



Desenvolvimento Web II

Framework Laravel 5

*Conceitos Iniciais / Rotas / Blade
(Menu Principal – SETA)*

Gil Eduardo de Andrade





Framework Laravel

Introdução – Framework

- Um Framework é uma estrutura de código projetada para criação de novas aplicações;
- Pode ser visto, também, como um kit contendo diversas funcionalidades utilizadas na construção de novos softwares;
- Agiliza o processo de desenvolvimento por já trazer prontas, funcionalidades básicas como acesso a banco de dados, *templates*, rotas, etc.





Framework Laravel

Introdução – Laravel

- A primeira versão do framework Laravel vou lançada em 2011, ela foi criada totalmente do zero e fornecia:
 - ORM (Mapeamento Objeto Relacional) personalizado, denominado Eloquent;
 - Sistema de módulos para extensão e *helpers* para formulários;
 - Validação, autenticação, etc;





Framework Laravel

Laravel 4

- A versão 4 foi reescrita por inteiro, a partir do zero, onde um grupo de componentes foi construído e denominado *Illuminate* (*motor da base de dados, vem junto com Eloquent*);
- Ela também introduziu componentes de e-mails, *facades* (interface estáticas para acesso aos recursos do Framework) e *database seeding*;





Framework Laravel

Laravel 5

- Apresentou uma estrutura de diretórios renovada, com adição e aperfeiçoamento das suas funcionalidades;
- Dentre elas tem-se: *Socialite* para autenticação em mídias sociais e *Elixir* para compilação de *assets* (recursos do projeto);





Framework Laravel

Instalação do Laravel (Linux Fedora 26 – git)

- Instalando *git* (sistema de controle de versão):
 - `# dnf install git` (*apt-get install git*)
 - <https://git-scm.com/download/linux>
- Instalando *Composer* (gerenciador de dependências):
 - <https://getcomposer.org>
 - Requisitos: HP 5.3.2 ou Superior
 - Dependências:
`# sudo dnf install php php-common php-cli php-pdo php-mbstring php-zip php-xml`
`# sudo systemctl restart httpd`





Framework Laravel

Instalação do Laravel

- Instalando *Composer* (gerenciador de dependências):
 - `# curl -sS https://getcomposer.org/installer | php`
 - `# sudo mv composer.phar /usr/local/bin/composer`
 - `# chmod +x /usr/local/bin/composer` (permissão execução)
 - `# composer -V` (visualizar versão, instalação OK)





Framework Laravel

Instalação do Laravel

- Instalando *Laravel* (Framework):
 - www.laravel.com
 - *# composer global require "laravel/installer"*
 - *# echo 'export PATH="\$PATH:\$HOME/.config/composer/vendor/bin"' >> ~/.bashrc*
(para o executável do Laravel ser encontrado pelo SO quando usarmos a linha de comando)
 - *Reiniciar o terminal (fechar e abrir novamente)*





Framework Laravel

Criando uma nova aplicação Laravel

- *# laravel new NomeProjeto*
- *# php artisan serve (inicia servidor artisan)*

(artisan é uma ferramenta de linha de comando do Laravel instala automaticamente quando criamos um novo projeto. Ela possibilita que várias classes, necessárias ao longo do desenvolvimento da aplicação, sejam criadas)





Servidor Apache

Definindo um host virtual para Aplicação

- Criar/Editar arquivos de configuração:
sudo gedit /etc/httpd/conf.d/default.conf &
(dominio padrao: localhost)
<VirtualHost _default_:80>
ServerName localhost
DocumentRoot /var/www/html
</VirtualHost>





Servidor Apache

Definindo um host virtual para Aplicação

- Criar/Editar arquivos de configuração:

sudo gedit /etc/httpd/conf.d/**nome_app**.conf &

(especificando um host virtual para aplicação que está sendo criada)

```
<VirtualHost *:80>
```

```
    ServerName nome_app.laravel
```

```
    DocumentRoot /var/www/html/laravel/nome_app/public
```

```
    <Directory /var/www/html/laravel/nome_app/public>
```

```
        Options -Indexes +FollowSymLinks +MultiViews
```

```
        AllowOverride All
```

```
        Order allow,deny
```

```
        Allow from all
```

```
        Require all granted
```

```
    </Directory>
```

```
</VirtualHost>
```





Servidor Apache

Definindo um host virtual para Aplicação

- Atualizar/Adicionar ao arquivo de hosts / definir hosts criados:

```
# sudo gedit /etc/hosts &
```

(linha que deve ser adicionada)

```
127.0.0.1      nome_app.laravel
```

```
# apachectl restart
```

