Pengembangan Aplikasi dengan Laravel

Langkah-Langkah Pembangunan Website

Husni

Husni@Trunojoyo.ac.id

Outline

- Struktur direktori Laravel
- Memahami rute dalam web.php
- Mengubah home page Laravel
- Menambah halaman ke dalam website
  - Membuat dan Menggunakan controller
- Integrasi Laravel dengan Twitter Bootstrap
  - Menggunakan CDN dan Precompiled Source Code Bootstrap
- Menambahkan komponen Twitter Bootstrap
- Mempelajari template Blade
  - Membuat dan memperluas Layout Master
- Menggunakan tema Bootstrap lain
- Memperhalus Layout website
  - Mengubah navbar & home page
- Rangkuman

Memahami Rute: routers/web.php

- Berfungsi sebagai routing. Memberitahukan Laravel untuk mengambil URL request dan menetapkan tindakan khusus untuk request tersebut.
- Default web.php: hanya satu rute

```
Route::get('/', function() {
    return view('welcome');
});
```

- Rute ini memberitahukan Laravel untuk mengembalikan **view welcome** saat ada **request GET** ke **URL root** (**/**).

Apa itu View?

- Jika kita ingin mengedit homepage maka harus mengubah isi view welcome
- View berisi HTML yang dilayani oleh aplikasi kita. Contoh view HTML:

```
<html>
<body>
    <p> A simple view </p>
</body
</html>
```

- Semua view diletakkan di dalam folder resources/views. File pertama yang ada di sana: **welcome.blade.php**

Mengubah Halaman Welcome

- Ganti baris-baris berikut:

```
<div class="links">
    <a href="https://laravel.com/docs">Documentation</a>
    <a href="https://laracasts.com">Laracasts</a>
    <a href="https://laravel-news.com">News</a>
    <a href="https://forge.laravel.com">Forge</a>
    <a href="https://github.com/laravel/laravel">GitHub</a>
</div>
```

- dengan:

```
<div class="quote">
    Allah tidak menguji kita kecuali untuk menaikkan derajat kita!
</div>
```


Laravel
localhost/blog/public/
Laravel
Allah tidak menguji kita kecuali untuk menaikkan derajat kita!

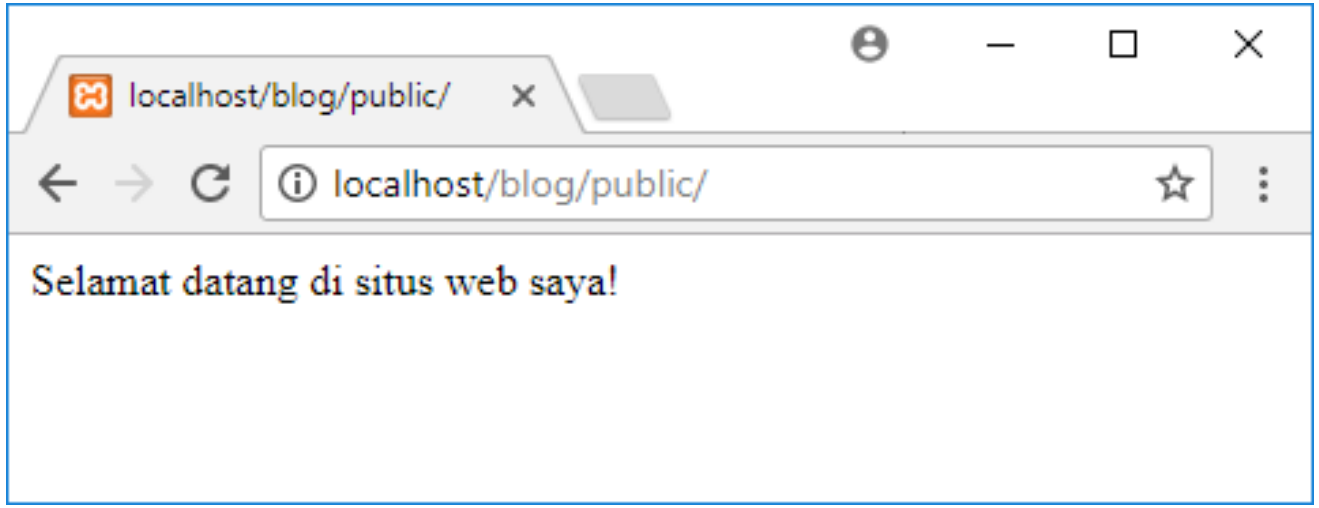
Mengubah Rute di web.php

- Ubah:

```
Route::get('/', function () {
    return view('welcome');
});
```

- Menjadi:

```
Route::get('/', function() {
    return 'Selamat datang di situs web saya!';
});
```



