

# Aulas SENAC

Téc Desenvolvimento de  
Sistemas  
Pablo Garcia  
Laravel



# O que é Laravel

- ❑ Laravel é um framework PHP de código aberto para desenvolvimento web que fornece um ecossistema completo para a criação de aplicações web.
- ❑ Ele segue o padrão MVC (Model-View-Controller) e é conhecido por sua sintaxe expressiva e elegante.
- ❑ Laravel oferece uma variedade de recursos poderosos, como injeção de dependência, uma camada de abstração de banco de dados expressiva, filas e trabalhos agendados, testes unitários e de integração, entre outros.



# Quais os principais benefícios de usar Laravel

- 1. Sintaxe Elegante e Expressiva:** Laravel é conhecido por sua sintaxe limpa e legível, o que facilita a escrita e manutenção do código.
- 2. Arquitetura MVC:** A arquitetura Model-View-Controller (MVC) do Laravel ajuda a organizar o código de maneira eficiente, separando a lógica de negócios da apresentação.
- 3. Eloquent ORM:** O Eloquent ORM do Laravel facilita a interação com o banco de dados, permitindo que você trabalhe com modelos e relacionamentos de maneira intuitiva.



# Quais os principais benefícios de usar Laravel

4. **Segurança Integrada:** Laravel possui várias funcionalidades de segurança integradas, como proteção contra injeção de SQL e cross-site scripting (XSS).
5. **Artisan CLI:** A linha de comando Artisan do Laravel oferece várias ferramentas úteis para automatizar tarefas comuns de desenvolvimento.
6. **Migrações de Banco de Dados:** As migrações do Laravel fornecem um sistema de controle de versão para a estrutura do banco de dados, facilitando a definição e o compartilhamento de alterações.

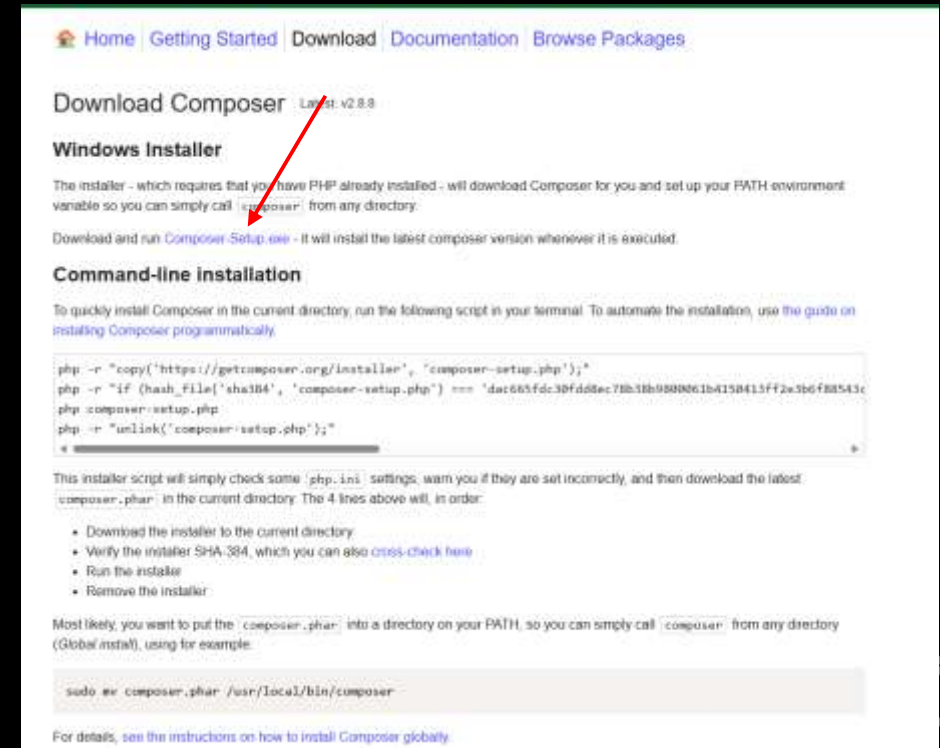
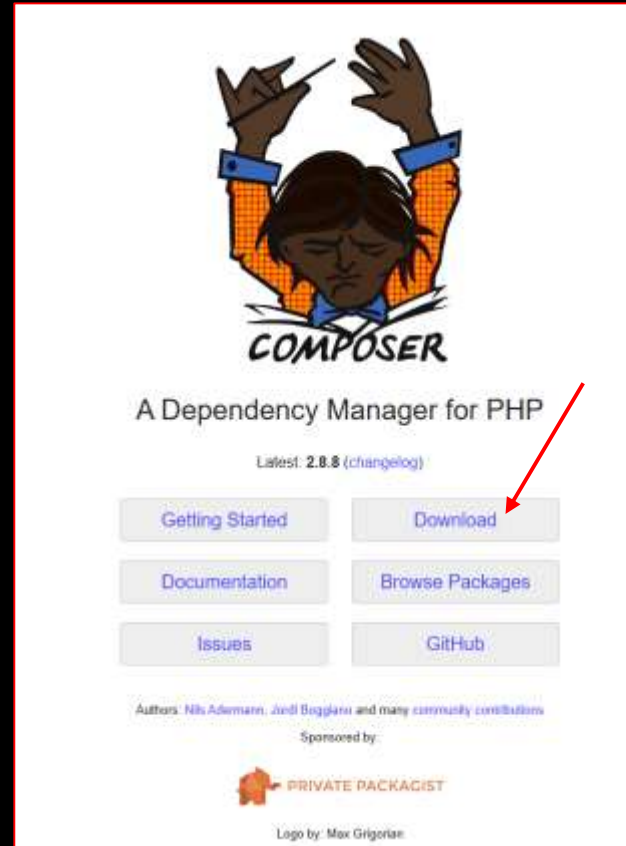