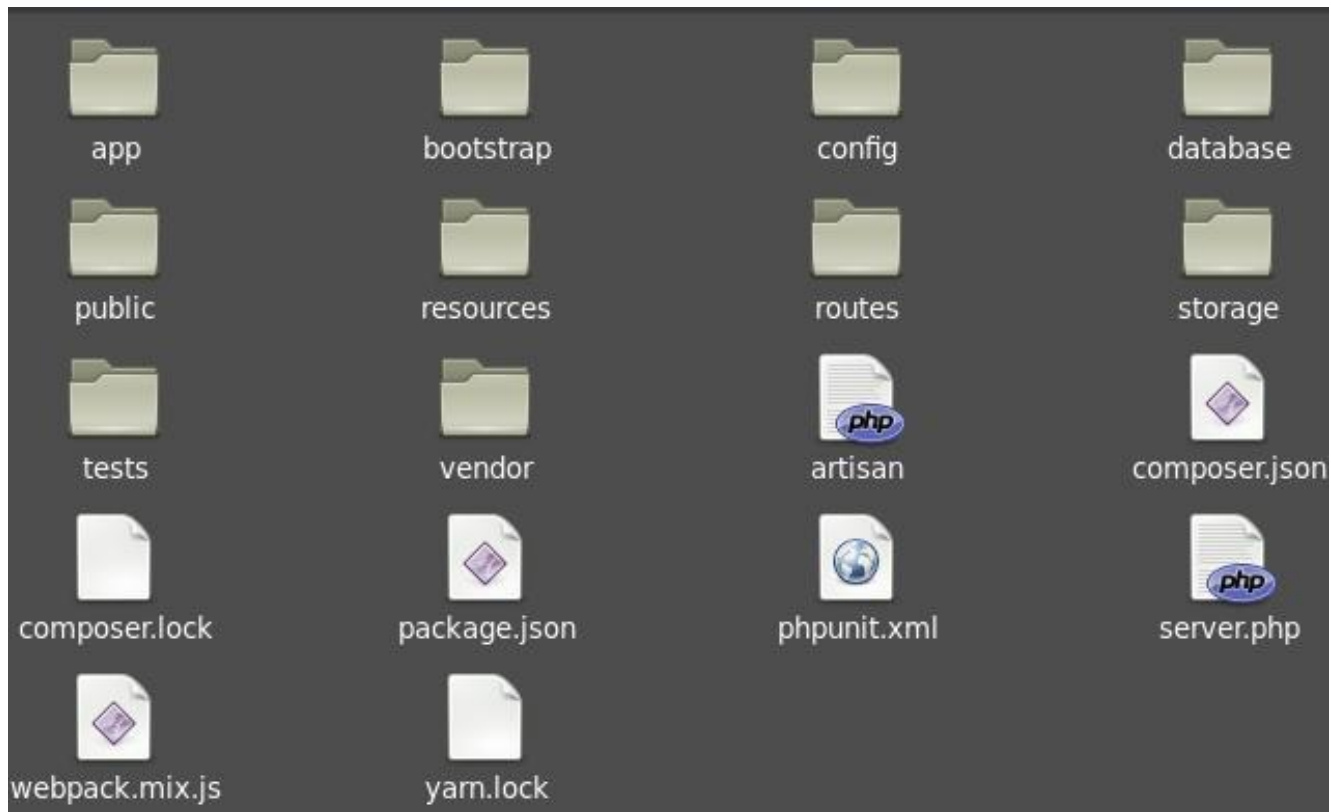
(necessário reiniciar o servidor apache)





Framework Laravel

Estrutura Básica de Diretórios da Aplicação





Framework Laravel

Principais Diretórios e Arquivos

- **Diretório “public”:**
 - .htaccess → implementa as URLs amigáveis – baseado na chamada (url) identifica qual é a controller responsáveis pela mesma;
 - index.php → efetua a chamada da controller default;
 - .env → arquivo de configuração de recursos, contém, por exemplo, as informações para conexão com o banco de dados, envio de e-mail (SMTP), etc;





Framework Laravel

Principais Diretórios e Arquivos

- **Diretório “app”:**
 - Diretório mais importante da aplicação se considerarmos o ponto de vista do desenvolvedor;
 - Nele encontra-se a pasta ***Http/Controllers*** onde estão armazenadas todas as classes de controle da aplicação;
 - Em algumas versões do Laravel a pasta ***Http*** contém o arquivo ***routes.php***, que contém as rotas da aplicação;





Framework Laravel

Principais Diretórios e Arquivos

- **Diretório “database”:**
 - Diretório onde ficam os arquivos para criação das tabelas, vínculo com as classes de modelo e povoamento;
 - **migration**: permite criar tabelas do banco;
 - **seeders** : permite povoar as tabelas do banco para testar aplicação;





Framework Laravel

Principais Diretórios e Arquivos

- **Diretório “resources”:**
 - Como o próprio nome diz é o diretório que contém alguns dos principais recursos da aplicação;
 - *diretório **assets***: configuração e definição do *CSS*, *javascript* e imagens;
 - *diretório **views***: onde são hospedadas as páginas de visualização da aplicação (*welcome.blade.php* é chamado por padrão, similar a um *index.php*);





Framework Laravel

Rotas: *(routes/web.php)*

- No arquivo ***web.php*** (ou routes.php em versões anteriores do Laravel) são definidas as rotas da aplicação, ou seja, quais páginas de visualização ou classes de controle devem ser invocadas quando uma determinada rota (URL) é requisitada;



Framework Laravel

Rota Inicial:

(os exemplos a seguir baseiam-se numa aplicação Laravel criada com nome **aula05**)
(além disso um virtual host foi configurado com nome **aula05.laravel**)

- A seguir é apresentada a página inicial da aplicação, logo após sua criação:



Resultado da chamada
da URL **aula05.laravel**
no navegador.