- Kita telah menggunakan fungsi anonymous: ***Closure***, yaitu fungsi yang tidak mempunyai nama

Closure vs. Controller

- Closure digunakan untuk menghandel routing dalam aplikasi kecil
- Pada aplikasi besar, dianjurkan menggunakan Controller
- Direkomendasikan menggunakan Controller. Struktur kode lebih baik dan mudah. Misal: Tindakan pengguna dihandel UserController, request ke posting ditangani oleh PostController.
- Kekurangan Controller: kita harus membuat file untuk setiap Controller, perlu sedikit waktu ekstra.

Menambah Halaman ke Aplikasi

- Website akan mempunyai 3 halaman:
  - Home
  - Tentang
  - Kontak
- Banyak halaman tersebut akan ditangani requestnya oleh PagesController.
- Cara membuat Controller:
  - Secara manual
  - Memanfaatkan artisan

Membuat Controller Secara Manual

- Buat file **PagesController.php** menggunakan Text Editor anda, letakkan di dalam **app/Http/Controllers/**
- Update isi file tersebut dengan isi berikut:

```
<?php
namespace App\Http\Controllers;

class PagesController extends Controller {
   public function home() {
         return view('welcome');
   }
}
```

Membuat Controller Menggunakan Artisan

- **Artisan** adalah tool baris perintah (*Command Line*) Laravel yang membantu kita untuk mengerjakan tugas-tugas yang kita “benci” lakukan secara manual
- Menggunakan Artisan kita dapat membuat models, views, controllers, migrations dan banyak hal lain.
- Bukalah Terminal atau Git Bash, jalankan perintah berikut:

```
php artisan make:controller PagesController
```

```
λ php artisan make:controller PagesController
Controller created successfully.
```

Membuat Controller Menggunakan Artisan

- Secara default, PagesController otomatis ini besisi:

```
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;

class PagesController extends Controller
{
    //
}
```

Membuat Controller Menggunakan Artisan

- Buatkan fungsi home() di dalam PagesController:

```php
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;

class PagesController extends Controller {
    public function home() {
        return view('welcome');
   }
}
```

Menggunakan PagesController

- Lakukan perubahan rute default pada file routes/web.php:

```
Route::get('/', 'PagesController@home’);
```

- Ini memberitahukan Laravel untuk mengeksekusi fungsi home (di dalam PagesController) saat ada request ke URL root (/)

Membuat Halaman lain

- PagesController mudah menangani halaman welcome, pembuatan halaman tentang dan kontak juga mudah.
- Edit file routes/web.php an tambahkan dua baris berikut:

```
Route::get(‘/tentang', 'PagesController@tentang');
Route::get(‘/kontak', 'PagesController@kontak’);
```

- Sehingga web.php sekarang menjadi:

```
Route::get('/', 'PagesController@home');
Route::get(‘/tentang', 'PagesController@tentang');
Route::get(‘/kontak', 'PagesController@kontak');
```

Membuat Halaman lain

- Pada PagesController tambahkan fungsi about dan contact berikut:

```
public function tentang() {
   return view('tentang');
}

public function kontak() {
   return view('kontak');
}
```

- Selanjutnya adalah membuatan file view **tentang** dan **kontak.**

View Tentang: resources/tentang.blade.php

- Salin file welcome.blade.php menjadi tentang.blade.php
- Lakukan perubahan, terutama bagian <title> dan <body> sebagai berikut:

```
<title>Halaman Tentang</title>
 <body>
        <div class="container">
            <div class="content">
                <div class="title">Tentang RAI-2017</div>
                <div class="quote">RAI-2017 merupakan situs web
kuliah Rekayasa Aplikasi Internet tahun 2017!</div>
            </div>
        </div>
    </body>
```

