



# Laravel controllers routes views

**PHP WebDevelopment 2019**

Милена Томова  
Vratsa Software

<https://vratsasoftware.com/>

# Table of Contents

- Controllers - Views - Routes
- Php artisan
- Plain controllers
- Blade engine
  - Master.blade.php
  - @include
  - @extends
  - @section('section\_name')
  - @yield

# Task

Task

- List the pages of VSS Students` System
- Setup the page structure
- Develop the navigation





# Intro

# Intro

To access the application use the url -

`localhost/path-to-your-project/public`

The script in ...public/index.php is executed and the home page of the application is rendered in the browser.



# Action flow

step 1

- in `app/routes/web.php` -
  - an obligatory route declaration is defined

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

that allows this view to be rendered when the ‘/’ end-point is accessed.

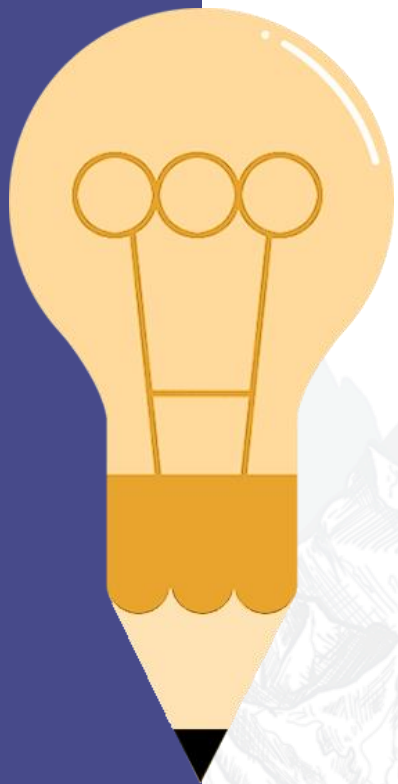




# Action flow

step 2

- in `app/resources/views/welcome.blade.php` -
  - a view file is created, holding the content to be displayed in the browser's window.



# Action flow - REMEMBER

```
return view('welcome');
```

is the

welcome.blade.php file, that resides in  
**resources/view** folder of the project







**Blade engine**

# Blade engine

**Blade** is the simple, yet powerful templating engine provided with Laravel.

Unlike other popular PHP templating engines, **Blade does not restrict** you from **using plain PHP code** in your views.

All Blade views are **compiled into plain PHP** code and **cached until** they are modified.

Blade view files use the **.blade.php** file extension and are typically **stored** in the **resources/views** directory.





# PHP Artisan



# PHP Artisan

**Artisan** is the command-line interface included with Laravel. It provides a number of helpful commands that can assist you while you build your application.

To view a list of all available Artisan commands

```
php artisan list
```

Every command has a "help" screen with full info for command's options and arguments

```
php artisan help command-name
```





# Controllers

# Controllers

Controllers group related request handling logic into a single class.

They are stored in the app/[Http/Controllers](#) directory.