# Instalação de um projeto Laravel

## Composer -

<https://getcomposer.org>

O Composer é um gerenciador de dependências popular que facilita a instalação e atualização de bibliotecas PHP, como o Laravel.



# Instalação de um projeto Laravel

**Composer** - composer  
-V

```
C:\Users\pablo>composer -v
```

Após a instalação, precisamos verificar se o Composer foi instalado corretamente para avançar com o Laravel.

```
Selecionar C:\Windows\system32\cmd.exe
Microsoft Windows [versão 10.0.22000.2538]
(c) Microsoft Corporation. Todos os direitos reservados.

C:\Users\pablo>composer -v

Composer version 2.8.4 2024-12-11 11:57:47

Usage:
  command [options] [arguments]

Options:
  -h, --help                Display help for the given command. When no command is given display help for the list
  -q, --quiet                Do not output any message
  -V, --version              Display this application version
  --ansi|--no-ansi           Force (or disable) ANSI output
  -n, --no-interaction       Do not ask any interactive question
  --profile                  Display timing and memory usage information
  --no-plugins               Whether to disable plugins.
  --no-scripts               Skips the execution of all scripts defined in composer.json file.
  -d, --working-dir=WORKING-DIR If specified, use the given directory as working directory.
  --no-cache                 Prevent use of the cache
  -v|vv|vvv, --verbose       Increase the verbosity of messages: 1 for normal output, 2 for more verbose output and
                              3 for debug

Available commands:
```



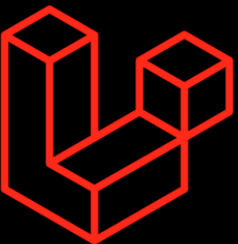
# Criando um projeto Laravel

```
1 laravel new example-app
```

↑ Documentação – Como Fazemos ↓

PROBLEMS OUTPUT TERMINAL PORTS DEBUG CONSOLE

```
PS C:\AulasSenacLaravel\14.PHP_Laravel_Pessoas_Back> composer create-project laravel/laravel:^10.0 ./
```



# Criando um projeto Laravel

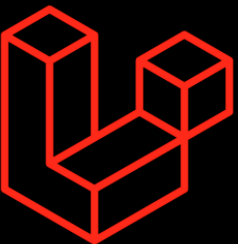
PROBLEMS OUTPUT TERMINAL PORTS DEBUG CONSOLE

PS C:\AulasSenacLaravel\14.PHP\_Laravel\_Pessoas\_Back> **composer** create-project laravel/laravel:^10.0 ./