View Kontak: resources/kontak.blade.php

- Salin file tentang.blade.php menjadi kontak.blade.php
- Lakukan perubahan, terutama bagian <title> dan <body> sebagai berikut:

```
<title>Halaman Kontak</title>
 <body>
        <div class="container">
            <div class="content">
                <div class="title">Alamat Kontak</div>
                <div class="quote">admin@rai2017.com</div>
            </div>
        </div>
    </body>
```

Tentang dan Kontak

Integrasi Laravel dan Bootstrap

- Framework front-end paling popular saat ini: **Twitter Bootstrap**
- Digunakan untuk dengan cepat mengembangkan aplikasi web yang *responsive* dan *mobile-ready*.
- Jutaan situs web pupolar dan cantik dibuat dengan Bootstrap.
- Bootstrap dan dokumentasinya dapat diperoleh di: http://getbootstrap.com
- 2 metode integrasi Laravel dengan Bootstrap paling top dan mudah:
  - Menggunaan CDN Bootstrap
  - Menggunakan File Bootstrap Precompiled

Menggunakan CDN Bootstrap

- Cara tercepat memasukkan bootstrap ke dalam Laravel
- Bukan **home.blade.php** dan letakkan kode berikut pada bagian **head**

```
<link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.4/css\
/bootstrap.min.css">
<link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.4/css\
/bootstrap-theme.min.css">

<script src="//code.jquery.com/jquery-1.11.3.min.js"></script>
<script src="https://maxcdn.bootstrapcdn.com/bootstrap/
3.3.4/js/bootstrap.min.js"></script>
```

https://getbootstrap.com/docs/3.3/getting-started/

Search

Bootstrap

Compiled and minified CSS, JavaScript, and fonts. No docs or original source files are included.

Download Bootstrap

Source code

Source Less, JavaScript, and font files, along with our docs. **Requires a Less compiler and some setup.**

Download source

Sass

Bootstrap ported from Less to Sass for easy inclusion in Rails, Compass, or Sass-only projects.

Download Sass

Bootstrap CDN

The folks over at MaxCDN graciously provide CDN support for Bootstrap's CSS and JavaScript. Just use these Bootstrap CDN links.

```
<!-- Latest compiled and minified CSS -->
<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css"
integrity="sha384-BVYiiSIFeK1dGmJRAkycuHAHRg32OmUcww7on3RYdg4Va+PmSTsz/K68vbdEjh4u" crossorigin="anonymous">

<!-- Optional theme -->
<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap-theme.min.css"
integrity="sha384-rHyoN1iRsVXV4nD0JutlnGaslCJuC7uwjduW9SVrLvRYooPp2bWYgmgJQIXwl/Sp" crossorigin="anonymous">

<!-- Latest compiled and minified JavaScript -->
<script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js" integrity="sha384-
Tc5IQib027qvyjSMfHjOMaLkfuWVxZxUPnCJA7l2mCWNIpG9mGCD8wGNIcPD7Txa" crossorigin="anonymous"></script>
```

