php** didalam folder **app/views/ivud** dan isikan dengan code dibawah ini.

---

```

1  @extends('layouts.master')
2  @section('content')
3      <div class="row">
4          <div class="col-lg-12">
5              <h1 class="page-header">
6                  Halaman
7                  <small>Ubah Profile</small>
8              </h1>
9              {{ Form::model($profilesbyid, array('route' =>
10 array('ivud.update', $profilesbyid->id), 'method' => 'PUT')) }}
11              <div class="form-group">
12                  {{ Form::label('nama', 'Nama') }}
13                  {{ Form::text('nama', null, array('class' => 'form-
14 control', 'placeholder' => 'masukkan nama')) }}
15                  {{ '<div>'. $errors->first('nama'). '</div>' }}
16              </div>
17              <div class="form-group">
18                  {{ Form::label('jeniskelamin', 'Jenis Kelamin') }}
19                  {{ Form::select('jeniskelamin', array('L' => 'Laki -
20 Laki', 'P' => 'Perempuan'), null, array('class' => 'form-
21 control', 'placeholder' => 'Pili Jenis Kelamin')) }}
22                  {{ '<div>'. $errors->first('jeniskelamin'). '</div>' }}
23              </div>
24              <div class="form-group">
25                  {{ Form::label('alamat', 'Alamat') }}
26                  {{ Form::textarea('alamat', null, array('class' => 'form-
27 control', 'placeholder' => 'masukkan alamat')) }}
28                  </div>
29                  {{ Form::submit('SIMPAN', array('class' => 'form-control')) }}
30
31              {{ Form::close() }}
32

```

---

---

```
30
31     < /div>
32 < /div>
33 @stop
34 // agar program berjalan lancar hapus spasi setelah tanda <
35
```

---

Untuk coba menjalankannya silahkan ketik url seperti ini.

---

```
1 http://localhost/projectlaravel/public/ivud/1/edit
```

---

atau bisa klik tombol ubah maka hasilnya seperti gambar dibawah ini.



Terakhir yaitu membuat code untuk delete data, silahkan code function **destroy** dalam file **IvudController.php** diganti menjadi seperti dibawah ini.

---

```
1 public function destroy($id)
2 {
3     $profile = Profile::findOrFail($id);
4     $profile->delete();
5     Session::flash('message', 'Data Berhasil Dihapus');
6     return Redirect::to('crud');
7 }
```

---

Untuk tutorial membuat CRUD dengan **Eloquent ORM dan RESTful Resource Controllers** sampai sini sudah selesai untuk menjalankan delete data bisa klik tombol hapus. Silahkan dipilih mana yang lebih baik menggunakan Query Builder atau Menggunakan Eloquent ORM semua tergantung pada masing – masing individu.

Jika anda mengikuti tutorial ini dari awal jangan lupa menambahkan code dibawah ini di folder **app/views/includes** lalu cari file **sidebar.blade.php**

---

```
1 < li>
    < a href="{ URL::to('ivud') }" > < i class="fa fa-fw fa-edit" > /i>
```

---



---

```
2 CRUD Eloquent ORM< /a>
3 < /li>
4
5 // agar program berjalan lancar hapus spasi setelah tanda <
```

---

## BAB VIII

### 8. Memanfaatkan Auth Bawaan Laravel Untuk Membuat Authentication User Sederhana

Authentication yang akan dibahas disini yaitu auth basic atau dasar bukan berbasis ACL (Access Control List) maupun RBAC (Role Based Access Control). Di Framework laravel sudah disediakan komponen atau fungsi - fungsi yang kita perlukan untuk membuat authentication. Sebelum membahas lebih jauh tentang auth laravel, kita harus paham apa itu auth. Auth dalam aplikasi adalah digunakan untuk membatasi user tertentu dalam mengakses aplikasi atau program bahkan menu tertentu. Dalam membuat auth kita biasanya memerlukan form login yang berisi inputan username atau email dan password. Sebelum belajar membuat auth akan saya perkenalan dahulu fungsi - fungsi auth yang ada di laravel.

#### Fungsi Dasar Auth

##### 1. Password

Di laravel secara default password disimpan dengan metode hash menggunakan Bcrypt.

Berikut ini adalah code untuk membuat password menggunakan metode Hash

---

```
1 $password = Hash::make('secret');
```

---

Untuk mengecek password sama atau tidak anda bisa menggunakan code dibawah ini.

---

```
1 if (Hash::check('secret', isikanpasswordyangtelahdihash))
2 {
3 }
4
```

---

Kalau code dibawah ini cek password rehashed.

```
1 if (Hash::needsRehash($hashed))
2 {
3     $hashed = Hash::make('secret');
4 }
5 }
6
```

## 2. Authenticating Users

Untuk masuk dalam aplikasi anda bisa menggunakan metode atau code dibawah ini.

```
1 if (Auth::attempt(array('email' => $email, 'password' => $password)))
2 {
3     return Redirect::intended('dashboard');
4 }
5 }
6
```

Sedangkan untuk mengecek apakah pengguna sudah benar – benar login bisa menggunakan code dibawah ini.

```
1 if (Auth::check())
2 {
3 }
4
```

Kalau kita lihat biasanya disebuah form login ada checkbox dengan atribut remember me atau ingat saya, untuk di laravel anda bisa menggunakannya seperti ini.

```
1 if (Auth::attempt(array('email' => $email, 'password' => $password), true))
2 {
3 }
4
```

Untuk mengecek apakah remember me atau ingat saya sudah bekerja apa belum ada bisa mencoba dengan menuliskan code dibawah ini.

```
1 if (Auth::viaRemember())
2 {
3 }
4
```

Pada suatu ketika anda membuat website, dalam website anda ada beberapa status user ada yang aktif, ada yang belum di konfirmasi, ada juga yang telah di banned karena melakukan suatu kesalahan. Ketika mereka yang dibanned ataupun belum dikonfirmasi tidak bisa login kita bisa menggunakan code dibawah ini.

```
1 if (Auth::attempt(array('email' => $email, 'password' => $password, 'active'  
2 => 1)))  
3 {  
4 }
```

Untuk mengakses data user yang telah login bisa menggunakan seperti ini.

```
1 $email = Auth::user()->email;
```

Atau untuk id bisa seperti ini

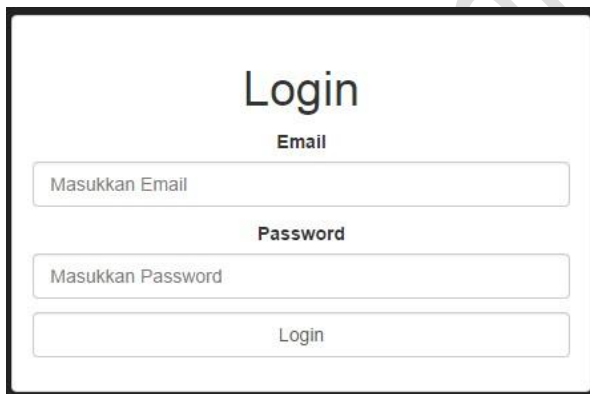
```
1 $id = Auth::id();
```

Sedangkan untuk logout anda bisa menggunakan code atau fungsi ini.

```
1 Auth::logout();
```

Itu adalah fungsi – fungsi yang bisa kita manfaatkan untuk membuat auth sederhana dan masih banyak fungsi yang belum saya bahas disini. Jadi silahkan dibaca baca di <http://laravel.com/docs/4.2/security>.

## Belajar Membuat Auth Sederhana



Ditutorial ini kita akan belajar membuat auth sederhana. Langkah awal yaitu konfigurasi file **database.php** didalam folder **app/config**. Buat database dan dengan tabel bernama users. Disini saya memanfaatkan migration yang ada di laravel dan juga sudah kita bahas ditutorial *sebelumnya*.

Jalankan perintah ini didalam command prompt / cmd.

```
1 php artisan migrate:make create_users_table
```

Untuk lebih jelasnya silahkan dilihat gambar dibawah ini.

```
Microsoft Windows [Version 6.3.9600]
(c) 2013 Microsoft Corporation. All rights reserved.

C:\Users\ant2526>cd \xampp\htdocs\projectlaravel

C:\xampp\htdocs\projectlaravel>php artisan migrate:make create_users_table
Created Migration: 2014_10_03_182159_create_users_table
Generating optimized class loader
C:\xampp\htdocs\projectlaravel>
```

**Cd \xampp\htdocs\projectlaravel**, digunakan untuk mengalihkan atau mengarahkan ke folder projectlaravel

**php artisan migrate:make create\_users\_table**, digunakan untuk membuat migration

Jika sukses maka akan dengan sendirinya akan dibuatkan file bernama **tanggal\_create\_users\_tabel.php** didalam folder **app/database/migrations**. Buka file tersebut dan isikan perintah atau code dibawah ini.

```
1  public function up()
2  {
3      Schema::create('users', function(Blueprint $table)
4      {
5          $table->increments('id');
6          $table->string('name', 50);
7          $table->string('username', 50);
8          $table->string('email', 300);
9          $table->string('password', 100);
10         $table->string('remember_token', 100)->nullable();
11         $table->timestamps();
12     });
13 }
14
15 public function down()
16 {
17     Schema::drop('users');
18 }
```

Jika selesai simpan dan di command prompt / cmd silahkan jalankan perintah ini.

```
1  php artisan migrate
```

Lebih jelasnya lihat gambar dibawah ini.

```
C:\xampp\htdocs\projectlaravel>php artisan migrate
Migrated: 2014_10_03_182159_create_users_table
C:\xampp\htdocs\projectlaravel>
```

Disini saya juga memanfaatkan seeder lavarel fungsinya untuk membuat / memasukan data dibuat untuk testing aplikasi atau web yang dibuat. Buat file bernama **UserTableSeeder.php** didalam folder **app/seeds** dan isikan code dibawah ini.

```
1 < ?php
2
3 class UserTableSeeder extends Seeder
4 {
5     public function run()
6     {
7         DB::table('users')->delete();
8         User::create(array(
9             'name' => 'Andri Riantana',
10            'username' => 'seputarpemrograman',
11            'email' => 'seputarpemrograman@gmail.com',
12            'password' => Hash::make('qwerty'),
13        ));
14    }
15 }
16
17 // Hilangkan spasi setelah tanda <
18
19
```


Kemudian buka file **DatabaseSeeder.php** yang berada di **app/seeds** ubah codenya menjadi seperti ini.

```
1 public function run()
2 {
3     Eloquent::unguard();
4     $this->call('UserTableSeeder');
5 }
6
```

Selesai silahkan bukan command prompt dan jalankan perintah berikut ini.

```
1 php artisan db:seed
```

Berikut ini saya sertakan gambarnya.



```
C:\xampp\htdocs\projectlaravel>php artisan db:seed
Seeded: UserTableSeeder
C:\xampp\htdocs\projectlaravel>
```

Jika berhasil sampai langkah ini maka didatabase anda sudah terdapat tabel bernama **users** dan beserta isinya.

Jika anda masih bingung membuat tabel dengan cara diatas anda bisa membuat secara manual dengan struktur tabel seperti dibawah ini.



