CẤU TRÚC THƯ MỤC - VÒNG ĐỜI REQUEST LARAVEL

Nguyên tắc chung

Với Laravel Framework, bạn không nhất thiết phải tuân thủ cấu trúc thư mục mặc định của Laravel. Bạn hoàn toàn có thể tái cấu trúc lại thư mục, miễn sao các Class tuân thủ nguyên tắc autoload của Composer.

Cấu trúc thư mục Laravel 8.x

Thư mục app

- Thư mục app chứa tất cả các Class của project

Thư mục app/Console

- Thư mục chứa các tập tin định nghĩa các câu lệnh trên artisan

Thư mục app/Exceptions

- Thư mục chứa các tập tin quản lý, điều hướng lỗi

Thư mục app/Http/Controllers

- Thư mục chứa các controller của project

Thư mục app/Http/Middleware

- Thư mục chứa các tập tin lọc và ngăn chặn các requests

Thư mục app/Providers

- Thư mục chứa các file thực hiện việc khai báo service và bind vào trong Service Container

Thư mục app/Models

- Thư mục chứa các model của project (Với Laravel 8 sẽ có sẵn thư mục Models)

Thư mục bootstrap

- Thư mục chứa những file khởi động của framework và những file cấu hình auto loading, route, và file cache

Thư mục config

- Thư mục chứa tất cả những file cấu hình

Thư mục database

- Thư mục chứa 2 thư mục migration (tạo và thao tác database) và seeds (tạo dữ liệu mẫu)

Thư mục database/factories

- Thư mục chứa các file định nghĩa các cột bảng dữ liệu để tạo ra các dữ liệu mẫu

Thư mục database/migrations

- Thư mục chứa các file tạo và chỉnh sửa dữ liệu

Thư mục database/seeds

- Thư mục chứa các file tạo dữ liệu thêm vào CSDL

Thư mục public

- Thư mục chứa file index.php giống như cổng cho tất cả các request vào project, bên trong thư mục còn chứa file JavaScript, và CSS

Thư mục resources

- Thư mục chứa những file view và raw, các file biên soạn như LESS, SASS, hoặc JavaScript. Ngoài ra còn chứa tất cả các file language trong project.

Thư mục resources/views

- Thư mục chứa các file view xuất giao diện người dùng

Thư mục routes

- Thư mục chứa tất cả các điều khiển route (đường dẫn) trong project.

Chứa các file route sẵn có: web.php, channels.php, api.php, và console.php

Thư mục routes/api.php

- Cấu hình các route liên quan đến API

Thư mục routes/web.php

- Cấu hình các route liên quan đến web (Có giao diện người dùng)

Thư mục storage

- Thư mục chứa các file biên soạn blade templates của bạn, file based sessions, file caches, và những file sinh ra từ project.
- Thư mục app, dùng để chứa những file sinh ra từ project.
- Thư mục framework, chứa những file sinh ra từ framework và caches.
- Thư mục logs, chứa những file logs.
- Thư mục /storage/app/public, lưu những file người dùng tạo ra như hình ảnh.

Thư mục tests

- Thư mục chứa những file tests

Thư mục vendor

- Thư mục chứa các thư mục, file thư viện của Composer

File .env

- File chứa các config chính của Laravel

File artisan

- File thực hiện lệnh của Laravel

File .gitattributes, .gitignore

- File dùng để xử lý git

File composer.json, composer.lock, composer-setup.php

- File sinh ra của composer

File package.json

- File chứa các package cần dùng cho projects

File phpunit.xml

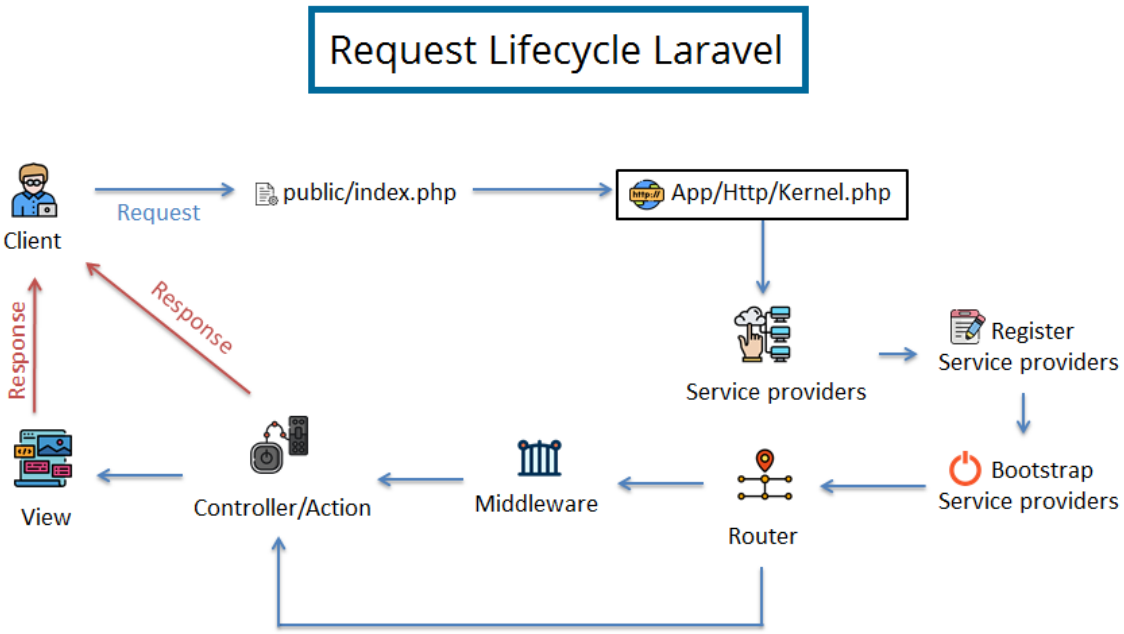
- File phpunit.xml, xml của phpunit dùng để testing project