Caminho

## Dica:

Pasta sempre deve  
estar vazia





# Criando um projeto Laravel

PROBLEMS OUTPUT TERMINAL PORTS DEBUG CONSOLE

PS C:\AulasSenacLaravel\14.PHP\_Laravel\_Pessoas\_Back> composer create-project laravel/laravel:^10.0 ./

Criando uma  
instalação local  
do Laravel

## Dica:

Se usarmos o  
comando: **laravel  
new** meu-app,  
geramos um  
instalação global  
do Laravel



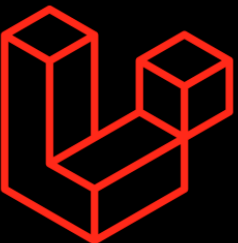
# Criando um projeto Laravel

```
PROBLEMS OUTPUT TERMINAL PORTS DEBUG CONSOLE  
PS C:\AulasSenacLaravel\14.PHP_Laravel_Pessoas_Back> composer create-project laravel/laravel:^10.0 ./
```

Versão do  
Laravel que  
quero instalar

## Dica:

Caso queira  
instalar a última  
versão não colocar  
nada aqui e ir  
direto para o nome  
do arquivo



# Criando um projeto Laravel

```
PROBLEMS OUTPUT TERMINAL PORTS DEBUG CONSOLE  
PS C:\AulasSenacLaravel\14.PHP_Laravel_Pessoas_Back> composer create-project laravel/laravel:^10.0 ./
```

## Dica:

Se colocarmos um nome ali, o Laravel criará mais um nível de pasta e será instalado lá dentro

./ indica que a instalação do Laravel deve ocorrer na pasta especificada

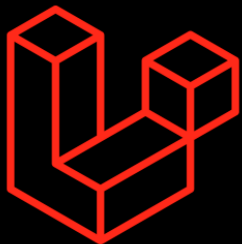


# Navegando nas pastas do

## Laravel

### App:

A pasta 'app' contém a maioria dos arquivos de código de sua aplicação, organizados em subpastas para facilitar o desenvolvimento.



EXPLORER

...

#### 14.PHP\_LARAVEL\_PESSOAS\_BACK

- > app
- > bootstrap
- > config
- > database
- > public
- > resources
- > routes
- > storage
- > tests
- > vendor
- .editorconfig
- .env
- .env.example
- .gitattributes
- .gitignore
- artisan
- composer.json
- composer.lock
- package.json
- phpunit.xml
- README.md
- vite.config.js

- > app
  - > Console
  - > Exceptions
  - > Http
    - > Controllers
      - Controller.php
      - UserController.php
    - > Middleware
      - Kernel.php
  - > Models
    - User.php
  - > Providers

# Navegando nas pastas do Laravel

## Http:

A subpasta 'Http' contém os controladores e middleware, responsáveis por gerenciar as requisições e respostas da aplicação.



### 14.PHP\_LARAVEL\_PESSOAS\_BACK

- > app
- > bootstrap
- > config
- > database
- > public
- > resources
- > routes
- > storage
- > tests
- > vendor
- .editorconfig
- .env
- .env.example
- .gitattributes
- .gitignore
- artisan
- composer.json
- composer.lock
- package.json
- phpunit.xml
- README.md
- vite.config.js

- > app
  - > Console
  - > Exceptions
  - > Http
    - > Controllers
      - Controller.php
      - UserController.php
    - > Middleware
      - Kernel.php
  - > Models
    - User.php
  - > Providers

# Navegando nas pastas do

## Laravel

### Models:

A subpasta 'Models' é onde você define as classes do modelo, que representam os dados da sua aplicação e sua lógica de negócios.