Berikut juga saya sertakan sql / query nya

```
1 CREATE TABLE IF NOT EXISTS `users` (  
2  
3   `id` int(10) unsigned NOT NULL AUTO_INCREMENT,  
4  
5   `name` varchar(50) COLLATE utf8_unicode_ci NOT NULL,  
6  
7   `username` varchar(50) COLLATE utf8_unicode_ci NOT NULL,  
8  
9   `email` varchar(300) COLLATE utf8_unicode_ci NOT NULL,  
10  
11  `password` varchar(100) COLLATE utf8_unicode_ci NOT NULL,  
12  
13  `remember_token` varchar(100) COLLATE utf8_unicode_ci DEFAULT NULL,  
14  
15  `created_at` timestamp NOT NULL DEFAULT '0000-00-00 00:00:00',  
16  
17  `updated_at` timestamp NOT NULL DEFAULT '0000-00-00 00:00:00',  
18  
19  PRIMARY KEY (`id`)  
20 ) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci  
21 AUTO_INCREMENT=2 ;  
  
1 INSERT INTO `users` (`id`, `name`, `username`, `email`, `password`,  
2   `remember_token`, `created_at`, `updated_at`) VALUES  
3  
4   (1, 'Andri Riantana', 'seputarpemrograman',  
5     'seputarpemrograman@gmail.com',  
6     '$2y$10$5XyoNvbr2mS5I9yCxoPzSODirKC44HgT7xGcYJYCA5D3YT3KXsREu',  
7     'Eq5sI4kijZlws93D5ItShaULr7tAyOeLxwloDALakiQiUQ2jFE16YL7EMYgq', '2014-10-  
8     03 11:34:32', '2014-10-03 11:56:17');
```

Selanjutnya buat file bernama **UserController.php** letakkan didalam folder **app/controllers** dan isikan code dibawah ini.

---

```

1 < ?php
2
3 class UserController extends \BaseController {
4
5     public function login()
6     {
7         return View::make('login');
8     }
9
10    public function doLogin()
11    {
12        $rules = array(
13            'email' => 'required|email',
14            'password' => 'required|alphaNum|min:5'
15        );
16        $validator = Validator::make(Input::all(), $rules);
17        if ($validator->fails()) {
18            return Redirect::to('login')
19                ->withErrors($validator)
20                ->withInput(Input::except('password'));
21        } else {
22            $userdata = array(
23                'email' => Input::get('email'),
24                'password' =>
25                    Input::get('password')
26            );
27            if (Auth::attempt($userdata)) {
28                return Redirect::to('/');
29            } else {
30                return Redirect::to('login');
31            }
32        }
33    }
34
35    public function logout()
36    {
37        Auth::logout();
38        return Redirect::to('login');
39    }
40
41 }
42
43 //Hilangkan spasi setelah tanda <

```

---

Buat file lagi bernama **login.blade.php** letakkan di folder **app/views** dan isikan code ini.

---

```

1 < !DOCTYPE html>
2 < html lang="en">
3 < head>

```

---

---

```

3 < title>Halaman Login< /title>
4 {{ HTML::style('assets/css/bootstrap.min.css') }}
5 {{ HTML::style('assets/css/sb-admin.css') }}
6 {{ HTML::style('assets/font-awesome-4.1.0/css/font-awesome.min.css') }}
7 < /head>
8 < body>
9     < div class="row">
10         < div class="col-lg-4 text-center">
11             < /div>
12         < div class="col-lg-4 text-center">
13             < div class="panel panel-default">
14                 < div class="panel-body">
15                     {{ Form::open(array('url' => 'login')) }}
16                     < h1>Login< /h1>
17                     < p>
18                         {{ $errors->first('email') }}
19                         {{ $errors->first('password') }}
20                     < /p>
21                     < p>
22                         {{ Form::label('email', 'Email') }}
23                         {{ Form::text('email',
24                             Input::old('email'), array('class' => 'form-
25                             control', 'placeholder' => 'Masukkan Email')) }}
26                         < /p>
27                         < p>
28                             {{ Form::label('password', 'Password')
29                             }}
30                             {{ Form::password('password',
31                             array('class' => 'form-control', 'placeholder' => 'Masukkan Password')) }}
32                             < /p>
33                             < p>{{ Form::submit('Login', array('class' => 'form-
34                             control')) }}< /p>
35                             {{ Form::close() }}
36                         < /div>
37                     < /div>
38                 < /div>
39                 < div class="col-lg-4 text-center">
40                     < /div>
41 < /div>
42 < /body>
43 < /html>
44
45 // Hilangkan spasi setelah tanda <
46
47

```

---

Selanjutnya buka file **routes.php** yang berada dalam folder **app**. Dan tambahkan code dibawah ini.

---

```

1 Route::group(array('before' => 'auth'), function()
2 {
3     Route::get('logout', array('uses' => 'UserController@logout'));
4
5     // Route yang ingin diproteksi taruh disini

```

---



---

```
5 });  
6  
7 Route::get('login', array('uses' => 'UserController@login'));  
8 Route::post('login', array('uses' => 'UserController@doLogin'));  
9
```

---

Jika anda mengikuti tutorial ini dari awal anda bisa tuliskan code seperti ini.

---

```
1 Route::group(array('before' => 'auth'), function()  
2 {  
3 Route::get('/', 'ProfileController@index');  
4 Route::get('profile/{nama}', 'ProfileController@profile');  
5 Route::get('profile/{id}/{nama}', 'ProfileController@profileview');  
6 Route::get('profile/{nama?}', function($nama = null)  
7 {  
8     return $nama;  
9 });  
10 Route::get('profile/{nama?}', function($nama = 'Andri')  
11 {  
12     return $nama;  
13 });  
14 Route::get('formsederhana', 'ProfileController@formsederhana');  
15 Route::post('formsederhana', 'ProfileController@postformsederhana');  
16 Route::get('crud', 'CrudController@index');  
17 Route::get('crud/create', 'CrudController@create');  
18 Route::post('crud/create', 'CrudController@store');  
19 Route::get('crud/edit/{id}', 'CrudController@edit');  
20 Route::post('crud/update/{id}', array('as' => 'crud.update', 'uses' =>  
21 'CrudController@update'));  
22 Route::get('crud/destroy/{id}', 'CrudController@destroy');  
23 Route::resource('ivud', 'IvudController');  
24 Route::get('logout', array('uses' => 'UserController@logout'));  
25 });  
26  
27 Route::get('login', array('uses' => 'UserController@login'));  
28 Route::post('login', array('uses' => 'UserController@doLogin'));  
29
```

---

Yang terakhir pastikan ada file bernama User.php didalam folder **app/models**. Sekarang coba jalankan dengan mengetikan url seperti ini.

---

```
1 http://localhost/projectlaravel/public/login  
2  
3 Untuk username / email : seputarpemrograman@gmail.com  
4 Password : qwerty
```

---

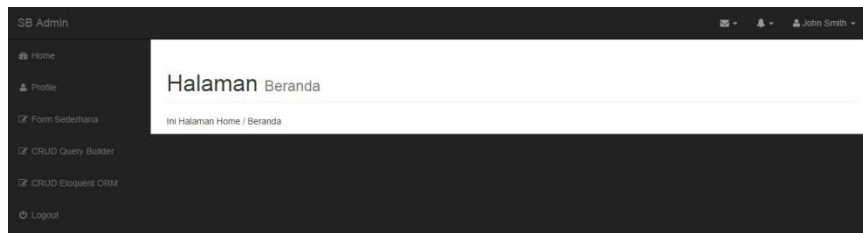
Oh ya jika anda mengikuti tutorial ini dari awal tambahkan code dibawah ini dalam file **sidebar.blade.php** dalam folder **app/views/includes**.

---

```
1 < li>
2     < a href="{{ URL::to('logout') }}">< i class="fa fa-fw fa-power-off"><
3 /i> Logout< /a>
4 < /li>
5 // Hilangkan spasi setelah tanda <
```

---

Jika anda sudah login akan menampilkan halaman seperti gambar dibawah ini.



Itulah cara membuat auth sederhana dalam framework laravel, mungkin anda akan sedikit bingung dengan tutorial diatas jika anda tidak mengikuti tutorial tentang laravel dari *awal*.

---

# BAB IX

## 9. Cara Menggunakan package sentry di framework laravel bagian install dan register

Sebelum kita membahas bagaimana cara install dan cara membuat register atau pendaftaran user menggunakan cartalyst sentry. Akan saya jelaskan apa itu package sentry, package sentry adalah sebuah package yang digunakan untuk mempermudah dalam manajemen user seperti pendaftaran, login, hak akses user menurut user atau grup user, aktivasi user, lupa password atau reset password. Jika anda biasanya untuk membuat itu secara manual akan lebih lama jadi jika anda memakai sentry akan lebih cepat dari biasanya karena sudah disediakan fungsi – fungsi seperti yang saya sebutkan diatas.

### Install Cartalyst Sentry

Pada kali ini kita menggunakan sentry pada framework laravel. Bagaimana cara install sentry di framework laravel. Sebelumnya siapkan project laravel yang siap digunakan jika anda belum punya silahkan **install framework laravel** lebih dahulu. Untuk tutorial ini saya menggunakan project yang telah dibuat sampai tutorial **CRUD Eloquent ORM dan RESTful Resource Controllers Laravel**. Tampilan Projectnya seperti gambar dibawah ini.



Tujuan menggunakan project diatas nantinya menu – menu yang sudah ada akan dibuat study kasus langsung ketika manajemen user nya. Jika project siap silahkan dibuka file **composer.json** lalu tambahkan perintah berikut ini.

```
1 "cartalyst/sentry": "2.1.*"
```

Maka di composer.json akan menjadi seperti ini.

```
1 ....
2 "require": {
3     "laravel/framework": "4.1.*",
4     "cartalyst/sentry": "2.1.*"
5 },
6 .....
```

Lalu jalankan perintah

---

```
1 composer update
```

---

Lebih jelasnya seperti gambar dibawah ini.

```
C:\xampp\htdocs\projectlaravel>composer update
Loading composer repositories with package information
Updating dependencies (including require-dev)
```

Jangan lupa arahkan commad prompt / cmd ke folder project anda caranya ketik

**cd \xampp\htdocs\namafolderproject**, \xampp\htdocs\ tergantung dimana anda meletakkan project yang telah dibuat.

Jika install selesai maka bisa dicek di folder vendor ada folder bernama **cartalyst** dan didalamnya ada folder bernama **sentry**. Selanjutnya setting **app.php** didalam folder **app/config**. Didalam array **\$providers** tambahkan baris code berikut ini.

---

```
1 'Cartalyst\Sentry\SentryServiceProvider',
```

---

Dan didalam array **\$aliases** tambahkan baris code seperti dibawah ini.

---

```
1 'Sentry' => 'Cartalyst\Sentry\Facades\Laravel\Sentry',
```

---

Kemudian setting **database.php** didalam folder **app/config** isi sesuai dengan konfigurasi database anda. Untuk konfigurasi saya seperti ini.