home.blade.php: welcome baru

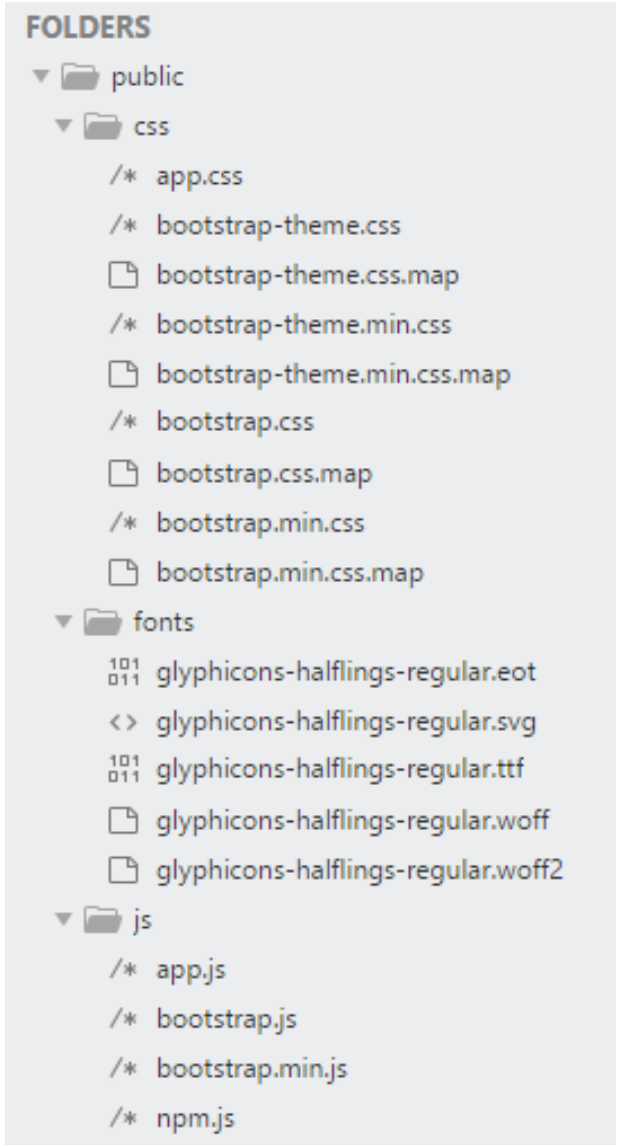
```
<html>
<head>
            <title>Home Page</title>
            <!-- Latest compiled and minified CSS -->
            <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css" integrity="sha384-
            BVYiiSIFeK1dGmJRAkycuHAHRg32OmUcww7on3RYdg4Va+PmSTsz/K68vbdEjh4u" crossorigin="anonymous">
            <!-- Optional theme -->
            <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap-theme.min.css" integrity="sha384-
            rHyoN1iRsVXV4nD0JutlnGaslCJuC7uwjduW9SVrLvRYooPp2bWYgmgJQIXwl/Sp" crossorigin="anonymous">
            <!-- Latest compiled and minified JavaScript -->
            <script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js" integrity="sha384-
            Tc5IQib027qvyjSMfHjOMaLkfuWVxZxUPnCJA7l2mCWNIpG9mGCD8wGNIcPD7Txa" crossorigin="anonymous"></script>
            <script src="http://code.jquery.com/jquery-3.2.1.min.js"></script>
</head>
<body>
            <div class="container">
                        <div class="content">
                                    <div class="title">Situs Web RAI-2017</div>
                                    <div class="quote">Selamat datang di Situs web RAI 2017.</div>
                        </div>
            </div>
</body>
</html>
```

Catatan: Lakukan perubahan pada file PagesController, arahkan home ke view(‘home’)

Menggunakan File Bootstrap *Precompiled*

- Twitter Bootstrap dibangun menggunakan Less: CSS pre-processor (lesscss.org)
- Less memungkinkan kita menggunakan variable, mixins, functions dan teknik lain untuk meningkatkan CSS
- Browser tidak memahami Less. Kita harus meng-**compile** semua file Less menggunakan **Less compiler** untuk menghasilkan file CSS
- Alhamdulillah, Bootstrap telah menyediakan CSS, JS & font tercompiler
- Download versi terakhir boostrap dari getbootstrap.com/getting-started
- Uncompress file bootstrap, diperoleh 3 folder: css, js dan fonts
- Pindahkan 3 folder tersebut ke dalam direktori public

File-file dari Bootstrap Tercompile



- Saat tutorial ini ditulis (Oktober 2017), Bootstrap terbaru: bootstrap-3.3.7
- Secara default, Laravel telah membuat folder **css** dan **fonts**. Folder **fonts** juga mengandung semua font jenis **glyphicons**.

Fungsi Asset

- Masukkan link ke file bootstrap precompiled pada bagian **head**
- Twitter Bootstrap memerlukan **jQuery**, dapat didownload dari: https://jquery.com/download
- Letakkan file jQuery ke dalam folder **public**, kemudian gunakan kode berikut untuk mensitasinya:

```
<script src="{!! asset('js/jquery-3.2.1.min.js') !!}"></script>
```

- Atau guakan **jQuery CDN** tanpa harus mendownloadnya:

```
<script src="//code.jquery.com/jquery-3.2.1.min.js"></script>
```

- **Kode lengkap bagian head dari home.blade.php:**

```
<link rel="stylesheet" type="text/css" href="{!!
asset('css/bootstrap.min.css') !!}" >
<link rel="stylesheet" href="{!! asset('css/bootstrap-theme.min.css') !!}">
<script src="{!! asset('js/jquery-3.2.1.min.js') !!}"></script>
<script src="{!! asset('js/bootstrap.min.js') !!}"></script>
```