Framework Laravel

Rota Inicial:

(compreendendo o funcionamento)

```
aula05.conf x
1 <VirtualHost *:80>
2     ServerName aula05.laravel
3     DocumentRoot /var/www/html/laravel/aula05/public
4     <Directory /var/www/html/laravel/aula05/public>
5         Options -Indexes +FollowSymLinks +MultiViews
6         AllowOverride All
7         Order allow,deny
8         Allow from all
9         Require all granted
10    </Directory>
11 </VirtualHost>
12
```

O host virtual criado (arquivo `aula05.conf`) configurou um servidor com nome **aula05.laravel** apontando para pasta **public/** da aplicação. Nela o arquivo `index.php` carregar todos os módulos necessários para execução da aplicação e efetua, por padrão, a leitura das rotas em **web.php**

```
web.php x
1 <?php
2
3 Route::get('/', function () {
4     return view('welcome');
5 });
6
```

O única rota, já definida por padrão quando o projeto Laravel é criado, é a rota para URL principal da aplicação (/). Ela pode ser vista no arquivo **web.php** e aponta para a **view welcome**, localizada no arquivo **welcome.blade.php**

Framework Laravel

Modificando e Testando a Rota Inicial:

(retornando uma mensagem HTML ao invés da view Welcome)

```
web.php
1 <?php
2
3 Route::get('/', function () {
4
5     $msg = "<h1>Rotas Laravel</h1>";
6     $msg .= "<h3>Página Inicial</h3>";
7     return $msg;
8 });
```

aula05.laravel

← → ↻ ⓘ aula05.laravel/

Rotas Laravel

Página Inicial

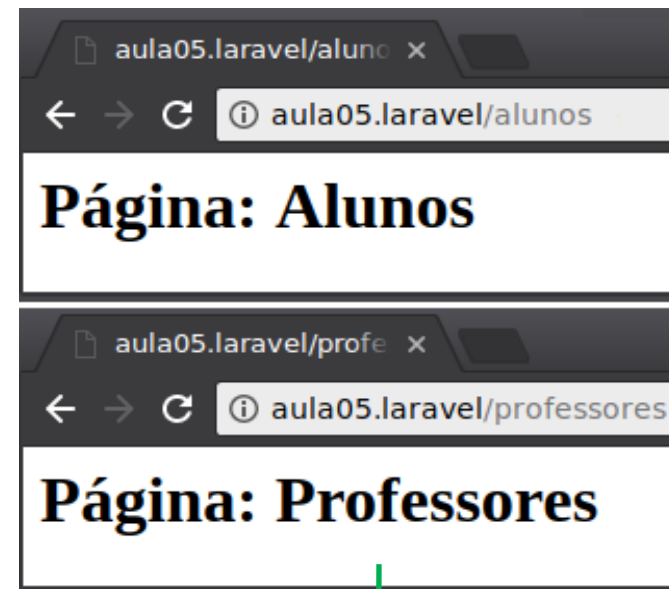
Ao mudar a codificação da rota para URL principal, dentro do arquivo **web.php**, a resposta para requisição **aula05.laravel** é alterada para mensagem especificada.

Framework Laravel

Criando e Testando Novas Rotas:

(Novas rotas (requisições via URL) podem ser especificadas e tratadas no arquivo **web.php**)

```
web.php
1 <?php
2
3 Route::get('/', function () {
4
5     $msg = "<h1>Rotas Laravel</h1>";
6     $msg .= "<h3>Página Inicial</h3>";
7     return $msg;
8 });
9
10 Route::get('/alunos', function () {
11     return "<h1>Página: Alunos</h1>";
12 });
13
14 Route::get('/professores', function () {
15     return "<h1>Página: Professores</h1>";
16 });
```



Respostas para as novas rotas criadas:
/alunos e /professores.