---

```
1 'mysql' => array(
2     'driver' => 'mysql',
3     'host' => 'localhost',
4     'database' => 'sentry',
5     'username' => 'root',
6     'password' => '',
7     'charset' => 'utf8',
8     'collation' => 'utf8_unicode_ci',
9     'prefix' => '',
10 ),
```

---

Jika selesai silahkan jalankan perintah dibawah ini di command prompt. Jika anda belum pernah belajar tentang php artisan migrate atau migration pada laravel anda bisa melihat tutorial **migration dan schema builder**.

---

```
1 php artisan migrate --package=cartalyst/sentry
```

---

Jika anda juga menggunakan project yang sama seperti tutorial ini jalankan juga peritah dibawah ini.

## 1 php artisan migrate

Jangan lupa arahkan path command prompt kedalam project anda.

```
C:\xampp\htdocs\projectlaravel>php artisan migrate --package=cartalyst/sentry
Migration table created successfully.
Migrated: 2012_12_06_225921_migration_cartalyst_sentry_install_users
Migrated: 2012_12_06_225929_migration_cartalyst_sentry_install_groups
Migrated: 2012_12_06_225945_migration_cartalyst_sentry_install_users_groups_pivot
Migrated: 2012_12_06_225988_migration_cartalyst_sentry_install_throttle
C:\xampp\htdocs\projectlaravel>
```

Sekarang silahkan cek didatabase anda pasti akan otomatis dibuatkan struktur tabel seperti gambar berikut ini.

<b>sentry.groups</b> <ul style="list-style-type: none"><li>id : int(10) unsigned</li><li>name : varchar(255)</li><li>permissions : text</li><li>created_at : timestamp</li><li>updated_at : timestamp</li></ul>	<b>sentry.users</b> <ul style="list-style-type: none"><li>id : int(10) unsigned</li><li>email : varchar(255)</li><li>password : varchar(255)</li><li>permissions : text</li><li>activated : tinyint(1)</li><li>activation_code : varchar(255)</li><li>activated_at : timestamp</li><li>last_login : timestamp</li><li>persist_code : varchar(255)</li><li>reset_password_code : varchar(255)</li><li>first_name : varchar(255)</li><li>last_name : varchar(255)</li><li>created_at : timestamp</li><li>updated_at : timestamp</li></ul>
<b>sentry.users_groups</b> <ul style="list-style-type: none"><li>user_id : int(10) unsigned</li><li>group_id : int(10) unsigned</li></ul>	
<b>sentry.migrations</b> <ul style="list-style-type: none"><li>migration : varchar(255)</li><li>batch : int(11)</li></ul>	
<b>sentry.throttle</b> <ul style="list-style-type: none"><li>id : int(10) unsigned</li><li>user_id : int(10) unsigned</li><li>ip_address : varchar(255)</li><li>attempts : int(11)</li><li>suspended : tinyint(1)</li><li>banned : tinyint(1)</li><li>last_attempt_at : timestamp</li><li>suspended_at : timestamp</li><li>banned_at : timestamp</li></ul>	<b>sentry.profiles</b> <ul style="list-style-type: none"><li>id : int(10) unsigned</li><li>nama : varchar(100)</li><li>jeniskelamin : enum('L','P')</li><li>alamat : varchar(255)</li><li>created_at : timestamp</li><li>updated_at : timestamp</li></ul>

Lalu jalankan juga perintah berikut ini tentunya melalui command prompt / cmd.

## 1 php artisan config:publish cartalyst/sentry

maka otomatis dibuatkan file untuk konfigurasi sentry, untuk cek silahkan cari file bernama **config.php** didalam folder **app/config/cartalyst/sentry**.

## Membuat Form Register / Pendaftaran dengan Cartalyst Sentry.

Kalau kita lihat di website resmi sentry setelah install sentry kita akan diajarkan authentication and login tapi ditutorial ini saya balik kita pelajari membuat register atau pendaftaran user terlebih dahulu. Jika anda ingin belajar langsung authentication and login anda bisa melihat tutorial bagian kedua setelah tutorial ini. Kembali pada pendaftaran user, untuk membuat user baru / mendaftarkan user baru bisa menggunakan fungsi **Sentry::register()** Pertama tama buatlah file bernama **UserController.php** didalam folder **Controllers**. Dan isikan denga code dibawah ini.

---

```
1 < ?php
2 class UserController extends \BaseController {
3     public function register()
4     {
5
6     public function doRegister()
7     {
8     }
9 }
10 // Hilangkan spasi setelah tanda <
11
12
```

---

Untuk fungsi atau function **register** ubah menjadi seperti ini.

---

```
1 public function register()
2 {
3     return View::make('sentry.register');
4 }
```

---

Untuk fungsi atau function **doRegister** ubah menjadi seperti dibawah ini.

---

```
1 public function doRegister()
2 {
3
4     $rules = array(
5         'first_name' => 'required',
6         'last_name' => 'required',
7         'email' => 'required|email|unique:users',
8         'password' => 'required|between:4,10|confirmed',
9         'password_confirmation' => 'between:4,10'
10    );
11    $validator = Validator::make(Input::all(), $rules);
```

---

---

```

12
13     if ($validator->fails()) {
14         return Redirect::to('register')
15             ->withErrors($validator);
16     } else {
17         try
18         {
19             $user = Sentry::register(array(
20                 'first_name' => Input::get( 'first_name' ),
21                 'last_name' => Input::get( 'last_name' ),
22                 'email' => Input::get( 'email' ),
23                 'password' => Hash::make(Input::get( 'password' )),
24                 'activated' => true,
25             ));
26             $activationCode = $user->getActivationCode();
27         }
28         catch (Cartalyst\Sentry\Users>LoginRequiredException $e)
29         {
30             Session::flash('message', 'Login field is required. ');
31             return Redirect::to('register');
32         }
33         catch (Cartalyst\Sentry\Users>PasswordRequiredException $e)
34         {
35             Session::flash('message', 'Password field is required. ');
36             return Redirect::to('register');
37         }
38         catch (Cartalyst\Sentry\Users>UserExistsException $e)
39         {
40             Session::flash('message', 'User with this login already
41 exists. ');
42             return Redirect::to('register');
43         }
44     }
45     Session::flash('message', 'Successfully Register, Please Login');
46     return Redirect::to('register');
47 }

```

---

Jika selesai silahkan dibuat folder bernama **sentry** pada folder **app/views**. Dan didalam folder sentry buat file bernama **register.blade.php** lalu isikan dengan code dibawah ini.

---

```

1 < !DOCTYPE html>
2 < html lang="en">
3 < head>
4 < title>Halaman Pendaftaran< /title>
5 {{ HTML::style('assets/css/bootstrap.min.css') }}
6 {{ HTML::style('assets/css/sb-admin.css') }}
7 {{ HTML::style('assets/font-awesome-4.1.0/css/font-awesome.min.css') }}
8 < /head>
9 < body>

```

---

```

9 < div class="row">
10     < div class="col-lg-4 text-center">
11
12     < /div>
13     < div class="col-lg-4 text-center">
14         < div class="panel panel-default">
15             < div class="panel-body">
16                 {{ Form::open(array('url' => 'register')) }}
17                 < h1>Register< /h1>
18                 @if ( Session::get('message') )
19                     < div>{{{ Session::get('message') }}}<
20 /div>
21                 @endif
22                 < p>
23                     {{ Form::label('first_name', 'Fisrt Name') }}
24                     {{ Form::text('first_name',
25 Input::old('first_name'), array('class' => 'form-
26 control','placeholder'=>'Fisrt Name')) }}
27                     {{ $errors->first('first_name') }}
28                 < /p>
29
30                 < p>
31                     {{ Form::label('last_name', 'Last Name') }}
32                     {{ Form::text('last_name',
33 Input::old('last_name'), array('class' => 'form-
34 control','placeholder'=>'Last Name')) }}
35                     {{ $errors->first('last_name') }}
36                 < /p>
37
38                 < p>
39                     {{ Form::label('email', 'Email') }}
40                     {{ Form::text('email', Input::old('email'),
41 array('class' => 'form-control','placeholder'=>'Email')) }}
42                     {{ $errors->first('email') }}
43                 < /p>
44
45                 < p>
46                     {{ Form::label('password', 'Password') }}
47                     {{ Form::password('password', array('class' =>
48 'form-control','placeholder'=>'Password')) }}
49                     {{ $errors->first('password') }}
50                 < /p>
51
52                 < p>
53                     {{ Form::label('password_confirmation',
54 'Konfirmasi Password') }}
55                     {{ Form::password('password_confirmation',
56 array('class' => 'form-control','placeholder'=>'Masukkan Konfirmasi
57 Password')) }}
58                     {{ $errors->first('password_confirmation') }}
59                 < /p>
60
61                 < p>{{ Form::submit('Register', array('class' =>
62 'form-control')) }}< /p>
63                 {{ Form::close() }}
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99

```



---

```

57         < /div>
58     < /div>
59 < /div>
60 < div class="col-lg-4 text-center">
61
62     < /div>
63
64 < /body>
65 < /html>
66
67 // Hilangkan spasi setelah tanda <

```

---

Selanjutnya didalam file **routes.php** dalam folder **app** silahkan ditambahkan code dibawah ini.

---

```

1 Route::get('register','UserController@register');
2 Route::post('register','UserController@doRegister');

```

---

Sekarang coba jalankan dengan url seperti ini.

---

```

1 http://localhost/projectlaravel/public/register

```

---

maka akan muncul form sebagai berikut.

The image shows a web form titled "Register". It has the following fields and labels:

- Fisrt Name**: Input field with label "Fisrt Name" above it.
- Last Name**: Input field with label "Last Name" above it.
- Email**: Input field with label "Email" above it.
- Password**: Input field with label "Password" above it.
- Konfirmasi Password**: Input field with label "Masukkan Konfirmasi Password" above it.
- Register**: A button at the bottom of the form.

Ketika kita isi semua inputan lalu klik button register maka data akan disimpan didatabase ditabel users. Sampai sini kita sudah bisa isntall dan membuat register atau pendaftaran menggunakan package sentry dilaravel.

# BAB X

## 10. Cara Menggunakan package sentry di framework laravel bagian CRUD Group

Setelah sebelumnya seputar pemrograman membahas cara install sentry dan membuat pendaftaran user sederhana menggunakan package sentry. Ditutorial kali ini kita akan membahas **groups** pada sentry. Groups sentry digunakan untuk manajemen user per group, contohnya group admin, group moderator, group member dan bisa kita tambah atau kurangi sesuai kebutuhan aplikasi kita. Disini kita akan membahas bagaimana cara kerja **create** ( membuat ) **group** dengan beberapa permission, **update** ( ubah ) **group** dan **delete** ( hapus ) **group**.

*Catatan : Groups Permissions, 0 = deny & 1 = allow.*

Tanpa basa basi langsung kita praktekkan membuat group dengan sentry. Pertama buatlah file bernama **GroupController.php** letakkan dalam folder **app/controllers**. Dan isi dengan code dibawah ini.