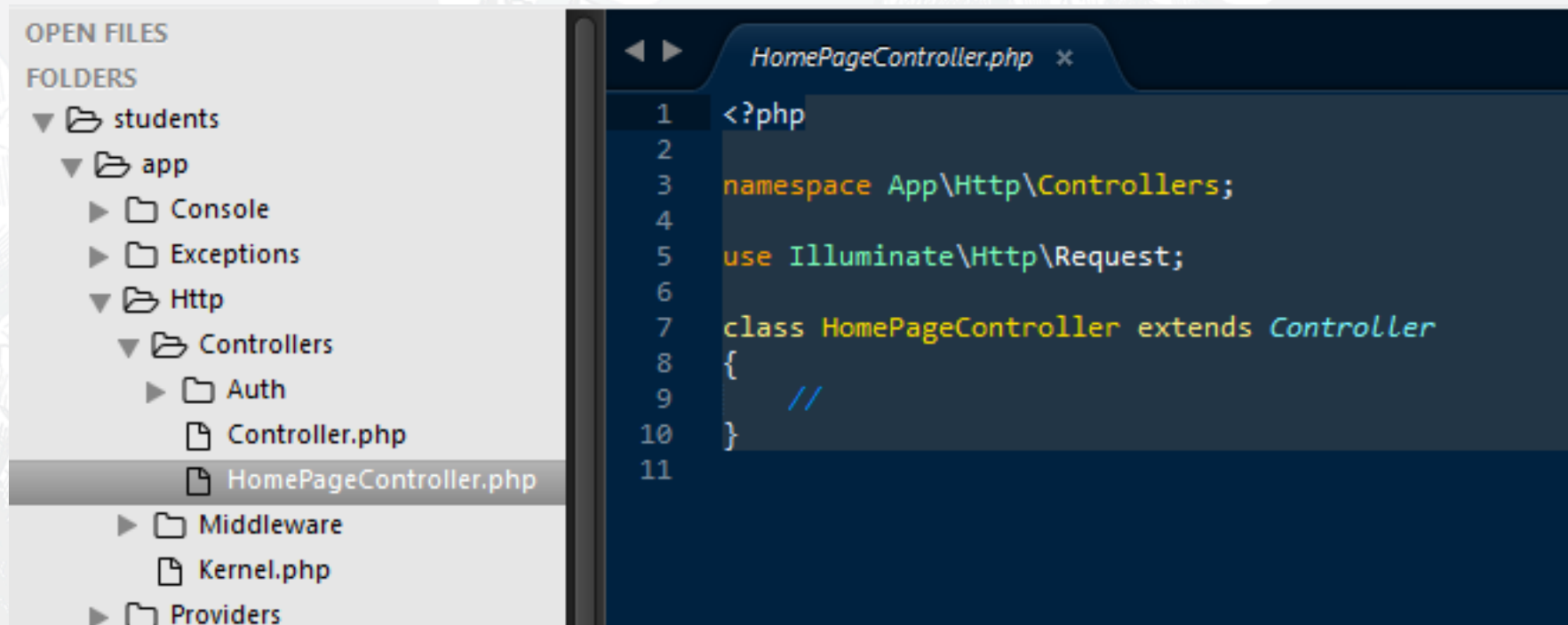




# Controllers

To create a plain controller execute the command

```
php artisan make:controller HomeController
```



# Controllers

First task of the newly created HomeController is to render the home page -

```
public function index() {  
    return view('home_page');  
}
```





# Views



# Views

In *app/resources/views* create

**home\_page.blade.php**

*This file will hold the content that will be displayed in the browser to the user who visits the Home page.*





# Routes

# Routes

In **routes/web.php** we have to allow the rendering of home page to happen when accessing the **'/'** end-point

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

With this declaration  
we allow the execution  
of the **index** method of the **HomeController**.





# Routes

## *Named routes*

```
Route::get('/', 'HomeController@index')->name('home');
```

Then we can easily create navigation links

```
<a href={{ route('home') }}> Home </a>
```

where

```
{{ route('home') }} is instead of <?php echo route('home') ?>
```





**master.blade.php**

# master.blade.php

*When there is repeated code in a number of views we can create a template file.*

*The repeated code is placed within this template file and by using*

*@yield('section\_name')*

*we call the unique lines of code for every view that will extend the template.*

*We can yield styles, js, html etc.*





# master.blade.php

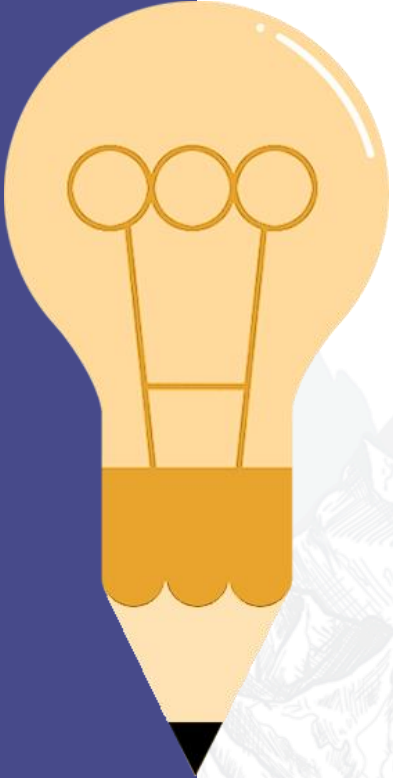
The template files are **usually named**

- *master.blade.php*
- *app.blade.php*
- *admin.blade.php*
- *etc*

and are placed in a layouts folder /if there is more than one template/.



# @extends('master')



*Every view.blade.php **starts** with the  
**@extends('template-name')** declaration,  
if it is not an independent view, but extends a template.*

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4     <meta charset="UTF-8">
5     <title>
6         @yield('title')
7     </title>
8 </head>
9 <body>
10     @yield('content')
11 </body>
12 </html>
```

master.blade.php