#### 14.PHP\_LARAVEL\_PESSOAS\_BACK

- > app
- > bootstrap
- > config
- > database
- > public
- > resources
- > routes
- > storage
- > tests
- > vendor
- .editorconfig
- .env
- .env.example
- .gitattributes
- .gitignore
- artisan
- composer.json
- composer.lock
- package.json
- phpunit.xml
- README.md
- vite.config.js

- > app
  - > Console
  - > Exceptions
  - > Http
    - > Controllers
      - Controller.php
      - UserController.php
    - > Middleware
      - Kernel.php
  - > Models
    - User.php
  - > Providers

# Navegando nas pastas do Laravel

## 14.PHP\_LARAVEL\_PESSOAS\_BACK

- > app
- > bootstrap
- > config
- > database
- > public
- > resources
- > routes
- > storage
- > tests
- > vendor
- .editorconfig
- .env
- .env.example
- .gitattributes
- .gitignore
- artisan
- composer.json
- composer.lock
- package.json
- phpunit.xml
- README.md
- vite.config.js

### database

#### factories

#### migrations

- 2014\_10\_12\_000000\_create\_users\_table.php
- 2014\_10\_12\_100000\_create\_password\_reset\_tokens\_table.php
- 2019\_08\_19\_000000\_create\_failed\_jobs\_table.php
- 2019\_12\_14\_000001\_create\_personal\_access\_tokens\_table.php

#### seeders

.gitignore

## Migrations:

Com as migrations, você pode criar novas tabelas, modificar ou alterar tabelas existentes, adicionar colunas e até mesmo preencher o banco de dados com dados iniciais.

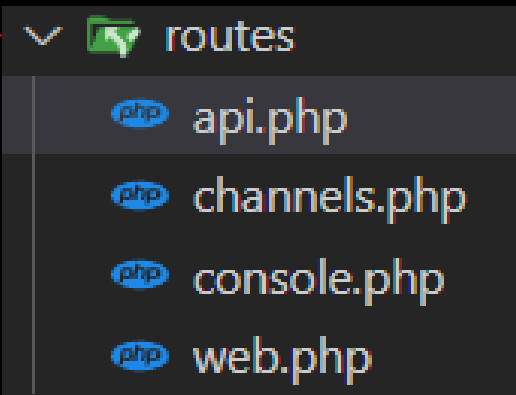
Isso elimina a necessidade de configurar manualmente o banco de dados e reduz significativamente as chances de erros.



# Navegando nas pastas do Laravel

## routes:

No Laravel, a pasta routes contém todos os arquivos de definição de rotas da aplicação. Esses arquivos são responsáveis por mapear URLs para funções específicas ou controladores que lidam com as requisições HTTP. A pasta routes geralmente inclui os seguintes arquivos principais:



14.PHP\_LARAVEL\_PESSOAS\_BACK

- > app
- > bootstrap
- > config
- > database
- > public
- > resources
- > routes
- > storage
- > tests
- > vendor
- .editorconfig
- .env
- .env.example
- .gitattributes
- .gitignore
- artisan
- composer.json
- composer.lock
- package.json
- phpunit.xml
- README.md
- vite.config.js