---

```
1 < ?php
2 class GroupController extends BaseController {
3
4 }
5 // agar program berjalan lancar hapus spasi setelah tanda <
```

---

Jika selesai sekarang buatlah sebuah folder bernama **sentry** didalam folder **app/views**. Lalu buat folder lagi didalam folder sentry yang baru dibuat dengan nama **group**. Didalam folder group ini buatlah beberapa file bernama **index.blade.php** , **create.blade.php** dan **edit.blade.php**. Setelah selesai membuat file – file ini buka file **index.blade.php** dan isikan code dibawah ini.

---

```
1 @extends('layouts.master')
2 @section('content')
3 < div class="row">
4     < div class="col-lg-12">
5         < h1 class="page-header">
6             Halaman
7             < small>Daftar Group< /small>
8         < /h1>
9         @if (Session::has('message'))
10             {{ Session::get('message') }}
11         @endif
12         < p>< a href="{{ URL::to('group/create') }}" class="btn btn-
13 primary" role="button">Tambah Group Baru< /a>< /p>
14     < div class="table-responsive">
```

---

---

```

12         < table class="table table-bordered table-
13 hover">
14             < thead>
15                 < tr>
16                     < th width="10">ID< /th>
17                     < th>Group< /th>
18                     < th width="146">Aksi< /th>
19                 < /tr>
20             < /thead>
21             < tbody>
22                 @foreach($group as $value)
23                     < tr>
24                         < td>{{{ $value->id }}}< /td>
25                         < td>{{{ $value->name }}}< /td>
26                         < td>
27                             {{ Form::open(array('url' =>
28 'group/' . $value->id)) }}
29                             < div class="btn-group">
30                                 < a href="{{ URL::to('group/'
31 . $value->id . '/edit') }}" class="btn btn-primary">Ubah< /a>
32                                 {{ Form::hidden('_method',
33 'DELETE') }}
34                                 {{Form::button('Hapus',
35 array('type' => 'submit', 'class' => 'btn btn-primary'))}}
36                                 < /div>
37                                 {{ Form::close() }}
38                             < /td>
39                         < /tr>
40                     @endforeach
41                 < /tbody>
42             < /table>
43         < /div>
44         {{{ $group->links() }}}
45     < /div>
46 @stop
47 // agar program berjalan lancar hapus spasi setelah tanda <

```

---

Jangan lupa disimpan terlebih dahulu, lalu buka file **GroupController.php** dan didalam class **GroupController** sisipkan code dibawah ini.

---

```

1 public function index()
2 {
3     $group= Group::paginate(5);
4     $group=
5     [
6         'group' => $group
7     ];
8     return View::make('sentry.group.index', $group);
9 }

```

---

---

8  
9

---

Hasilnya seperti ini.

---

```
1 < ?php
2
3 class GroupController extends \BaseController {
4
5     public function index()
6     {
7         $group = Group::paginate(5);
8         $group =
9             [
10                 'group' => $group
11             ];
12         return View::make('sentry.group.index', $group);
13     }
14 }
15
16 // agar program berjalan lancar hapus spasi setelah tanda <
```

---

Buka file **routes.php** didalam folder **app** dan tambahkan code ini.

---

```
1 Route::resource('group', 'GroupController');
```

---

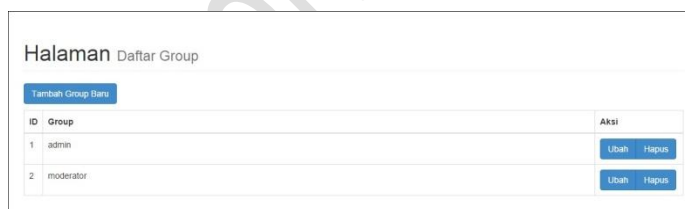
Sampai sini list atau daftar group telah selesai anda bisa mencoba dengan mengakses url seperti ini.

---

```
1 http://localhost/projectlaravel/public/group
```

---

Jika sudah ada isinya didalam database makan kurang lebih hasilnya seperti gambar dibawah ini.



ID	Group	Aksi
1	admin	Ubah Hapus
2	moderator	Ubah Hapus

Kita lanjutkan dengan membuat create (tambah) group, buka kembali controller **GroupController.php** dan tambah kan code dibawah ini persis dibawah function index.

---

```
1 public function create()
2 {
3     return View::make('sentry.group.create');
4 }
```

---

---

```

5
6 public function store()
7 {
8     $rules = array(
9         'group' => 'required',
10    );
11
12    $validator = Validator::make(Input::all(), $rules);
13
14    if ($validator->fails()) {
15        return Redirect::to('group/create')->withErrors($validator)-
16>withInput();
17    } else {
18        try
19        {
20            $group = Sentry::createGroup(array(
21                'name' => Input::get('group'),
22                'permissions' => Input::get('cb'),
23            ));
24        }
25        catch (Cartalyst\Sentry\Groups\NameRequiredException $e)
26        {
27            Session::flash('message', 'Name field is required');
28            return Redirect::to('group');
29        }
30        catch (Cartalyst\Sentry\Groups\GroupExistsException $e)
31        {
32            Session::flash('message', 'Group already exists');
33            return Redirect::to('group');
34        }
35
36        Session::flash('message', 'Data Berhasil Ditambahkan');
37        return Redirect::to('group');
38    }
39 }

```

---

Lalu buka file **create.blade.php** didalam folder **group** yang telah dibuat tadi dan isikan code dibawah ini.

---

```

1 @extends('layouts.master')
2 @section('content')
3 < div class="row">
4     < div class="col-lg-12">
5         < h1 class="page-header">
6             Halaman
7             < small>Tambah Group< /small>
8         < /h1>
9
10        {{ Form::open(array('url' => 'group')) }}

```

---

```

11         {{ Form::label('group', 'Group') }}
12         {{ Form::text('group', null, array('class' => 'form-
13 control','placeholder'=>'masukkan group')) }}
14         {{ '< div>'. $errors->first('group'). '< /div>' }}
15     < /div>
16
17     < div class="form-group">
18         {{ Form::label('user', 'User') }} :
19         < label class="checkbox-inline">
20             {{Form::checkbox('cb[user.read]', '1')}} Read
21         < /label>
22         < label class="checkbox-inline">
23             {{Form::checkbox('cb[user.create]', '1')}} Create
24         < /label>
25         < label class="checkbox-inline">
26             {{Form::checkbox('cb[user.update]', '1')}} Update
27         < /label>
28         < label class="checkbox-inline">
29             {{Form::checkbox('cb[user.destroy]', '1')}} Destroy
30         < /label>
31     < /div>
32
33     < div class="form-group">
34         {{ Form::label('crud', 'CRUD Query Builder') }} :
35         < label class="checkbox-inline">
36             {{Form::checkbox('cb[cqb.read]', '1')}} Read
37         < /label>
38         < label class="checkbox-inline">
39             {{Form::checkbox('cb[cqb.create]', '1')}} Create
40         < /label>
41         < label class="checkbox-inline">
42             {{Form::checkbox('cb[cqb.update]', '1')}} Update
43         < /label>
44         < label class="checkbox-inline">
45             {{Form::checkbox('cb[cqb.destroy]', '1')}} Destroy
46         < /label>
47     < /div>
48
49     < div class="form-group">
50         {{ Form::label('crud', 'CRUD Eloquent ORM') }} :
51         < label class="checkbox-inline">
52             {{Form::checkbox('cb[ceo.read]', '1')}} Read
53         < /label>
54         < label class="checkbox-inline">
55             {{Form::checkbox('cb[ceo.create]', '1')}} Create
56         < /label>
57         < label class="checkbox-inline">
58             {{Form::checkbox('cb[ceo.update]', '1')}} Update
59         < /label>
60         < label class="checkbox-inline">
61             {{Form::checkbox('cb[ceo.destroy]', '1')}} Destroy
62         < /label>
63     < /div>
64
65     {{ Form::submit('SIMPAN', array('class' => 'form-control')) }}
66
67     {{ Form::close() }}
68

```

---

```
59     < /div>
60 < /div>
61 @stop
62 // agar program berjalan lancar hapus spasi setelah tanda <
```

---

Untuk menjalankannya anda bisa klik tombol tambah group ataupun mengakses dengan url seperti ini.

---

1 <http://localhost/projectlaravel/public/group/create>

---

Berikut ini gambar dari form create group sentry.

Halaman Tambah Group



*Catatan : untuk checkbox permodul memang masih saya buat secara manual, jadi bisa anda kembangkan misal membuat tabel lagi modul dan aksi. Tapi ini sudah lebih dari cukup dimanis karena kita bisa mengatur permodul dan fungsinya.*

Selanjutnya adalah membuat update (ubah) group, buka kembali controller **GroupController.php** dan tambahkan code dibawah ini tepat dibawah function **store**.

---

```
1 public function edit($id)
2 {
3     $groupbyid = Group::findOrFail($id);
4     $groupbyid =
5     [
6         'groupbyid' => $groupbyid
7     ];
8     return View::make('sentry.group.edit', $groupbyid);
9 }
10 public function update($id)
11 {
12     $rules = array(
13         'group' => 'required',
14     );
15     $validator = Validator::make(Input::all(), $rules);
16     if ($validator->fails()) {
17         return Redirect::to('group/' . $id . '/edit')->withErrors($validator)-
18 >withInput();
19     } else {
20         try
```

---

---

```

21     {
22     $group= Sentry::findGroupById($id);
23     $arrexes = array();
24     foreach ($group->permissions as $key=> $value) {
25         if (array_key_exists($key, Input::get('cb'))) {
26             $arrexes[$key] = '1';
27         }
28     }
29     $arrexes2= array();
30     foreach ($group->permissions as $key=> $value) {
31         if (!array_key_exists($key, $arrexes)) {
32             $arrexes2[$key] = '0';
33         }
34     }
35     $arrexes3= array_merge($arrexes2,Input::get('cb'));
36     $group->name = Input::get('group');
37     $group->permissions = $arrexes3;
38     $group->permissions = array();
39
40     if ($group->save())
41     {
42         Session::flash('message', 'Data Berhasil Diubah');
43         return Redirect::to('group');
44     }
45     else
46     {
47         Session::flash('message', 'Data Gagal Diubah');
48         return Redirect::to('group');
49     }
50 }
51 catch (Cartalyst\Sentry\Groups\NameRequiredException $e)
52 {
53     Session::flash('message', 'Name field is required');
54     return Redirect::to('group');
55 }
56 catch (Cartalyst\Sentry\Groups\GroupExistsException $e)
57 {
58     Session::flash('message', 'Group already exists');
59     return Redirect::to('group');
60 }
61 catch (Cartalyst\Sentry\Groups\GroupNotFoundException $e)
62 {
63     Session::flash('message', 'Group was not found. ');
64     return Redirect::to('group');
65 }
66 }
67 }

```

---

Sekarang buka file **edit.blade.php** lalu isikan code seperti dibawah ini.