Framework Laravel

View – Blade:

- Linguagens de *backend* como PHP funcionam relativamente bem com *template*, contudo é muito cansativo e deselegante utilizar a tag `<?php inline` ao longo de toda uma view HTML, por exemplo;
- Por isso, o Laravel disponibiliza o **Blade** como uma *engine* (motor) de *template* (modelo, layout);





Framework Laravel

View – Blade:

- O ***Blade*** possui um conjunto relativamente grande de funcionalidades, desde a manipulação e apresentação de variáveis PHP até estruturas mais complexas de programação como condição e laço de repetição;
- Contudo, nessa aula iremos abordar conceitos mais básicos vinculados a criação de *views* com extensão *.blade*;





Framework Laravel

Criando Views com extensão *“.blade”*:

(Arquivo criado no diretório */resources/views/principal.blade.php*)

```
principal.blade.php x
1 <!DOCTYPE html>
2 <html lang="en">
3   <head>
4     <title>Aula05 - Laravel</title>
5   </head>
6   <body>
7     <h1>Conceitos Iniciais / Rotas / Views/ Blade</h1>
8
9     <div>
10       @yield('cabecalho')
11     </div>
12
13     <div>
14       @yield('conteudo')
15     </div>
16
17     <div>
18       <b>&copy;2018 &raquo; Gil Eduardo de Andrade</b>
19     </div>
20   </body>
21 </html>
```

Define uma seção “*cabecalho*” que será preenchida por outra *view* que herdará as característica da *principal.blade.php*

Define uma seção “*conteudo*” que será preenchida por outra *view* que herdará as característica da *principal.blade.php*



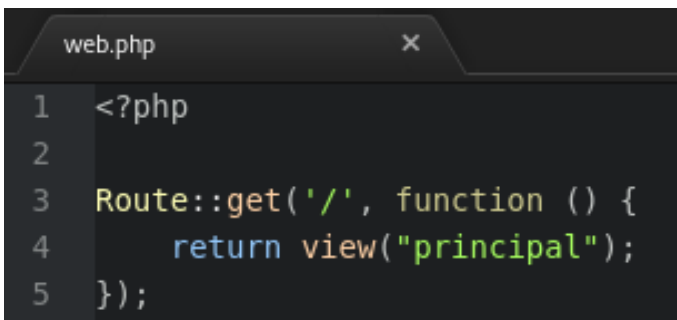
Framework Laravel

Invocando a views “principal.blade.php”:

(A rota principal (URL “/”) foi modificada no arquivo web.php)



Layout padrão criado na view “principal.blade.php”

A screenshot of a code editor showing the 'web.php' file. The code is as follows:

```
1 <?php
2
3 Route::get('/', function () {
4     return view("principal");
5 });
```

An arrow points from the right side of the code block towards a text box on the right.

Modificação no arquivo de rotas “web.php” para invocar a view “principal.blade.php” quando a rota principal (página inicial) da aplicação é requisitada via URL.



Framework Laravel

Criando uma view que herda “*principal.blade.php*”:

(Arquivo criado no diretório `/resources/views/alunos.blade.php`)

```
alunos.blade.php x
1  @extends('principal')
2
3  @section('cabecalho')
4      <h5>(Seção Blade Cabeçalho)</h5>
5      <h2>Alunos Cadastrados</h2>
6  @stop
7
8  @section('conteudo')
9      <h5>(Seção Blade Conteúdo)</h5>
10     <h3>Gil Eduardo</h3>
11     <h3>João Paulo</h3>
12     <br>
13     @stop
```

Herda o Layout padrão definido na view do arquivo “*principal.blade.php*”

Implementa/adiciona o conteúdo da seção “cabecalho” definida em “*principal.blade.php*”

Implementa/adiciona o conteúdo da seção “conteudo” definida em “*principal.blade.php*”

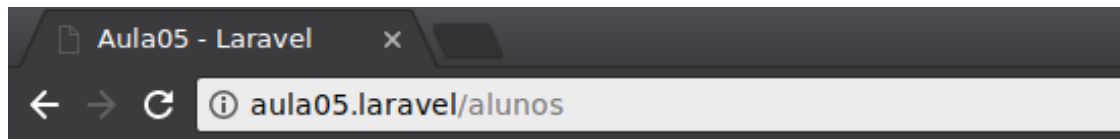




Framework Laravel

Invocando a view “alunos.blade.php”:

(A rota “/alunos” foi modificada no arquivo **web.php** para apresentar o arquivo **alunos.blade.php**)



Conceitos Iniciais / Rotas / Views/ Blade

(Seção Blade Cabeçalho)

Alunos Cadastrados

(Seção Blade Conteúdo)

Gil Eduardo

João Paulo

©2018 » Gil Eduardo de Andrade

```
web.php
1 <?php
2
3 Route::get('/', function () {
4     return view("principal");
5 });
6
7 Route::get('/alunos', function () {
8     return view("alunos");
9 });
```

No arquivo de rotas configura a requisição “/alunos” para invocar a view “alunos.blade.php”. Ao requisitar a URL “aula05.laravel/alunos” temp-se como resultado a figura ao lado.