```
1 @extends('master')
2
3 @section('title', 'Начална страница')
4
5 @section('content')
6     <ul>
7         <li><a href="{{ route('home') }}">Начална</a></li>
8         <li><a href="#">Регистрация</a></li>
9         <li><a href="{{ route('profile') }}">Профил</a></li>
10        <li><a href="{{ route('homework') }}">Домашни</a></li>
11    </ul>
12 @endsection
13
```

home\_page.blade.php





**Organise Blade files**

# Organise Blade files

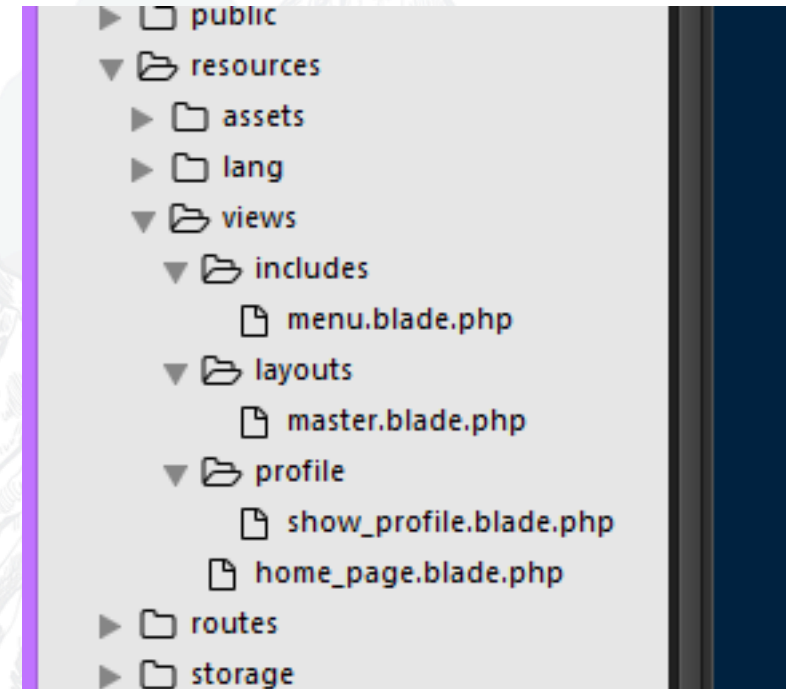
We can separate repeated pieces of codes in separate blade files.

Organise them in includes folders.

And include them where needed by

```
@extends('template-folder.template-name')
```

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



# Organise Blade files

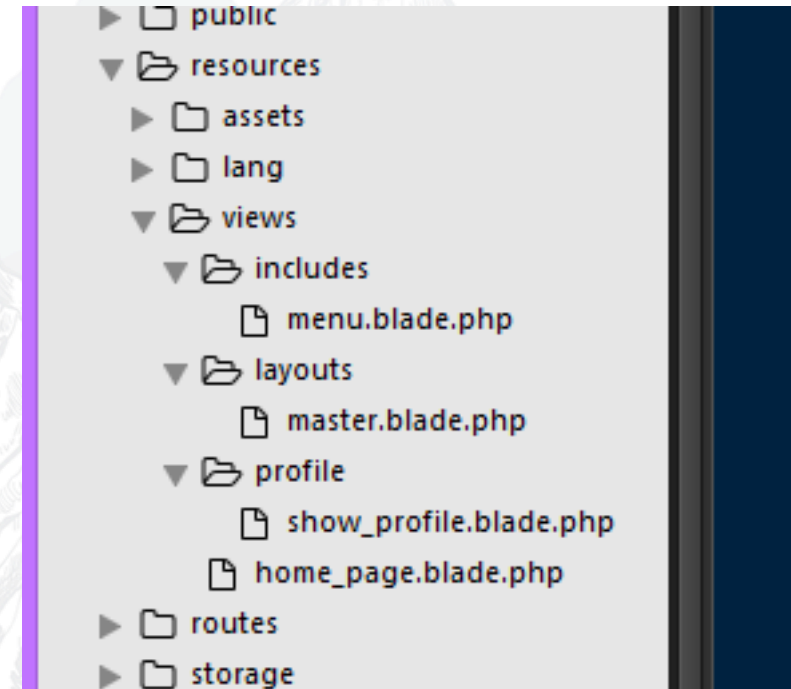
It is very probable that your project will have or will expand to lots of view files.

It is a good practise to organise them in folders.

If a template is in a folder - it is referred by

`@include('folder.filename')`

`@includes('includes.menu')`





# Questions?



Гнездото  
Coworking

Цялостен  
курс по  
програми  
ране

Дизайн  
курс

Курс по  
дигит.  
маркетинг

MindHub



# Partners



**Telerik  
Academy**



**MindHub**

**ПРОМЯНАТА**

# Trainings @ Vratsa Software



- Vratsa Software – High-Quality Education, Profession and Jobs
  - [www.vratsasoftware.com](http://www.vratsasoftware.com)
- The Nest Coworking
  - [www.nest.bg](http://www.nest.bg)
- Vratsa Software @ Facebook
  - [www.fb.com/VratsaSoftware](http://www.fb.com/VratsaSoftware)
- Slack Channel
  - [www.vso.slack.com](http://www.vso.slack.com)