---

```

1  @extends('layouts.master')
2  @section('content')
    < div class="row">

```

---



```

3      < div class="col-lg-12">
4          < h1 class="page-header">
5              Halaman
6          < small>Ubah Group< /small>
7      < /h1>
8      {{!--*/ $perm = json_decode($groupbyid->permissions, true) /*--}}
9
10     {{ Form::model($groupbyid, array('route' => array('group.update',
11 $groupbyid->id), 'method' => 'PUT')) }}
12
13     < div class="form-group">
14         {{ Form::label('group', 'Group') }}
15         {{ Form::text('group', $groupbyid->name, array('class' =>
16 'form-control', 'placeholder' => 'masukkan group')) }}
17         {{ '< div>'. $errors->first('group'). '< /div>' }}
18     < /div>
19
20     < div class="form-group">
21         {{ Form::label('user', 'User') }} :
22         < label class="checkbox-inline">
23             {{Form::checkbox('cb[user.read]', '1',
24 (!empty($perm['user.read']) == 1 ? true : false))}} Read
25         < /label>
26         < label class="checkbox-inline">
27             {{Form::checkbox('cb[user.create]', '1',
28 (!empty($perm['user.create']) == 1 ? true : false))}} Create
29         < /label>
30         < label class="checkbox-inline">
31             {{Form::checkbox('cb[user.update]', '1',
32 (!empty($perm['user.update']) == 1 ? true : false))}} Update
33         < /label>
34         < label class="checkbox-inline">
35             {{Form::checkbox('cb[user.destroy]', '1',
36 (!empty($perm['user.destroy']) == 1 ? true : false))}} Destroy
37         < /label>
38     < /div>
39
40     < div class="form-group">
41         {{ Form::label('crud', 'CRUD Query Builder') }} :
42         < label class="checkbox-inline">
43             {{Form::checkbox('cb[cqb.read]', '1',
44 (!empty($perm['cqb.read']) == 1 ? true : false))}} Read
45         < /label>
46         < label class="checkbox-inline">
47             {{Form::checkbox('cb[cqb.create]', '1',
48 (!empty($perm['cqb.create']) == 1 ? true : false))}} Create
49         < /label>
50         < label class="checkbox-inline">
51             {{Form::checkbox('cb[cqb.update]', '1',
52 (!empty($perm['cqb.update']) == 1 ? true : false))}} Update
53         < /label>
54         < label class="checkbox-inline">
55             {{Form::checkbox('cb[cqb.destroy]', '1',
56 (!empty($perm['cqb.destroy']) == 1 ? true : false))}} Destroy
57         < /label>
58     < /div>

```

---

```

51         < div class="form-group">
52             {{ Form::label('crud', 'CRUD Eloquent ORM') }} :
53             < label class="checkbox-inline">
54                 {{Form::checkbox('cb[ceo.read]', '1',
55 (!empty($perm['ceo.read']) == 1 ? true : false))}} Read
56             < /label>
57             < label class="checkbox-inline">
58                 {{Form::checkbox('cb[ceo.create]', '1',
59 (!empty($perm['ceo.create']) == 1 ? true : false))}} Create
60             < /label>
61             < label class="checkbox-inline">
62                 {{Form::checkbox('cb[ceo.update]', '1',
63 (!empty($perm['ceo.update']) == 1 ? true : false))}} Update
64             < /label>
65             < label class="checkbox-inline">
66                 {{Form::checkbox('cb[ceo.destroy]', '1',
67 (!empty($perm['ceo.destroy']) == 1 ? true : false))}} Destroy
68             < /label>
69         < /div>
70         {{ Form::submit('SIMPAN', array('class' => 'form-control')) }}
71     {{ Form::close() }}
72 < /div>
73 @stop
74
// agar program berjalan lancar hapus spasi setelah tanda <

```

---

Silahkan disimpan terlebih dahulu lalu untuk menjalankan bisa klik tombol ubah, maka hasilnya akan nampak seperti gambar ini.

---

Yang terakhir adalah code untuk hapus group, silahkan dibuka file **GroupController.php** dan tambahkan code dibawah ini dibawah function **update**.

---

```

1 public function destroy($id)
2 {
3     try
4     {
5         $group = Sentry::findGroupById($id);
6         $group->delete();
7     }
8     catch (Cartalyst\Sentry\Groups\GroupNotFoundException $e)
9     {
10         echo 'Group was not found.';
11     }
12 }

```

---

---

```
9      }
10
11      Session::flash('message', 'Data Berhasil Dihapus');
12      return Redirect::to('group');
13  }
14
```

---

Selesai jika anda coba klik atau menekan tombol hapus maka data group otomatis terhapus.

*Untuk keterangan memang tidak semua saya jelaskan jadi kalau ada pertanyaan silahkan tuliskan dikomentar dibawah ini, dan juga jangan lupa di share agar lebih bermanfaat lagi bagi yang lainnya.*

## BAB XI

### 11. Cara Menggunakan package sentry di framework laravel bagian CRUD User

Ditutorial bagian **grup package sentry** digunakan untuk memberikan permission atau hak akses setiap group dan user mempunyai satu grup per user. Bagian kali ini kita akan membahas mengenai create read update dan delete user menggunakan package sentry di framework laravel. Bagian create user nanti kita akan mengisikan pada tabel users dan user\_groups dimana tabel user\_groups inilah tempat digunakan untuk memilih group yang telah dbuat sebelumnya. Biar tidak bingung langsung saja kita praktekan. Buka project sebelumnya jika tidak mengikuti dari awal mungkin akan bingung, jadi saya anjurkan mengikuti dari awal sejak cara **install package sentry** dan dilanjutkan membuat **group sentry**.

Ditutorial sebelumnya kita sudah membuat controller user, bernama **UserController.php** lalu sekarang buat folder bernama **user** didalam folder **app/views/sentry**. Jika sudah selesai buatlah file – filer bernama **index.blade.php**, **create.blade.php** dan **edit.blade.php** didalam folder **user** yang barusan dibuat. Sekarang silahkan dibuka file bernama **UserController.php** didalam folder **app/controllers**. Tambahkan code / function berikut ini.

---

```
1 public function index()
2 {
3     $user = User::paginate(5);
4     $user =
5     [
6         'user' => $user
7     ]
8 }
```

---

---

```

6     ];
7     return View::make('sentry.user.index', $user);
8 }
9

```

---

Kemudian buka file **index.blade.php** didalam folder **app/views/sentry/user** dan isi dengan code dibawah ini.

---

```

1  @extends('layouts.master')
2  @section('content')
3      <div class="row">
4          <div class="col-lg-12">
5              <h1 class="page-header">
6                  Halaman
7                  <small>Daftar User</small>
8              </h1>
9              @if (Session::has('message'))
10                 {{ Session::get('message') }}
11             @endif
12             <p><a href="{{ URL::to('user/create') }}" class="btn btn-
13 primary" role="button">Tambah User Baru</a></p>
14             <div class="table-responsive">
15                 <table class="table table-bordered table-
16 hover">
17                     <thead>
18                         <tr>
19                             <th width="10">ID</th>
20                             <th>Nama</th>
21                             <th>Email</th>
22                             <th width="147">Aksi</th>
23                         </tr>
24                     </thead>
25                     <tbody>
26                         @foreach($user as $value)
27                             <tr>
28                                 <td>{{ $value->id }}</td>
29                                 <td>{{ $value->first_name.'
30 '.$value->last_name }}</td>
31                                 <td>{{ $value->email }}</td>
32                                 <td>
33                                     {{ Form::open(array('url' =>
34 'user/'.$value->id)) }}
35                                     <div class="btn-group">
36                                         <a href="{{ URL::to('user/' .
37 $value->id . '/edit') }}" class="btn btn-primary">Ubah</a>
38                                         {{ Form::hidden('_method',
39 'DELETE') }}
40                                         {{ Form::button('Hapus',
41 array('type' => 'submit', 'class' => 'btn btn-primary')) }}
42                                         </div>
43                                         {{ Form::close() }}
44                                     </td>
45                             </tr>
46                         @endforeach
47                     </tbody>

```

---

---

```

40             < /table>
41         < /div>
42         {{ $user->links() }}
43     < /div>
44 < /div>
45 @stop
46
47 // agar program berjalan lancar hapus spasi setelah tanda <
48
49

```

---

Simpan, lalu untuk selanjutnya buka file **routes.php** didalam folder **app**, tambahkan code dibawah ini.

---

```

1 Route::resource('user', 'UserController');

```

---

Sekarang kita sudah bisa mencoba menjalankan untuk melihat daftar user, dengan mengetikan url berikut ini.

---

```

1 http://localhost/projectlaravel/public/user

```

---

Hasilnya akan nampak seperti gambar dibawah ini.

ID	Nama	Email	Aksi
1	andi niantana	andi@gmail.com	Ubah Hapus
2	seputar pemrograman	moderator@gmail.com	Ubah Hapus

Untuk langkah berikutnya buka file **UserController.php** yang berada dalam folder **app/controllers**. Tambahkan function atau code dibawah ini, dibawah function **index**.

---

```

1 public function create()
2 {
3     $group = Group::lists('name', 'id');
4     $data =
5     [
6         'group' => $group
7     ];
8     return View::make('sentry.user.create', $data);
9 }
10 public function store()
11 {
12     $rules = array(
13         'nama_depan' => 'required',
14         'nama_belakang' => 'required',
15         'email' => 'required|email|unique:users',
16         'password' => 'required|between:4,11|confirmed',

```

---

---

```

16         'confirmation_password' => 'between:4,11',
17         'group' => 'required',
18     );
19     $validator = Validator::make(Input::all(), $rules);
20
21     if ($validator->fails()) {
22         return Redirect::to('user/create')->withErrors($validator)-
23 >withInput();
24     } else {
25
26         try
27         {
28             $user = Sentry::createUser(array(
29                 'first_name' => Input::get('nama_depan'),
30                 'last_name' => Input::get('nama_belakang'),
31                 'email' => Input::get('email'),
32                 'password' => Input::get('password'),
33                 'activated' => true,
34             ));
35
36             $groupbyid = Sentry::findGroupById(Input::get('group'));
37             $user->addGroup($groupbyid);
38
39             catch (Cartalyst\Sentry\Users>LoginRequiredException $e)
40             {
41                 Session::flash('message', 'Login field is required.');
```

Tutorial 4 GratisSS

```

42                 return Redirect::to('user');
```

Tutorial 4 GratisSS

```

43             }
44             catch (Cartalyst\Sentry\Users>PasswordRequiredException $e)
45             {
46                 Session::flash('message', 'Password field is required.');
```

Tutorial 4 GratisSS

```

47                 return Redirect::to('user');
```

Tutorial 4 GratisSS

```

48             }
49             catch (Cartalyst\Sentry\Users>UserExistsException $e)
50             {
51                 Session::flash('message', 'User with this login already
52 exists.');
```

Tutorial 4 GratisSS

```

53                 return Redirect::to('user');
```

Tutorial 4 GratisSS

```

54             }
55             catch (Cartalyst\Sentry\Groups\GroupNotFoundException $e)
56             {
57                 Session::flash('message', 'Group was not found.');
```

Tutorial 4 GratisSS

```

58                 return Redirect::to('user');
```

Tutorial 4 GratisSS

```

59             }
60         }
61         Session::flash('message', 'Data Berhasil Ditambahkan');
62         return Redirect::to('user');
```

Tutorial 4 GratisSS

```

63     }
64 }
```

---

Jangan lupa mengganti konfigurasi file **config.php** di folder **app/config/packages/cartalyst/sentry**. cari code seperti ini

---

```
1 'hasher' => 'native',
```

---

ganti menjadi seperti dibawah ini

---

```
1 'hasher' => 'sha256',
```

---

Kegunaannya agar password otomatis di enkripsi.