Framework Laravel

Criando a view: “*professores.blade.php*”:

(Arquivo criado no diretório `/resources/views/professores.blade.php`)

```
professores.blade.php x
1  @extends('principal')
2
3  @section('cabecalho')
4      <h5>(Seção Blade Cabeçalho)</h5>
5      <h2>Professores Cadastrados</h2>
6  @stop
7
8  @section('conteudo')
9      <h5>(Seção Blade Conteúdo)</h5>
10     <h3>Diego Hoss</h3>
11     <h3>Valério Brusamolín</h3>
12     <br>
13     @stop
```

Herda o Layout padrão definido na view do arquivo “*principal.blade.php*”

Implementa/adiciona o conteúdo da seção “cabecalho” definida em “*principal.blade.php*”

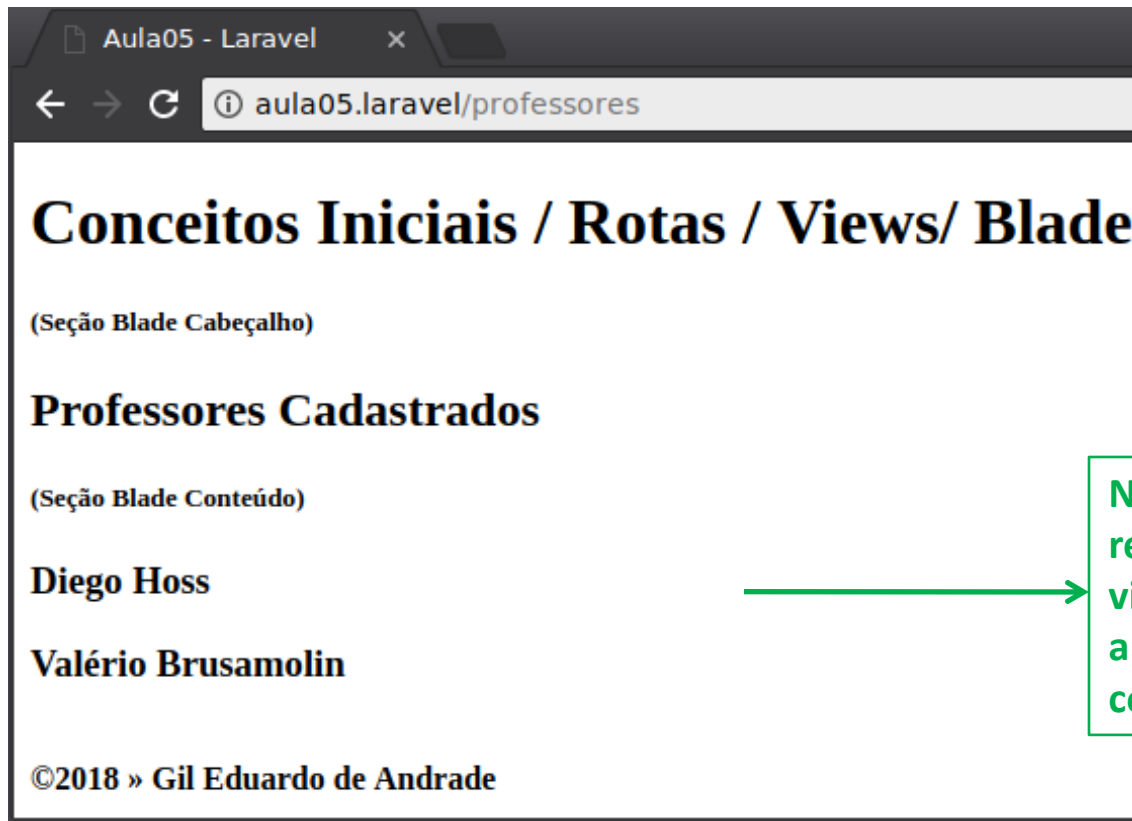
Implementa/adiciona o conteúdo da seção “conteudo” definida em “*principal.blade.php*”



Framework Laravel

Invocando a view “alunos.blade.php”:

(A rota “/professores” foi modificada em **web.php** para apresentar o arquivo **professores.blade.php**)



The screenshot shows the 'web.php' file in a code editor. The code defines two routes: a GET route for '/alunos' that returns the 'alunos' view, and a GET route for '/professores' that returns the 'professores' view. A green arrow points from the text box on the right to the route definition for '/professores'.

```
1 <?php
7 Route::get('/alunos', function () {
8     return view("alunos");
9 });
10
11 Route::get('/professores', function () {
12     return view("professores");
13 });
```

No arquivo de rotas configura a requisição “/alunos” para invocar a view “alunos.blade.php”. Ao requisitar a URL “aula05.laravel/alunos” temp-se como resultado a figura ao lado.



