1. O que é o Laravel

Ele é um framework PHP, um dos mais utilizados hoje em dia.

O Laravel se destaca de outros frameworks principalmente pela rápida evolução que tem.
Hoje ele está na versão 5.7.

Sempre que possível vale a pena conferir o site laravel.com.
Outra vantagem desse framework é a sua documentação: laravel.com/docs.

É uma ótima prática conferir a documentação de qualquer plugin, programa ou biblioteca que deseje utilizar.
Elas sempre mostram o passo a passo para começar, os requisitos mínimos e o caminho das pedras para utilizar a sintaxe de certos comandos as biblioteca que o framework carrega.

O Laravel possui uma alta abstração de código.
Isso quer dizer que constantemente vamos perceber que será preciso digitar menos para fazer a mesma coisa do que no PHP plano, por exemplo.
Também vai forçar o programador a reutilizar muita parte do código (o que é ótimo).

A sintaxe é bastante clean.
O seu código no final do dia fica lindo!
Página principal do Laravel: Love beautiful code? We do too.

Ele é totalmente orientado a objetos e utiliza o conceito MVC (model, view, controller).

Como evolui bastante você vai poder contar com um suporte da comunidade bastante ativa.

Todos os frameworks possuem alguma documentação, mas a do Laravel eu gosto bastante por ser bastante organizada, fácil de entender e direta.
Você vai conseguir encontrar 80% das soluções dos problemas que tiver aqui.
E quando não encontrar basta pesquisar no Google.

Tem bastante programador utilizando esse framework, então certamente já passaram pelo problema que está procurando uma solução.

2. Instalação do Composer

O Composer: https://getcomposer.org/download/

Gerenciador de dependências, que o Laravel utiliza para garantir que todas as bibliotecas estejam presentes e atualizadas.
Vamos utilizar o composer algumas vezes durante o desenvolvimento do projeto.
Mas a primeira vai ser agora, vamos instalar juntos por aqui?

Basta acessar o Shell (do XAMP) ou o Terminal (do MacOS/Linux) copiar e colar linha por linha:

```
php -r "copy('https://getcomposer.org/installer', 'composer-setup.php');"

php -r "if (hash_file('SHA384', 'composer-setup.php') ===
'93b54496392c062774670ac18b134c3b3a95e5a5e5c8f1a9f115f203b75bf9a129d5daa8ba6a13e2cc8a1da0806388a8'
) { echo 'Installer verified'; } else { echo 'Installer corrupt'; unlink('composer-setup.php'); }
echo PHP_EOL;"

php composer-setup.php

php -r "unlink('composer-setup.php');"
```

Agora o comando muda dependendo se você estiver utilizando Windows ou MacOS/Linux:

Para Windows: `echo @php "%~dp0composer.phar" %*>composer.bat`
Para MaxOS/Linux: `mv composer.phar /usr/local/bin/composer`

No final, para ver se o Composer foi instalado com sucesso basta digitar:

```
composer -V
```

Se tudo deu certo você vai receber de volta algo como:

```
Composer version 1.0.0 2016-01-10 20:34:53
```

3. Instalação do Laravel

Agora sim!

Com o Composer devidamente instalado podemos fazer a instalação do Laravel e de todos os arquivos que ele precisa para fazer seu projeto funcionar.

Basta agora rodar o seguinte comando:

```
composer create-project laravel/laravel MeuProjeto
```

O composer vai fazer o download de tudo que o Laravel precisa para dentro de um diretório como mesmo nome do projeto que escolheu.
Neste caso a pasta vai se chamar "`MeuProjeto`".

4. **Executando o Laravel**

Por fim, vamos agora visualizar a primeira página do Laravel funcionando no navegador.

Nessas aulas não vamos utilizar o Apache como nosso servidor web.
Vamos utilizar um "mini" servidor web que já vem com o nosso framework.