Lalu buka file **create.blade.php** dan isikan code ini.

---

```
1 @extends('layouts.master')
2 @section('content')
3 < div class="row">
4     < div class="col-lg-12">
5         < h1 class="page-header">
6             Halaman
7             < small>Tambah User< /small>
8         < /h1>
9         {{ Form::open(array('url' => 'user')) }}
10
11         < div class="form-group">
12             {{ Form::label('nama_depan', 'Nama Depan') }}
13             {{ Form::text('nama_depan', null, array('class' => 'form-
14 control','placeholder'=>'masukkan nama depan')) }}
15             {{ '< div>'. $errors->first('nama_depan'). '< /div>' }}
16         < /div>
17         < div class="form-group">
18             {{ Form::label('nama_belakang', 'Nama Belakang') }}
19             {{ Form::text('nama_belakang', null, array('class' =>
20 'form-control','placeholder'=>'masukkan nama belakang')) }}
21             {{ '< div>'. $errors->first('nama_belakang'). '< /div>' }}
22         < /div>
23         < div class="form-group">
24             {{ Form::label('email', 'Email') }}
25             {{ Form::text('email', null, array('class' => 'form-
26 control','placeholder'=>'masukkan email')) }}
27             {{ '< div>'. $errors->first('email'). '< /div>' }}
28         < /div>
29         < div class="form-group">
30             {{ Form::label('password', 'Password') }}
31             {{ Form::password('password', array('class' => 'form-
32 control','placeholder'=>'masukkan password')) }}
33             {{ '< div>'. $errors->first('password'). '< /div>' }}
34         < /div>
35         < div class="form-group">
36             {{ Form::label('password_confirmation', 'Password
37 Confirmation') }}
38             {{ Form::password('password_confirmation', array('class'
39 => 'form-control','placeholder'=>'masukkan password confirmation')) }}
40             {{ '< div>'. $errors->first('password_confirmation'). '<
41 /div>' }}
42         < /div>
43     < /div>
44 < /div>
```

---

---

```

38         {{ Form::label('group', 'Group') }}
39         {{ Form::select('group', $group, null, array('class' =>
40 'form-control', 'placeholder' => 'Select Group...')); }}
41         {{ '< div>'. $errors->first('group'). '< /div>' }}
42     < /div>
43     {{ Form::submit('SIMPAN', array('class' => 'form-control')) }}
44
45     {{ Form::close() }}
46
47 < /div>
48 @stop
49 // agar program berjalan lancar hapus spasi setelah tanda <
50

```

---

Sekarang buat model, buka folder **app/models** dan buat file bernama **Group.php**, isikan dengan code berikut ini.

---

```

1 < ?php class Group extends Eloquent {
2
3 }
4
5 // agar program berjalan lancar hapus spasi setelah tanda <

```

---

Simpan, untuk menjalankan bisa klik tombol tambah user baru atau mengakses url seperti berikut ini.

---

```

1 http://localhost/projectlaravel/public/user/create

```

---

Hasilnya akan nampak seperti gambar dibawah ini.

Halaman Tambah User

Nama Depan

Nama Belakang

Email

Password

Password Confirmation

Group

Selanjutnya kita akan membuat form yang digunakan untuk mengubah / update data user. Buka file **UserController.php** tambahkan code dibawah ini, dibawah function **store**.

---

```

1 public function edit($id)
2 {
3     $userbyid = Sentry::findUserByID($id);

```

---



---

```

4     $groupbyuser = $userbyid->getGroups();
5     $groupbyuser = json_decode($groupbyuser, true);
6     $group = Group::lists('name', 'id');
7     $data =
8     [
9         'userid' => $userid,
10        'group' => $group,
11        'groupbyuser' => $groupbyuser[0]['id']
12    ];
13    return View::make('sentry.user.edit', $data);
14 }
15
16 public function update($id)
17 {
18     $rules = array(
19         'nama_depan' => 'required',
20         'nama_belakang' => 'required',
21         'email' => 'required|email|unique:users,email,' . $id,
22         'password' => 'required|between:4,11|confirmed',
23         'confirmation_password' => 'between:4,11',
24         'group' => 'required',
25     );
26
27     $validator = Validator::make(Input::all(), $rules);
28
29     if ($validator->fails()) {
30         return Redirect::to('user/' . $id . '/edit')->withErrors($validator)-
31 >withInput();
32     } else {
33         try
34         {
35             $user = Sentry::findUserById($id);
36             $groupbyuser = $user->getGroups();
37             $groupbyuser = json_decode($groupbyuser, true);
38             $groupbyuser = Sentry::findGroupById($groupbyuser[0]['id']);
39             $user->removeGroup($groupbyuser);
40
41             $groupbyuser = Sentry::findGroupById(Input::get('group'));
42             $user->addGroup($groupbyuser);
43
44             $user->first_name = Input::get('nama_depan');
45             $user->last_name = Input::get('nama_belakang');
46             $user->email = Input::get('email');
47             $user->password = Input::get('password');
48
49             if ($user->save())
50             {
51                 Session::flash('message', 'Data Berhasil Ditambahkan');
52                 return Redirect::to('user');
53             }
54             else
55             {
56                 Session::flash('message', 'Data Gagal Ditambahkan');
57                 return Redirect::to('user');
58             }
59         }
60     }
61 }

```

---

---

```

52     }
53     catch (Cartalyst\Sentry\Users\UserExistsException $e)
54     {
55         Session::flash('message', 'User with this login already
56 exists. ');
57         return Redirect::to('user');
58     }
59     catch (Cartalyst\Sentry\Users\UserNotFoundException $e)
60     {
61         Session::flash('message', 'User was not found. ');
62         return Redirect::to('user');
63     }
64     catch (Cartalyst\Sentry\Groups\GroupNotFoundException $e)
65     {
66         Session::flash('message', 'Group was not found. ');
67         return Redirect::to('user');
68     }
69 }

```

---

Lalu buka file **edit.blade.php** yang telah dibuat sebelumnya didalam folder **app/views/sentry/user** lalu tambahkan code ini.

---

```

1  @extends('layouts.master')
2  @section('content')
3      < div class="row">
4          < div class="col-lg-12">
5              < h1 class="page-header">
6                  Halaman
7                  < small>Ubah User< /small>
8              < /h1>
9
10             {{ Form::model($userbyid, array('route' => array('user.update',
11 $userbyid->id), 'method' => 'PUT')) }}
12
13             < div class="form-group">
14                 {{ Form::label('nama_depan', 'Nama Depan') }}
15                 {{ Form::text('nama_depan', $userbyid->first_name,
16 array('class' => 'form-control', 'placeholder' => 'masukkan nama depan')) }}
17                 {{ '< div>'. $errors->first('nama_depan'). '< /div>' }}
18             < /div>
19             < div class="form-group">
20                 {{ Form::label('nama_belakang', 'Nama Belakang') }}
21                 {{ Form::text('nama_belakang', $userbyid->last_name,
22 array('class' => 'form-control', 'placeholder' => 'masukkan nama belakang')) }}
23                 {{ '< div>'. $errors->first('nama_belakang'). '< /div>' }}
24             < /div>
25             < div class="form-group">
26                 {{ Form::label('email', 'Email') }}
27                 {{ Form::text('email', null, array('class' => 'form-
28 control', 'placeholder' => 'masukkan email')) }}
29                 {{ '< div>'. $errors->first('email'). '< /div>' }}
30             < /div>

```

---

---

```

27         < div class="form-group">
28             {{ Form::label('password', 'Password') }}
29             {{ Form::password('password', array('class' => 'form-
control', 'placeholder' => 'masukkan password')) }}
30             {{ '< div>'. $errors->first('password'). '< /div>' }}
31         < /div>
32         < div class="form-group">
33             {{ Form::label('password_confirmation', 'Password
34 Confirmation') }}
35             {{ Form::password('password_confirmation', array('class'
=> 'form-control', 'placeholder' => 'masukkan password confirmation')) }}
36             {{ '< div>'. $errors->first('password_confirmation'). '<
37 /div>' }}
38         < /div>
39         < div class="form-group">
40             {{ Form::label('group', 'Group') }}
41             {{ Form::select('group', $group, $groupbyuser,
array('class' => 'form-control', 'placeholder' => 'Select Group...')) }}
42             {{ '< div>'. $errors->first('group'). '< /div>' }}
43         < /div>
44         {{ Form::submit('SIMPAN', array('class' => 'form-control')) }}
45     {{ Form::close() }}
46 < /div>
47 < /div>
48 @stop
49
50 // agar program berjalan lancar hapus spasi setelah tanda <

```

---

Untuk mencoba menjalankan silahkan klik tombol ubah, lalu ubah data yang diinginkan dan klik simpan. Berikut hasil dari form ubah / update user menggunakan sentry.

Halaman Ubah User

Nama Depan	<input type="text" value="andi"/>
Nama Belakang	<input type="text" value="nantana"/>
Email	<input type="text" value="admin@gmail.com"/>
Password	<input type="password" value="masukkan password"/>
Password Confirmation	<input type="password" value="masukkan password confirmation"/>
Group	<input type="text" value="admin"/>
<input type="button" value="SIMPAN"/>	

Sekarang buka file **UserController.php** dan tambahkan code dibawah ini, dibawah function **update**.

---

```

1 public function destroy($id)
2 {
3     try
4     {
5         $user = Sentry::findUserById($id);

```

---

---

```
5      $user->delete();
6    }
7    catch (Cartalyst\Sentry\Users\UserNotFoundException $e)
8    {
9        Session::flash('message', 'User was not found. ');
10       return Redirect::to('user');
11    }
12    Session::flash('message', 'Data Berhasil Dihapus');
13    return Redirect::to('user');
14 }
15
```

---

Selesai untuk menjalankannya cukup klik tombol hapus.

Untuk tambahan saja, untuk menambahkan menu group dan user disidebar berikut ini codenya tambahkan di file **sidebar.blade.php**.

---

```
1 < li>< a href="{ URL::to('group') }" > List Group Sentry< /a>< /li>
2 < li>< a href="{ URL::to('user') }" > List User Sentry< /a>< /li>
```


---

Tutorial diatas hanya cara dasar **menggunakan package cartalyst sentry** untuk membuat user, mengubah user dan menghapus user serta memilih group untuk si user. Sekian tutorial package sentry untuk laravel bagian user untuk selanjutnya kita akan belajar **authentication and login cartalyst sentry** menggunakan framework laravel. Jika ada hal yang kurang dipahami anda bisa berkomentar dibawah ini. Dan jangan lupa di share ya kawan terimakasih.