Framework Laravel

Controllers:

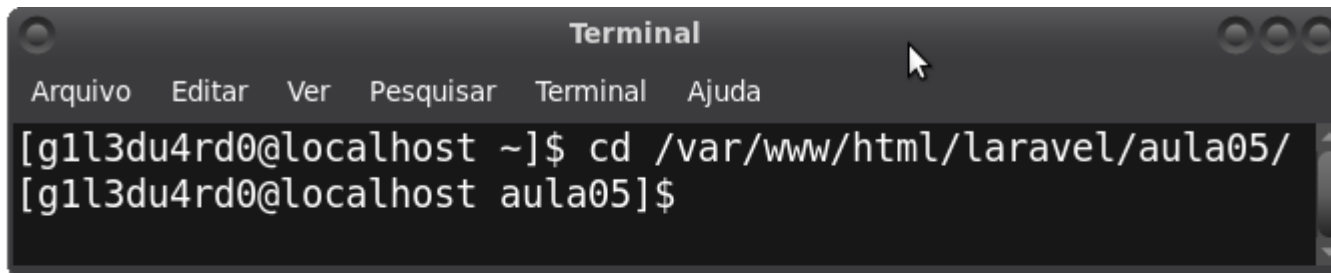
- As classes de Controle são responsáveis pela lógica da aplicação e por costumam tratar as requisições recebidas pelo arquivo de rotas;
- Essas classes estendem, por padrão, as funcionalidades da super classe “***Controller***”, e localizam-se em “***app/Http/Controllers***”;



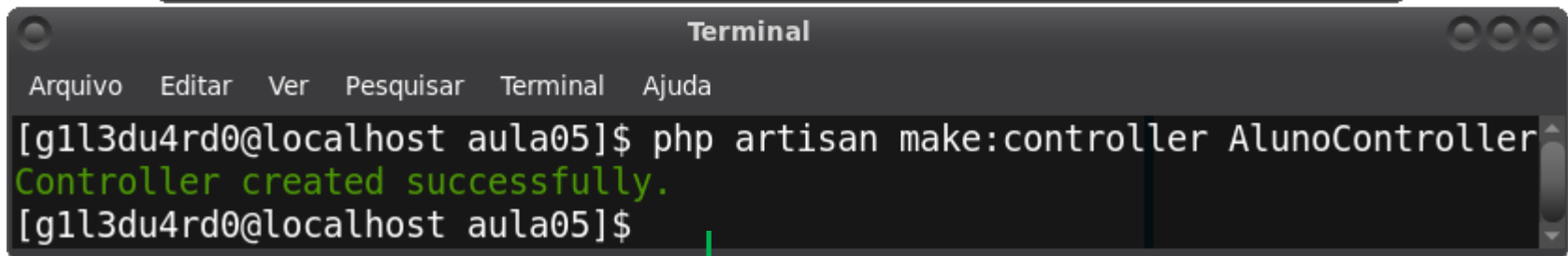
Framework Laravel

Criando Controllers via artisan:

(Esse procedimento deve ser feito via terminal, dentro da pasta da aplicação)



```
Terminal
Arquivo  Editar  Ver  Pesquisar  Terminal  Ajuda
[g1l3du4rd0@localhost ~]$ cd /var/www/html/laravel/aula05/
[g1l3du4rd0@localhost aula05]$
```



```
Terminal
Arquivo  Editar  Ver  Pesquisar  Terminal  Ajuda
[g1l3du4rd0@localhost aula05]$ php artisan make:controller AlunoController
Controller created successfully.
[g1l3du4rd0@localhost aula05]$
```



Ao criarmos um projeto em Laravel o *artisan* é instalado automaticamente e fica disponível na raiz da aplicação. Como é uma ferramenta de linha de comando permite que classes de Controle sejam criadas via terminal e armazenadas em “*app/Http/Controllers*”.

Framework Laravel

Redirecionando Requisições para Controllers:

*(A rota “/alunos” foi modificada em **web.php** para invocar um método da **AlunoController.php**)*

```
AlunoController.php x
1 <?php
2
3 namespace App\Http\Controllers;
4
5 use Illuminate\Http\Request;
6
7 class AlunoController extends Controller {
8
9     public function listar() {
10         return view('alunos');
11     }
12 }
```

O método “listar” da classe de controle “AlunoController.php” invoca a view “alunos.blade.php”.

```
web.php x
1 <?php
2
3 Route::get('/', function () {
4     return view("principal");
5 });
6
7 Route::get('/alunos', 'AlunoController@listar');
```

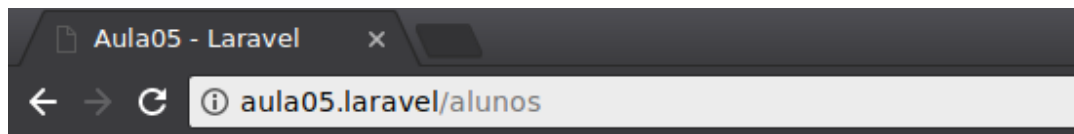
A rota relativa a requisição “/alunos” é direcionada para o método “listar” da classe de controle “AlunoController.php”.