Link Relatif

- Kita telah mengunakan fungsi **asset** untuk menghubungkan file CSS dan JS ke aplikasi kita.
- Fungsi **asset** juga dapat digunakan untuk link ke images, fonts dan file publik lain.
- Selain fungsi asset, kita dapat gunakan link relatif:

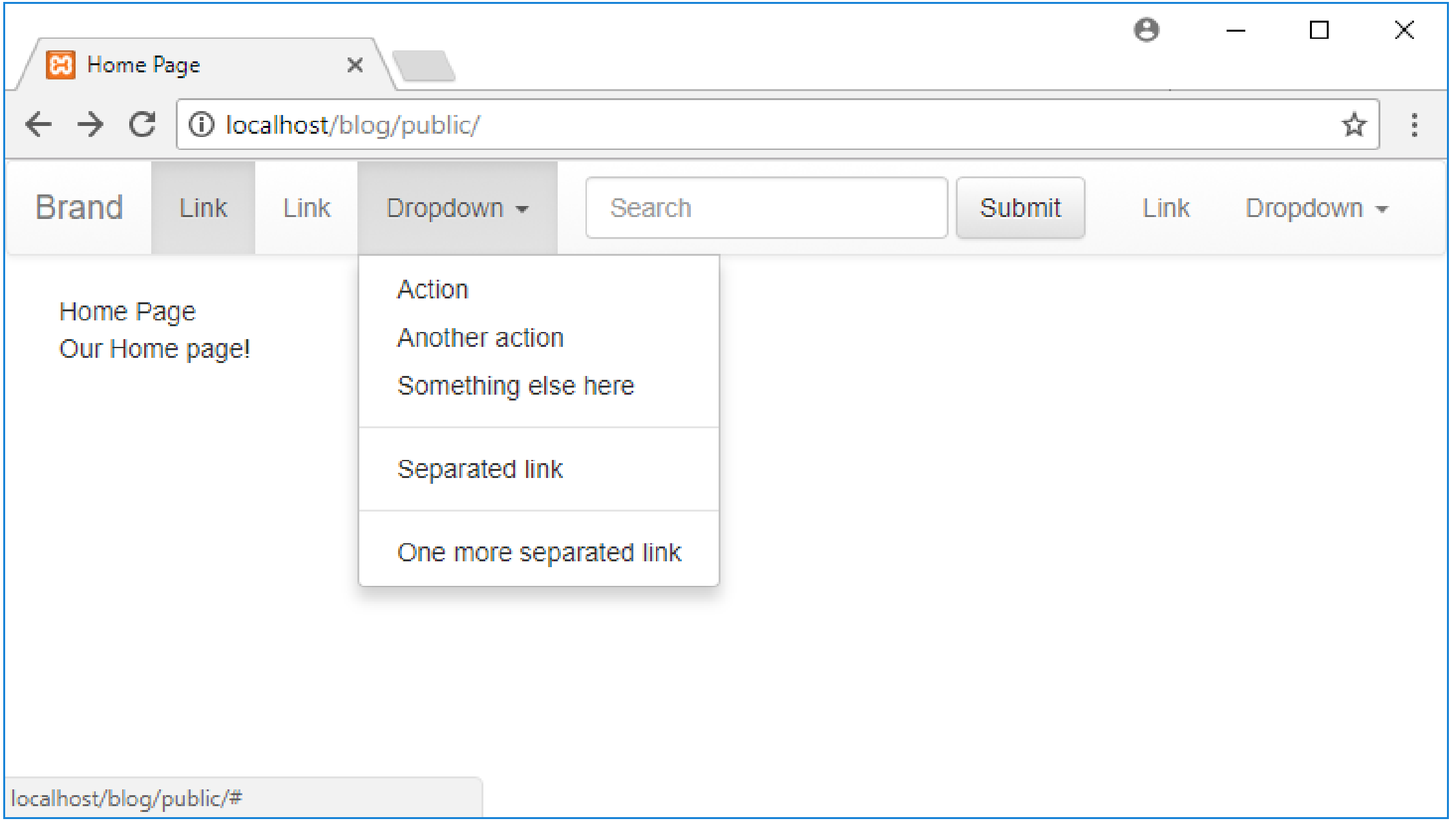
```
<link rel="stylesheet" type="text/css" href="/css/bootstrap.min.css"
>
<link rel="stylesheet" href="/css/bootstrap-theme.min.css">
<script src="/js/jquery-3.2.1.min.js"></script>
<script src="/js/bootstrap.min.js"></script>
```