# BAB XII

## 12. Cara Menggunakan Package Sentry Di Framework Laravel Bagian Authentication Dan Login

Akhirnya kita sampai pada tutorial authentication and login pada sentry di framework laravel. Pada tutorial kali ini kita akan membahas bagaimana cara membuat authentication dan login cartalyst sentry di framework laravel. Pada dasarnya komponen yang perlu kita pahami di tutorial ini adalah **Sentry::authenticate** dan **hasAccess**. Ingat agar lancar anda mengikuti tutorial ini anda harus mengikuti dari [awal tutorial sentry ini](#). Berikut ini tampilan halaman login yang dibuat menggunakan package cartalyst sentry dan framework laravel.



Pada tutorial sebelumnya kita sudah membahas membuat group dan user. Pada group ketika kita tambah dan ubah group kita juga menambah atau mengubah bagian permissions. Di tutorial group untuk modul seperti masih saya buat manual (daftar modul dengan checkbox masih manual belum dalam database, baca [bagian group](#)). Disitu ada beberapa modul yang saya list **User**, **CRUD Query Builder** dan **CRUD Eloquent ORM**. Lebih jelasnya lihat gambar dibawah ini.

User: ☐ Read ☐ Create ☐ Update ☐ Destroy

CRUD Query Builder: ☒ Read ☒ Create ☒ Update ☒ Destroy

CRUD Eloquent ORM: ☐ Read ☐ Create ☐ Update ☐ Destroy

Dari gambar diatas kita yang akan kita uji coba di modul bagian **CRUD Query Builder**. Kalau kita centang read saja misal maka aplikasi bisa diakses untuk membaca saja daftar profile. Jika kita centang read dan create saja, maka aplikasi hanya bisa membaca dan membuat profile baru dan seterusnya.

Kita langsung praktekan saja, buat file bernama **login.blade.php** letakkan pada folder **app/views/sentry**. Setelah itu file login.blade.php isikan dengan code dibawah ini.

---

```
1 < !DOCTYPE html>
2 < html lang="en">
3 < head>
  < title>Halaman Login< /title>
```

---

---

```

4      {{ HTML::style('assets/css/bootstrap.min.css') }}
5      {{ HTML::style('assets/css/sb-admin.css') }}
6      {{ HTML::style('assets/font-awesome-4.1.0/css/font-awesome.min.css')
7  }}
8  < /head>
9  < body>
10
11      < div class="row">
12          < div class="col-lg-4 text-center">
13
14              < /div>
15              < div class="col-lg-4 text-center">
16                  < div class="panel panel-default">
17                      < div class="panel-body">
18                          {{ Form::open(array('url' => 'login')) }}
19                          < h1>Login< /h1>
20                          < p>
21                              @if (Session::has('message'))
22                                  {{ Session::get('message') }}
23                              @endif
24                              {{ $errors->first('email') }}
25                              {{ $errors->first('password') }}
26                          < /p>
27
28                          < p>
29                              {{ Form::label('email', 'Email') }}
30                              {{ Form::text('email',
31 Input::old('email'), array('class' => 'form-
32 control','placeholder'=>'Masukkan Email')) }}
33                          < /p>
34
35                          < p>
36                              {{ Form::label('password', 'Password')
37 array('class' => 'form-control','placeholder'=>'Masukkan Password')) }}
38                          < /p>
39
40                          < p>{{ Form::submit('Login', array('class'
41 => 'form-control')) }}< /p>
42                          {{ Form::close() }}
43                      < /div>
44                  < /div>
45              < /div>
46          < div class="col-lg-4 text-center">
47
48              < /div>
49      < /div>
50
51  < /body>
52  < /html>

```

// hilangkan spasi setelah tanda < agar program berjalan dengan benar

---

Jika selesai simpan, selanjutnya buka **UserController.php** didalam folder **controllers** yang telah kita buat sebelumnya dan tambahkan function atau code dibawah ini.

---

```
1 public function login()
2 {
3     return View::make('sentry.login');
4 }
5 public function doLogin()
6 {
7     $rules = array(
8         'email' => 'required|email',
9         'password' => 'required|alphaNum|min:5'
10    );
11    $validator = Validator::make(Input::all(), $rules);
12
13    if ($validator->fails()) {
14        return Redirect::to('login')
15            ->withErrors($validator)
16            ->withInput(Input::except('password'));
17    } else {
18        try
19        {
20
21            $credentials = array(
22                'email' => Input::get('email'),
23                'password' => Input::get('password')
24            );
25
26            // digunakan untuk login
27            $user = Sentry::authenticate($credentials, false);
28            if ($user) {
29                return Redirect::to('/');
30            }
31        }
32        catch (Cartalyst\Sentry\Users\LoginRequiredException $e)
33        {
34            Session::flash('message', 'Login field is required. ');
35            return Redirect::to('login')
36                ->withInput(Input::except('password'));
37        }
38        catch (Cartalyst\Sentry\Users>PasswordRequiredException $e)
39        {
40            Session::flash('message', 'Password field is required. ');
41            return Redirect::to('login')
42                ->withInput(Input::except('password'));
43        }
44        catch (Cartalyst\Sentry\Users\WrongPasswordException $e)
45        {
46            Session::flash('message', 'Wrong password, try again. ');
47            return Redirect::to('login')
48                ->withInput(Input::except('password'));
```

---

---

```

46         ->withInput(Input::except('password'));
47     }
48     catch (Cartalyst\Sentry\Users\UserNotFoundException $e)
49     {
50         Session::flash('message', 'User was not found. ');
51         return Redirect::to('login')
52         ->withInput(Input::except('password'));
53     }
54     catch (Cartalyst\Sentry\Users\UserNotActivatedException $e)
55     {
56         Session::flash('message', 'User is not activated. ');
57         return Redirect::to('login')
58         ->withInput(Input::except('password'));
59     }
60     // The following is only required if the throttling is enabled
61     catch (Cartalyst\Sentry\Throttling\UserSuspendedException $e)
62     {
63         Session::flash('message', 'User is suspended. ');
64         return Redirect::to('login')
65         ->withInput(Input::except('password'));
66     }
67     catch (Cartalyst\Sentry\Throttling\UserBannedException $e)
68     {
69         Session::flash('message', 'User is banned. ');
70         return Redirect::to('login')
71         ->withInput(Input::except('password'));
72     }
73 }

```

---

Kemudian pada file **filters.php** di folder **app** tambahkan code ini.

---

```

1 // digunakan untuk mengecek apakah user / pengguna sudah login denga benar
2 Route::filter('sentry_auth', function() {
3     if(!Sentry::check()){
4         return Redirect::to('login');
5     }
6 });
7 // digunakan untuk mengecek apakah route / url / halaman yang diakses
8 // di user mempunyai hak untuk mengaksesnya
9 Route::filter('hasAccess', function($route, $request, $value)
10 {
11     try
12     {
13         $user = Sentry::getUser();
14         if( ! $user->hasAccess($value))
15         {
16             // jika user tidak mempunyai hak untuk akses maka dikembalikan ke
17             // halaman login.
18             // ini bisa anda ubah ke error 404 / 503 atau bahkan ke halaman yang

```

---



---

```

18 anda buat sendiri
19     return Redirect::to('login');
20     }
21     }
22     catch (Cartalyst\Sentry\Users\UserNotFoundException $e)
23     {
24         return Redirect::to('login');
25     }
26 });
27
28

```

---

Sekarang yang terakhir buka file **routes.php** didalam folder **app**. Cari code dibawah ini jika mengikuti dari awal tutorial.

---

```

1 Route::get('crud', 'CrudController@index');
2 Route::get('crud/create', 'CrudController@create');
3 Route::post('crud/create', 'CrudController@store');
4 Route::get('crud/edit/{id}', 'CrudController@edit');
5 Route::post('crud/update/{id}', array('as' => 'crud.update', 'uses' =>
6 'CrudController@update'));
7 Route::get('crud/destroy/{id}', 'CrudController@destroy');

```

---

Lalu hapus dan ubah menjadi seperti dibawah ini, jika tidak mengikuti tutorial dari awal bisa menambahkan route seperti ini.

---

```

1 // route digunakan untuk login
2 Route::get('login', array('uses' => 'UserController@login'));
3 Route::post('login', array('uses' => 'UserController@doLogin'));
4 // membuat group route dan sebelum mengakses route didalam group route ini
5 kita
6 // harus login terlebih dahulu di tandai dengan code "'before' =>
7 'sentry_auth'"
8 Route::group(array('before' => 'sentry_auth'), function()
9 {
10     Route::get('crud', array(
11     // as digunakan untuk penamaan route silahkan dibaca bagian route
12     'as' => 'crud.index',
13     // digunakan untuk mengecek user berhak atau tidak mengakses route ini.
14     Lihat fuction di filter
15     //sebelumnya "hasAccess" dan cq.b.read ini adalah permissions atau kalau
16     kita lihat di view group bagian
17     //checkbox ada checkbox dengan nama "cq.b.read".
18     'before' => 'hasAccess:cq.b.read',
19     'uses' => 'CrudController@index'
20     ));
21
22     Route::get('crud/create', array(
23     'as' => 'crud.create',
24     'before' => 'hasAccess:cq.b.create',
25     'uses' => 'CrudController@create'
26     ));
27
28

```

---

---

```
23 });
24
25 Route::post('crud/create', array(
26     'as' => 'crud.store',
27     'before' => 'hasAccess:cqb.create',
28     'uses' => 'CrudController@store'
29 ));
30
31 Route::get('crud/edit/{id}', array(
32     'as' => 'crud.edit',
33     'before' => 'hasAccess:cqb.update',
34     'uses' => 'CrudController@edit'
35 ));
36
37 Route::post('crud/update/{id}', array(
38     'as' => 'crud.update',
39     'before' => 'hasAccess:cqb.update',
40     'uses' => 'CrudController@update'
41 ));
42
43 Route::get('crud/destroy/{id}', array(
44     'as' => 'crud.destroy',
45     'before' => 'hasAccess:cqb.destroy',
46     'uses' => 'CrudController@destroy'
47 ));
48
49 // route untuk logout
50 Route::get('logout', array('uses' => 'UserController@logout'));
51
52 });
```

---

Jika anda mengikuti tutorial ini dari awal, maka anda bisa menambahkan menu logout pada menu sidebar aplikasi ini dengan code dibawah ini.

---

```
1 < li>
2     < a href="{ URL::to('logout') }">< i class="fa fa-fw fa-power-off"><
3 /i> Logout< /a>
4 < /li>
5
6 // Hilangkan spasi setelah tanda < agar program berjalan lancar
```

---

Selesai dilahkan buat group terlebih dahulu, lalu buat user dan coba jalankan serta pahami apa yang terjadi. Jika ada kendala silahkan komentar dibawah ini. Itulah tutorial **cara membuat login** menggunakan cartalyst sentry di **framework laravel**.