Framework Laravel

Redirecionando Requisições para Controllers:

(O resultado é o mesmo visto anteriormente, contudo aplicando, agora, o conceito de Controllers)



Conceitos Iniciais / Rotas / Views/ Blade

(Seção Blade Cabeçalho)

Alunos Cadastrados

(Seção Blade Conteúdo)

Gil Eduardo

João Paulo

©2018 » Gil Eduardo de Andrade

A apresentação da *view* ocorre da mesma forma como visto anteriormente, quando esta era invocada diretamente dentro do arquivo de rotas. Contudo, dentro do método da classe de controle é possível implementar algumas lógicas que serão necessárias futuramente, como passagem e recebimento de parâmetros, filtro de dados, entre outros. O mesmo pode ser feito para “*professores*”.



Framework Laravel

Criando um Menu Principal:

(Para esse procedimento foi criada uma nova view denominada “main.blade.php”)

```
main.blade.php x
1 @extends('principal')
2
3 @section('cabecalho')
4     
5     &nbsp;<h2>Menu Principal</h2><br>
6 @stop
7
8 @section('conteudo')
9     <a href="/alunos"></a>
10    <a href="/professores"></a>
11    <br><br>
12 @stop
```

Menu Principal da aplicação, para carregamento das imagens utilizamos a função `url()` do Blade, que retorna o caminho para a raiz da aplicação, diretório “public”, como configurado no host virtual. Observe os links “/alunos” e “/professores”.



Framework Laravel

Criando um Menu Principal:

(Resultado da criação do menu principal, a view “main.blade.php”)





Framework Laravel

Adaptações no Blade principal:

*(Um link para retornar a **main.blade.php** foi criado na **alunos** e **professores.blade.php**)*

```
alunos.blade.php  x
1  @extends('principal')
2
3  <a href="{{ url('/') }}">Home</a>
4  @section('cabecalho')
5      <h5>(Seção Blade Cabeçalho)</h5>
6      <h2>Alunos Cadastrados</h2>
7  @stop
8
9  @section('conteudo')
10 <h5>(Seção Blade Conteúdo)</h5>
11 <h3>Gil Eduardo</h3>
12 <h3>João Paulo</h3>
13 <br>
14 @stop
```

```
professores.blade.php  x
1  @extends('principal')
2
3  <a href="{{ url('/') }}">Home</a>
4  @section('cabecalho')
5      <h5>(Seção Blade Cabeçalho)</h5>
6      <h2>Professores Cadastrados</h2>
7  @stop
8
9  @section('conteudo')
10 <h5>(Seção Blade Conteúdo)</h5>
11 <h3>Diego Hoss</h3>
12 <h3>Valério Brusamolin</h3>
13 <br>
14 @stop
```