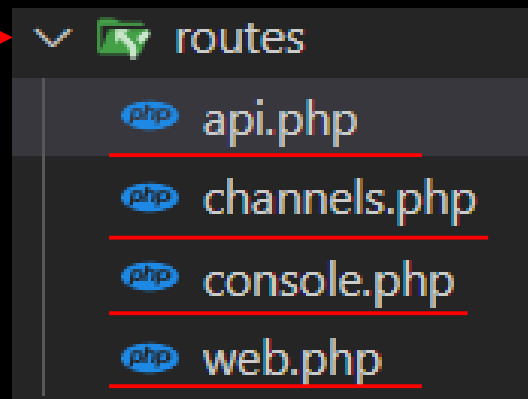
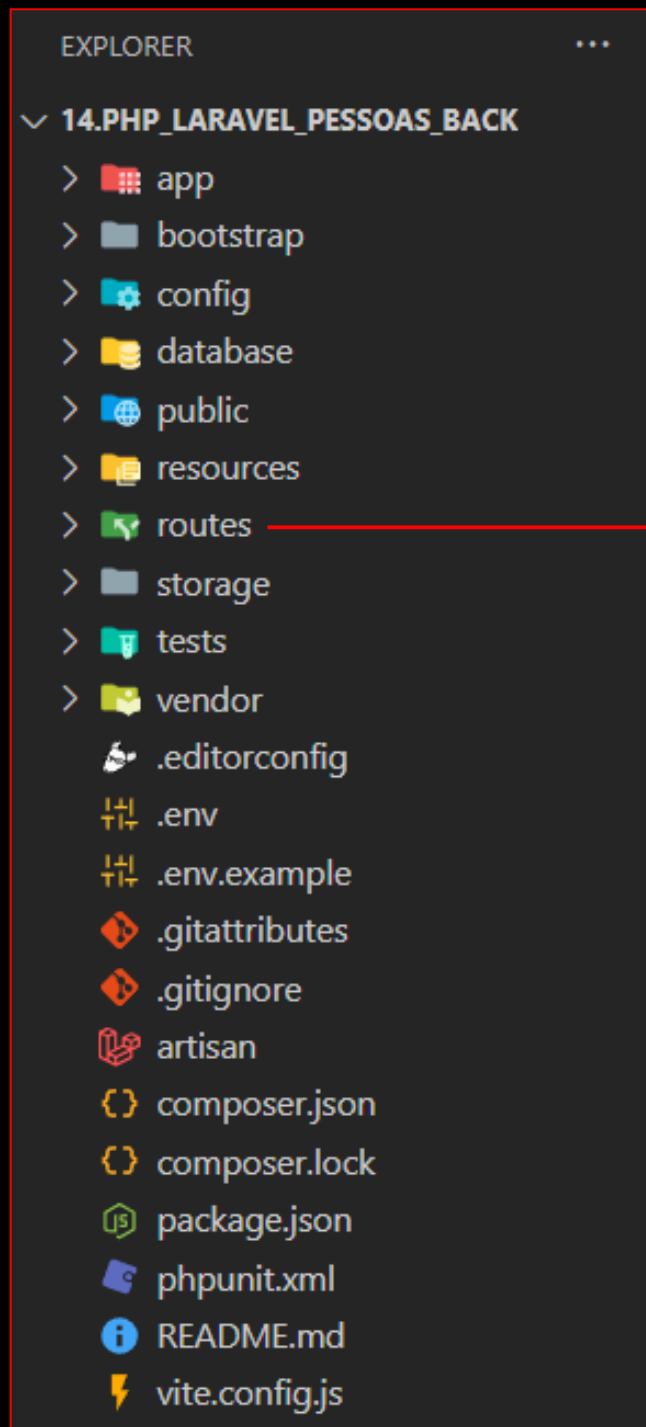
# Navegando nas pastas do Laravel

**api.php:** Define as rotas para a API da aplicação. As rotas definidas em api.php são sem estado (não mantêm informações de sessão entre as requisições).

**channels.php:** Define todos os canais de transmissão de eventos que a aplicação pode usar.

**console.php:** Define todas as rotas baseadas em console (comandos Artisan).

**web.php:** Define as rotas para a interface web da aplicação



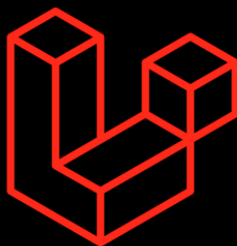
# Navegando nas pastas do Laravel

## `.env`:

O arquivo `.env` é essencial para definir variáveis de ambiente em aplicações, ajudando na configuração adequada do ambiente de desenvolvimento.

Gerenciar dados sensíveis como credenciais de banco de dados no arquivo `.env` permite maior segurança e proteção contra vazamentos de informação.























Uma configuração eficiente do arquivo `.env` pode simplificar o desenvolvimento e a implementação de aplicações, facilitando a colaboração em equipe.



EXPLORER

...

### 14.PHP\_LARAVEL\_PESSOAS\_BACK

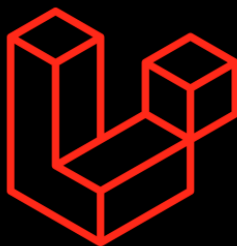
- >  app
- >  bootstrap
- >  config
- >  database
- >  public
- >  resources
- >  routes
- >  storage
- >  tests
- >  vendor
-  .editorconfig
-  `.env`
-  `.env.example`
-  .gitattributes
-  .gitignore
-  artisan
-  composer.json
-  composer.lock
-  package.json
-  phpunit.xml
-  README.md
-  vite.config.js

# Navegando nas pastas do Laravel

## Dica:

Lembrar de apagar estas pastas  
para compartilhar projetos  
PHP/Laravel.