Para isso basta rodar o comando:

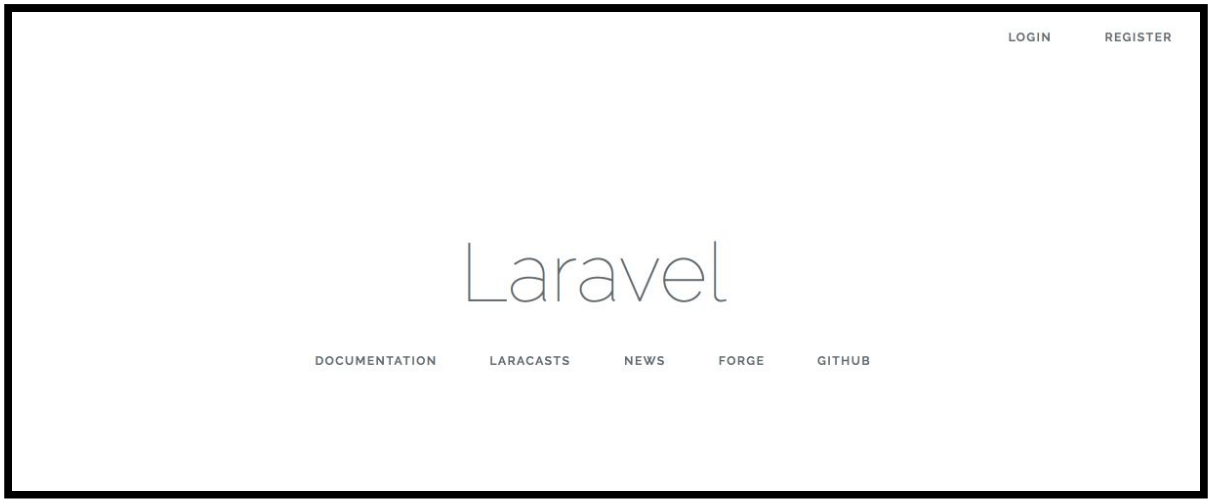
```
php artisan serve
```

Você vai receber uma resposta mais ou menos assim:

```
Laravel development server started: <http://127.0.0.1:8000>
```

Pronto!
Se tudo der certo basta ir até o seu navegador e navegar para: http://localhost:8000

Quando a página de "bem vindo" for exibida significa que o Laravel está instalado e pronto para você começar a desenvolver seu código.



5. **Estrutura de Diretórios do Laravel**

   Aquele diretório criado automaticamente é o seu projeto agora.

   Dentro dele vamos sempre perceber a existência destes diretórios:

   - app
   - config
   - database
   - public
   - resources
   - routes
   - storage
   - tests
   - vendor

   Sempre que possível, não esqueçam de subir o conteúdo da aula no seu repositório do GitHub.

   Vamos aprender mais sobre estes diretórios e para que eles servem nas próximas aulas, mas agora vamos nos concentrar no diretório "routes".

6. **As Rotas** (routes/web.php)

   Tudo no Laravel começa com uma rota.

   As rotas organizam em um único arquivo todos os caminhos (ou URLs) do seu site.

   Diferente do PHP que já conhecemos, que cada arquivo é um endereço do seu site, no Laravel o endereço é configurado apenas nas rotas.
   É exatamente como um índice de todos os recursos do seu projeto que os usuários e outros programas poderão acessar.

   Toda instalação nova do Laravel possui a rota padrão representada apenas pela barra "/".
   Ela é o que mais se aproxima do arquivo "index.php" que estávamos utilizando até então.

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

   Neste exemplo, toda vez que a rota "/" for acessada pelo usuário o Laravel vai exibir uma "view" chamada "welcome".
   Trata-se exatamente daquela página que visualizamos quando acessamos o endereço http://localhost:8000 no navegador.

Para criar uma nova "página" no seu site basta criar uma nova rota aqui.

Também é possível criar uma rota que receba um parâmetro através da URL.
Assim:

```
Route::get('/filme/{id}', function ($id) {
    return "O filme pesquisado foi: " . $id;
});
```

Neste exemplo quando o usuário acessar o endereço http://localhost:8000/filme/10 automaticamente o Laravel vai entender que precisa executar o que está dentro da *function*.

Se o usuário acessar esse endereço vai ter como resposta um texto que diz:

```
O filme pesquisado foi: 10
```

Existem algumas formas de rotas que podem ser configuradas.
A mais comum é a rota do tipo GET.
Os demais tipos de rota (POST, PATCH, PUT, DELETE) a gente vai utilizar mais pra frente quando formos trabalhar com formulários e com APIs.