home.blade.php

```
<html>
<head>
	<title>..::RAI-2017::..</title>
	<link rel="stylesheet" type="text/css" \
		href="{!! asset('css/bootstrap.min.css') !!}" >
	<link rel="stylesheet" href="{!! asset('css/bootstrap-theme.min.css') !!}">
	<script src="{!! asset('js/jquery-3.2.1.min.js') !!}"></script>
	<script src="{!! asset('js/bootstrap.min.js') !!}"></script>
</head>
<body>
	<div class="container">
		<div class="content">
			<div class="title">Situs web RAI-2017</div>
			<div class="quote">Selamat datang di Situs web RAI-2017.</div>
		</div>
	</div>
</body>
</html>
```

Menambah Komponen Bootstrap

- Ada banyak komponen Bootstrap siap pakai untuk membuat navbars, labels, dropdowns, panels, dll. Lihat di: http://getbootstrap.com/components
- Bagaimana menggunakan komponen? Salin contoh penggunaannya ke dalam aplikasi (view) kita.
- Contoh home.blade.php yang sudah ditambahkan komponen navbar:


Home Page
localhost/blog/public/
Brand
Link
Link
Dropdown
Search
Submit
Link
Dropdown
Action
Another action
Something else here
Separated link
One more separated link
Home Page
Our Home page!
localhost/blog/public/#

Template Blade

- Blade is an official Laravel's templating engine. It's very powerful, but it has very simple syntax.
- We use Blade to build layouts for our Laravel applications.
- Blade view files have **.blade.php** file extension. The **home view** and **other views** that we've been using are Blade templates.
- Usually, we put all Blade templates in **resources/views** directory. The great thing is, we can use plain PHP code in a Blade view.
- All Blade expressions begin with **@**. For example: @section, @if, @for, etc.
- Blade also supports all PHP loops and conditions: **@if, @elseif, @else, @for, @ foreach, @while, etc**. For instance, you can write a if else statement in Blade like this:

Blade vs. PHP

```
@if ($product == 1)
      {!! $product->name !!}
@else
      There is no product!
@endif
```

Sama dengan kode PHP:

```
if ($product ==1) {
      echo $product->name;
} else {
      echo("There is no product!");
}
```

Membuat Layout Master

- The typical web application has a **master layout**. The layout consists header, footer, sidebar, etc.
- Using a master layout, we can easily place one element on every view. For instance, we can use the same header and footer for all pages.
- It helps to make our code look clearer and save our time a lot.
- To get started, we will create a master layout called **master.blade.php**

master.blade.php

```
<html>
<head>
	<title> @yield('title') </title>
	<link rel="stylesheet" href="css/bootstrap.min.css">
	<link rel="stylesheet" href="css/bootstrap-theme.min.css">
	<script src="js/jquery-3.2.1.min.js"></script>
	<script src="js/bootstrap.min.js"></script>
</head>
<body>
	@include('shared.navbar')
	@yield('content')
</body>
</html>
```

master.blade.php

- Mari kita lihat baris demi baris kode:

<title> @yield('title') **</title>**

- Instead of putting a title here, we use **@yield** directive to get the title from another section.

@include('shared.navbar')

- We use **@include** directive to include other Blade views. You may notice that we've embed a view called **navbar** here.
- However, we don't have the navbar view yet, let's create a **shared** directory and put **navbar.blade.php** there.