A pasta **vendor** e o **.env** já estão  
listados no **.gitignore**.



EXPLORER

...

### 14.PHP\_LARAVEL\_PESSOAS\_BACK

- > app
- > bootstrap
- > config
- > database
- > public
- > resources
- > routes
- > storage
- > tests
- > vendor
- .editorconfig
- .env
- .env.example
- .gitattributes
- .gitignore
- artisan
- composer.json
- composer.lock
- package.json
- phpunit.xml
- README.md
- vite.config.js

# Comandos Laravel

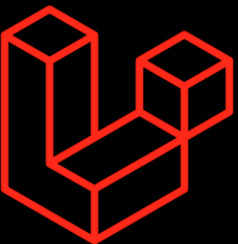
PROBLEMS OUTPUT TERMINAL PORTS DEBUG CONSOLE

PS C:\AulasSenacLaravel\14.PHP\_Laravel\_Pessoas\_Back> composer create-project laravel/laravel:^10.0 ./

## Dica:

Se colocarmos um nome ali, o Laravel criará mais um nível de pasta e será instalado lá dentro

Criar um projeto  
Laravel na pasta  
indicada



# Comandos Laravel

PROBLEMS

OUTPUT

TERMINAL

PORTS

DEBUG CONSOLE

```
PS C:\AulasSenacLaravel\14.PHP_Laravel_Pessoas_Back> composer install
```

## Dica:

Cada vez que baixamos um projeto Laravel, de algum lugar ou repositório, devemos dar um **composer install**, para que a pasta **vendor** e outras dependências sejam instaladas.

Atualiza/instala  
dependências  
necessárias em um  
projeto Laravel



# Comandos Laravel

PROBLEMS OUTPUT TERMINAL PORTS DEBUG CONSOLE

PS C:\AulasSenacLaravel\14.PHP\_Laravel\_Pessoas\_Back> php artisan serve

INFO Server running on [http://127.0.0.1:8000].

Press Ctrl+C to stop the server

Inicia um servidor embutido no php para teste da aplicação

## Dica:

Para parar o servidor tem que apertar Ctrl+c no terminal.



# Comandos Laravel

PROBLEMS OUTPUT TERMINAL PORTS DEBUG CONSOLE

PS C:\AulasSenacLaravel\14.PHP\_Laravel\_Pessoas\_Back> php artisan migrate

Faz a migração das  
tabelas para o banco  
de dados

## Dica:

Caso o banco de dados configurado na  
**.env** não exista, o banco de dados  
também é criado.



# Comandos Laravel

PROBLEMS

OUTPUT

TERMINAL

PORTS

DEBUG CONSOLE

PS C:\AulasSenacLaravel\14.PHP\_Laravel\_Pessoas\_Back> php artisan migrate:fresh

Atualiza as tabelas  
do banco de dados

## Dica:

Caso já haja um banco de dados, com dados e que não queiramos perder, somente alterar ou acrescentar uma tabela, por exemplo.





# Comandos Laravel

```
PROBLEMS OUTPUT TERMINAL PORTS DEBUG CONSOLE  
PS C:\AulasSenacLaravel\14.PHP_Laravel_Pessoas_Back> php artisan make:Controller CarController --resource
```

## Dica:

Caso não queira as funções padrões, basta não colocar o `--resource` ao final.

Cria uma controller já com o nome especificado, sua classe e funções básicas, ligadas a rotas `api.php`



# Comandos Laravel

PROBLEMS OUTPUT TERMINAL PORTS DEBUG CONSOLE

```
PS C:\AulasSenacLaravel\14.PHP_Laravel_Pessoas_Back> php artisan make:model CarModel -mcr
```

## Dica:

- m para criar uma migration.
- c para criar um controller.
- r para indicar que o controller gerado deve ser um controller de recurso.

Cria model, migration  
e controller juntas e  
com os nomes  
coerentes



# Obrigado

Téc Desenvolvimento de  
Sistemas  
Pablo Garcia  
Laravel