# BAB XIII

## 13. Cara Menggunakan Package Sentry Di Framework Laravel Bagian Reset Password atau Forgot Password User

Pada kesempatan kali ini seputar pemrograman akan membahas **reset password** atau **forgot password** user. Pada tutorial sebelumnya kita sudah membahas install, register, group, user sampai auth dan login pada cartalyst sentry di framework laravel. Untuk membuat reset password caranya sangat mudah karena sentry sendiri sudah menyediakan fungsinya. Salah satunya yaitu menggunakan `getResetPasswordCode()` , dimana fungsi ini akan megenerate dan mengambil reset password code didalam database. Berikut ini adalah tampilan reset password sentry di framework laravel.



Langsung saja kita praktekan pertama bukan project anda, jika ingin lebih mudah anda harus mengikut tutorial ini dari awal. Jika project sudah terbuka cari file **UserController.php** didalam folder **app/controllers**. Jika sudah tambahkan fuction / code dibawah ini.

```
1 public function reset()
2 {
3     return View::make('sentry.reset');
4 }
5 public function doReset()
6 {
7
8     try
9     {
10         $email = Input::get('email');
11         $user = Sentry::findUserByLogin($email);
12         Session::put('sessid', $user->id);
13         $resetCode = $user->getResetPasswordCode();
14
15         Mail::send('emails.auth.reminder', array('token'=>$resetCode),
16 function($message) use($email)
17     {
18         $message->from(
19             Input::get('administrator@seputarpemrograman.com'),
20             Input::get('seputarpemrograman') );
21     }
22 }
```

---

```

20     $message->to($email, 'User')->subject( Input::get('Reset
21 Password') );
22     });
23     }
24     catch (Cartalyst\Sentry\Users\UserNotFoundException $e)
25     {
26         Session::flash('message', 'User was not found. ');
27         return Redirect::to('password/reset');
28     }
29     Session::flash('message', 'Silahkan buka email anda untuk reset
password');
30     return Redirect::to('login');
31 }
32 }
33
34 public function validation($token)
35 {
36     $data =
37     [
38         'token' => $token
39     ];
40     return View::make('sentry.validation', $data);
41 }
42
43 public function doValidation($token)
44 {
45     try
46     {
47         $id = Session::get('sessid');
48         $user = Sentry::findUserById($id);
49
50         if ($user->checkResetPasswordCode($token))
51         {
52             if ($user->attemptResetPassword($token,
Input::get('password')))
53             {
54                 return Redirect::to('/');
55             }
56             else
57             {
58                 Session::flash('message', 'Reset Password Tidak
Berhasil');
59                 return Redirect::to('validation/{token}');
60             }
61         }
62         else
63         {
64             // The provided password reset code is Invalid
65         }
66     }
67     catch (Cartalyst\Sentry\Users\UserNotFoundException $e)
68     {
69         Session::flash('message', 'User was not found. ');
70         return Redirect::to('validation/{token}');
71     }
72 }

```

---

---

68  
69 }

---

Lalu simpan, sekarang buatlah file bernama **reset.blade.php** didalam folder **app/views/sentry** lalu isikan dengan code ini.

---

```
1 < !DOCTYPE html>
2 < html lang="en">
3 < head>
4     < title>Halaman Reset Password< /title>
5     {{ HTML::style('assets/css/bootstrap.min.css') }}
6     {{ HTML::style('assets/css/sb-admin.css') }}
7     {{ HTML::style('assets/font-awesome-4.1.0/css/font-awesome.min.css') }}
8 < /head>
9 < body>
10     < div class="row">
11         < div class="col-lg-4 text-center">
12
13         < /div>
14         < div class="col-lg-4 text-center">
15             < div class="panel panel-default">
16                 < div class="panel-body">
17                     {{ Form::open(array('url' =>
18 'password/reset')) }}
19                     < p>
20                         @if (Session::has('message'))
21                             {{ Session::get('message') }}
22                         @endif
23
24                         {{ $errors->first('email') }}
25                     < /p>
26
27                     < p>
28                         {{ Form::label('email', 'Email') }}
29                         {{ Form::text('email',
30 Input::old('email'), array('class' => 'form-
31 control','placeholder'=>'Masukkan Email')) }}
32                     < /p>
33
34                     < p>{{ Form::submit('Kirim', array('class'
35 => 'form-control')) }}< /p>
36                     {{ Form::close() }}
37                 < /div>
38             < /div>
39         < /div>
40     < /div>
```

---

---

```
41
42 < /body>
43 < /html>
44
45 // hilangkan tanda spasi setelah tanda <
46
```

---

Simpan, kemudian buat file sekali lagi bernama **validation.blade.php** didalam folder **app/views/sentry** lalu isi dengan code dibawah ini.

---

```
1 < !DOCTYPE html>
2 < html lang="en">
3 < head>
4     < title>Halaman Reset Password< /title>
5     {{ HTML::style('assets/css/bootstrap.min.css') }}
6     {{ HTML::style('assets/css/sb-admin.css') }}
7     {{ HTML::style('assets/font-awesome-4.1.0/css/font-awesome.min.css') }}
8 < /head>
9 < body>
10     < div class="row">
11         < div class="col-lg-4 text-center">
12
13         < /div>
14         < div class="col-lg-4 text-center">
15             < div class="panel panel-default">
16                 < div class="panel-body">
17                     {{ Form::open(array('route' =>
18 array('check.validation', $token))) }}
19                     < p>
20                         @if (Session::has('message'))
21                             {{ Session::get('message') }}
22                         @endif
23                     < /p>
24
25                     < p>
26                         {{ Form::label('password', 'Password') }}
27                         {{ Form::password('password', array('class' => 'form-control', 'placeholder'=>'Password')) }}
28                         {{ $errors->first('password') }}
29                     < /p>
30
31                     < p>
32                         {{
33 Form::label('password_confirmation', 'Konfirmasi Password') }}
34                         {{
35 Form::password('password_confirmation', array('class' => 'form-
36 control', 'placeholder'=>'Konfirmasi Password')) }}
37                         {{ $errors->first('password_confirmation') }}
38                     < /p>
```

---

---

```

38
39             < p>{{ Form::submit('Reset', array('class'
40 => 'form-control')) }}< /p>
41                 {{ Form::close() }}
42             < /div>
43         < /div>
44     < /div>
45     < div class="col-lg-4 text-center">
46
47         < /div>
48     < /div>
49 < /body>
50 < /html>
    // hilangkan tanda spasi setelah tanda <

```

---

Selesai simpan, lalu buka file **login.blade.php** didalam folder **app/views/sentry** lalu modifikasi atau tambahkan code dibawah ini ditempat yang anda suka.

---

```

1 < a href="{{ URL::to('password/reset') }}">Reset Password< /a>
2
3 // hilangkan tanda spasi setelah tanda <

```

---

Sekarang buka file bernama **reminder.blade.php** didalam folder **app/views/emails/auth**. Dan rubah menjadi seperti ini.

---

```

1 < !DOCTYPE html>
2 < html lang="en-US">
3     < head>
4         < meta charset="utf-8">
5     < /head>
6     < body>
7         < h2>Password Reset< /h2>
8
9         < div>
10             To reset your password, complete this form: {{
11 URL::to('validation', array($token)) }}.
12         < /div>
13     < /body>
14 < /html>
15 // hilangkan tanda spasi setelah tanda <

```

---

Untuk selanjutnya buka file **routes.php** didalam folder **app** tambahkan juga code atau route berikut ini.

---

```

1 Route::get('password/reset', 'UserController@reset');
2 Route::post('password/reset', 'UserController@doReset');
3

```

---

```
4 Route::get('validation/{token}', array('as' => 'validation', 'uses' =>
5 'UserController@validation'));
Route::post('validation/{token}', array('as' => 'check.validation', 'uses'
=> 'UserController@doValidation'));
```

Yang terakhir jangan lupa konfigurasi **mail.php** didalam folder **app/config**. Jika selesai konfigurasi untuk mencoba silahkan klik link reset password atau bisa mengakses url dibawah ini.

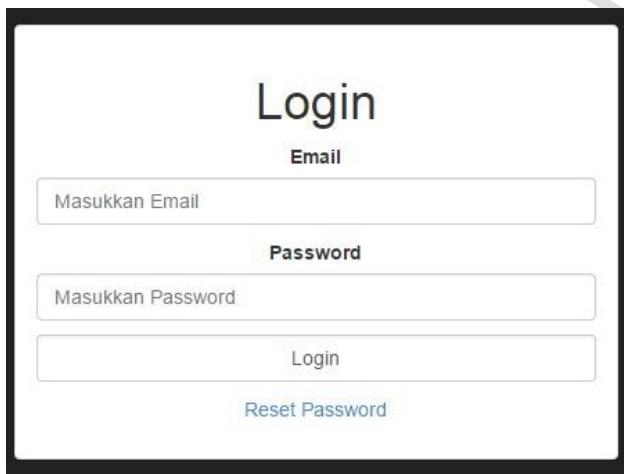
1 <http://localhost/projectlaravel/public/password/reset>

Maka akan muncul form yang digunakan memasukkan email agar code reset terkirim ke email yang kita inputkan sebelumnya, berikut tampilannya.



The screenshot shows a web form titled "Email". It contains a text input field with the placeholder text "Masukkan Email" and a "Kirim" (Send) button below it.

Sedangkan tampilan halaman login menjadi seperti ini.



The screenshot shows a web form titled "Login". It contains two text input fields: the first is labeled "Email" with the placeholder "Masukkan Email", and the second is labeled "Password" with the placeholder "Masukkan Password". Below these fields is a "Login" button and a link labeled "Reset Password" in blue text.

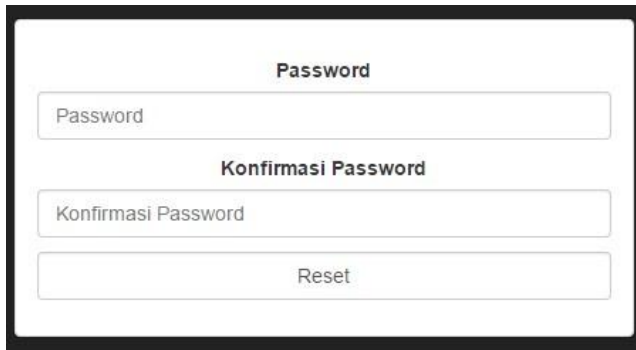
Untuk mencoba tentunya klik link reset password lalu akan muncul seperti form diatas isikan email sesuai yang ada didalam database lalu klik tombol kirim dan cek inbox email anda. Didalam pesan email anda akan ada sebuah link lalu klik, contohnya seperti dibawah ini.

#### Password Reset

To reset your password, complete this form: <http://localhost/projectlaravel/public/validation/WGc9nhH0eo1oZrTRvfmBeyWt2ZrcMxP7faaPQjWe2>

Jika kita klik link tersebut maka akan muncul form untuk memasukkan password baru, seperti gambar dibawah ini.





The image shows a web form for password confirmation. It is enclosed in a black border. At the top, the word "Password" is centered. Below it is a text input field with the placeholder text "Password". Underneath that, the words "Konfirmasi Password" are centered. Below this is another text input field with the placeholder text "Konfirmasi Password". At the bottom of the form is a button labeled "Reset".

**Makan setelah inputkan password dengan benar akan diarahkan ke halaman utama aplikasi atau website.**