@yield('content')

- As you see it's really convenient for us to not display the content of any pages here. We simply use **@yield** directive to embed a section called **content** from other views.

shared/navbar.blade.php

```
<nav class="navbar navbar-default">
<div class="container-fluid">
<!-- Brand and toggle get grouped for better mobile display -->
<div class="navbar-header">
<button type="button" class="navbar-toggle collapsed" data-toggle="c\
ollapse" data-target="#bs-example-navbar-collapse-1">
<span class="sr-only">Toggle navigation</span>
<span class="icon-bar"></span>
<span class="icon-bar"></span>
<span class="icon-bar"></span>
</button>
<a class="navbar-brand" href="#">Brand</a>
</div>
```

```
<!-- Collect the nav links, forms, and other content for toggling -->
<div class="collapse navbar-collapse" id="bs-example-navbar-collapse-1">
<ul class="nav navbar-nav">
<li class="active"><a href="#">Link <span class="sr-only">(curre\
nt)</span></a></li>
<li><a href="#">Link</a></li>
<li class="dropdown">
<a href="#" class="dropdown-toggle" data-toggle="dropdown" r\
ole="button" aria-expanded="false">Dropdown <span class="caret"></span></a>
<ul class="dropdown-menu" role="menu">
<li><a href="#">Action</a></li>
<li><a href="#">Another action</a></li>
<li><a href="#">Something else here</a></li>
<li class="divider"></li>
<li><a href="#">Separated link</a></li>
<li class="divider"></li>
<li><a href="#">One more separated link</a></li>
</ul>
</li>
</ul>
```

```
<form class="navbar-form navbar-left" role="search">
<div class="form-group">
<input type="text" class="form-control" placeholder="Search">
</div>
<button type="submit" class="btn btn-default">Submit</button>
</form>
<ul class="nav navbar-nav navbar-right">
<li><a href="#">Link</a></li>
<li class="dropdown">
<a href="#" class="dropdown-toggle" data-toggle="dropdown" r\
ole="button" aria-expanded="false">Dropdown <span class="caret"></span></a>
<ul class="dropdown-menu" role="menu">
<li><a href="#">Action</a></li>
<li><a href="#">Another action</a></li>
<li><a href="#">Something else here</a></li>
<li class="divider"></li>
<li><a href="#">Separated link</a></li>
</ul>
</li>
</ul>
</div><!-- /.navbar-collapse -->
</div><!-- /.container-fluid -->
</nav>
```

Memanfaatkan Master Layout

- View home akan menjadi turunan dari master

```
@extends('master')
@section('title', 'RAI-2017')
@section('content')
<div class="container">
   <div class="content">
        <div class="title">Situs web RAI-2017</div>
        <div class="quote">Selamat datang di RAI 2017.</div>
   </div>
</div>
@endsection
```