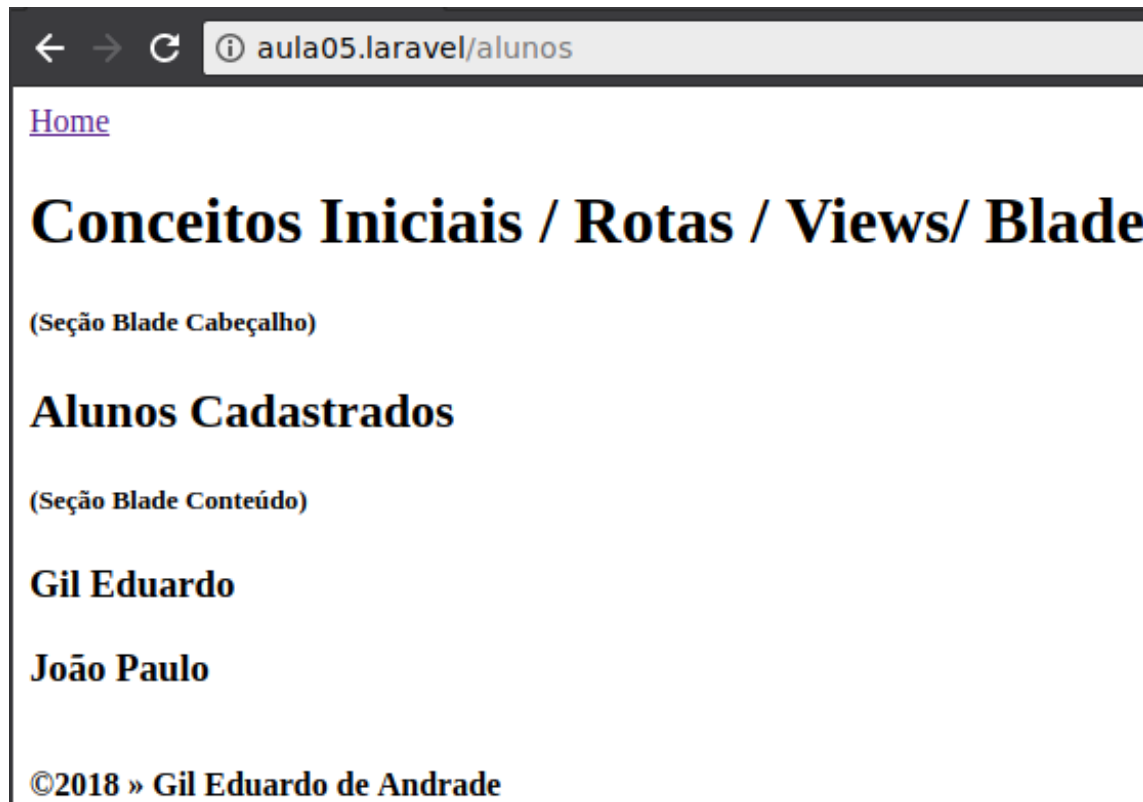




Framework Laravel

Adaptações no Blade principal:

*(Resultado da criação do link de retorno nas views **alunos** e **professores.blade.php**)*





Cirando uma Aplicação Laravel Rapidamente

Composer e Artisan

Composer.phar / php artisan serve

Gil Eduardo de Andrade





Composer e Artisan Serve

Baixando e instalando “composer.phar”:

(fonte: <https://getcomposer.org/download/>)

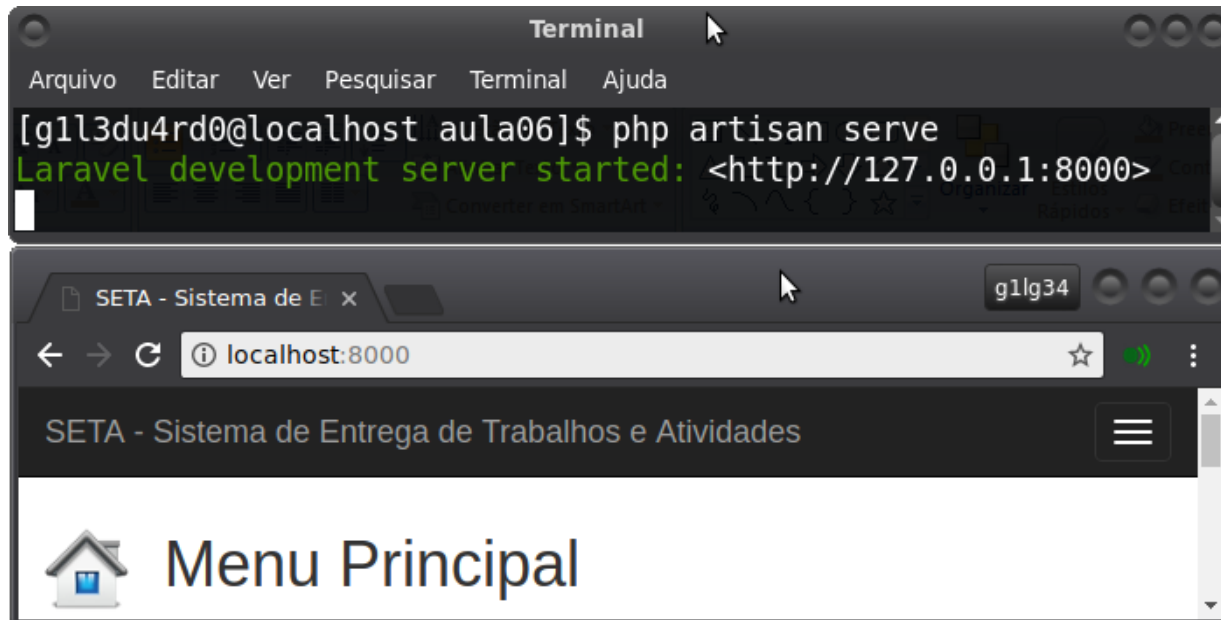
- `php -r "copy('https://getcomposer.org/installer', 'composer-setup.php');"`
- `php -r "if (hash_file('SHA384', 'composer-setup.php')
=== '544e09ee996cdf60ece3804abc52599c22b1f40f4323403c44d44fdadd5864
75ca9813a858088ffbc1f233e9b180f061') { echo 'Installer verified'; } else {
echo 'Installer corrupt'; unlink('composer-setup.php'); } echo PHP_EOL;"`
- `php composer-setup.php`
- `php -r "unlink('composer-setup.php');"`



Composer e Artisan Serve

Criando projeto Laravel via *composer.phar*:

- `./composer.phar create-project laravel/laravel aula 5.5 --prefer-dist`
- `php artisan serve`





Conceitos Iniciais

Exemplos Utilizados no Documento

http://www.gileduardo.com.br/ifpr/dwii/downloads/dwii_exdoc05.zip

Código-fonte da Aplicação SETA

http://www.gileduardo.com.br/ifpr/dwii/downloads/dwii_dica05.zip

Exercício sobre o Conteúdo

http://www.gileduardo.com.br/ifpr/dwii/downloads/dwii_pratica05.pdf