Memanfaatkan Master Layout

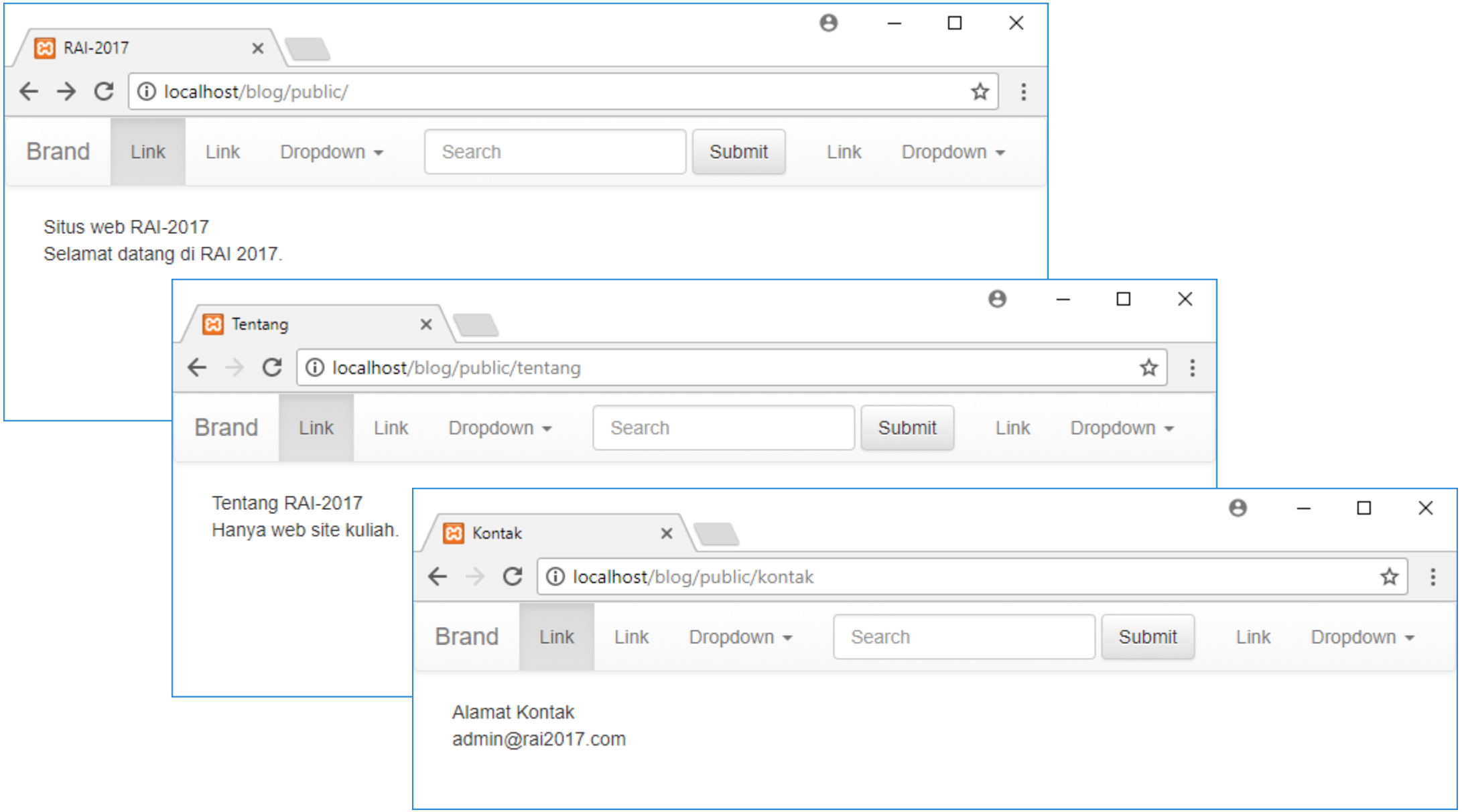
- Kita gunakan direktif **@extends** untuk menurunkan **master layout**
- Untuk mengubah judul dari home page, gunakan direktif **@section**.
- @section('title', 'Home')
- It’s the “short way”. If we have a long content, we can use **@section** and **@endsection** to inject our **content** into the master layout.

```
<div class="container">
    <div class="content">
            <div class="title">Situs web RAI-2017</div>
            <div class="quote">Selamat datang di RAI 2017.</div>
    </div>
</div>
```

View Tentang dan Kontak

```
@extends('master')
@section('title', 'Tentang')
@section('content')
    <div class="container">
        <div class="content">
            <div class="title">Tentang RAI-2017</div>
            <div class="quote">Hanya web site kuliah.</div>
        </div>
    </div>
@endsection
```

```
@extends('master')
@section('title', 'Kontak')
@section('content')
    <div class="container">
        <div class="content">
            <div class="title">Alamat Kontak</div>
            <div class="quote">admin@rai2017.com</div>
        </div>
    </div>
@endsection
```


RAI-2017
localhost/blog/public/
Brand
Link
Link
Dropdown
Search
Submit
Link
Dropdown
Situs web RAI-2017
Selamat datang di RAI 2017.
Tentang
localhost/blog/public/tentang
Brand
Link
Link
Dropdown
Search
Submit
Link
Dropdown
Tentang RAI-2017
Hanya web site kuliah.
Kontak
localhost/blog/public/kontak
Brand
Link
Link
Dropdown
Search
Submit
Link
Dropdown
Alamat Kontak
admin@rai2017.com

Tema Lain di Bootstrap

- Bootstrap punya banyak tema
- Contoh tema yang popular:
  - http://bootswatch.com
  - http://fezvrasta.github.io/bootstrap-material-design
  - http://designmodo.github.io/Flat-UI

Rangkuman

- Good job! We now have a fully responsive template! We will use this template to build other applications to learn more about Laravel.
- In this chapter, you've learned many things:
    1. You've known about Laravel structure, how Laravel works and where to put the files.
    2. You've learned about Laravel routes.
    3. You've learned Controllers. Now you can be able to create web pages using Controllers.
    4. You've known what Blade is. It's easy to create Blade templates for your next amazing applications.
    5. You've known how to integrate Twitter Bootstrap, CSS, JS and apply different Bootstrap themes.
- Pada kuliah berikutnya kita akan praktekkan (di Lab) cara membuat aplikasi CRUD (Create, Read, Update, Delete) memanfaatkan framework Laravel