File webpack.mix.js

- File dùng để build các webpack

Vòng đời Request Laravel

Vòng đời Request là quá trình khi có một yêu cầu từ phía người dùng, Laravel sẽ tiếp nhận và xử lý nó như thế nào và cuối cùng Request đó sẽ trả về kết quả như thế nào.



Bước 01: Tiếp nhận Request và khởi động (Bootstrap)

Người dùng (Client) sẽ gửi Request tới file public/index.php, đây sẽ là file chạy đầu tiên khi có Request từ phía người dùng.

Đăng ký cơ chế autoload

```
require __DIR__.'/../vendor/autoload.php';
```

Hiểu một cách đơn giản, Autoload sẽ thay thế các lệnh require, include. Chỉ cần trong các file có tên Class trùng với tên file và có namespace theo quy tắc, hệ thống sẽ tự động load các file đó mà không cần phải sử dụng các lệnh require hoặc include

Import ứng dụng

```
$app = require_once __DIR__.'/../bootstrap/app.php';
```

Trong file `bootstrap/app.php` sẽ thực hiện các nhiệm vụ sau:

- Tạo ứng dụng
- Đăng ký các interface cần thiết
- Trả về đối tượng ứng dụng ($app)

Bước 02: Chạy ứng dụng

```
$kernel =
$app->make(Illuminate\Contracts\Http\Kernel::class);

$response = $kernel->handle(
$request = Illuminate\Http\Request::capture()
);

$response->send();

$kernel->terminate($request, $response);
```

Bước này vẫn ở trong file `public/index.php`

Hệ thống sẽ nhận đối tượng $app trả về từ bước 1, sau đó thực hiện 2 công việc:

- Xử lý Request
- Trả về Response

Bước 03: HTTP Kernel

HTTP Kernel sẽ thực hiện các công việc trước khi Request được thực thi (Tiền xử lý):

- Xử lý lỗi
- Cấu hình log
- Xác định môi trường
- Xác thực bảo mật
- Bộ lọc trung gian mặc định (Middleware default)
- HTTP Session

Và rất nhiều công việc khác

Bước 04: Service providers

Hiểu một cách cơ bản Service Providers chính là trung tâm khởi tạo các ứng dụng của Laravel. Service Providers khởi động rất nhiều thành phần khác nhau trong core cũng như các package được cài thêm vào

Bước 05: Router

Sau khi hoàn tất công việc ở bước 4, request sẽ gửi đến Router để định tuyến đến các Controller tương ứng. Ngay lúc này sẽ xảy ra 2 trường hợp:

- Trường hợp 01: Router => Controller => Action
- Trường hợp 02: Router => Middleware => Controller => Action

Với trường hợp 2, Request sẽ phải chạy qua bộ lọc trung gian (Middleware) trước khi vào Controller tương ứng

Bước 06: Middleware

Middleware chính là bộ lọc trung gian giúp lọc Request từ phía Router, trong Middleware lập trình viên sẽ thiết lập các điều kiện để cho Request đi tiếp hoặc dừng lại.

Middleware được chia thành 3 loại:

- Global Middleware
- Route Middleware
- Middleware Groups

Bước 07: Controller/Action

Thành phần này sẽ xử lý các Request, sau đó trả về Response.

Trong quá trình xử lý bước này còn phải gọi Models để thao tác với cơ sở dữ liệu.

Bước 08: Phương thức trả về

Sau khi Request được xử lý xong, Response sẽ được trả về cho người dùng (Clients) thông qua View (Giao diện), sẽ có những trường hợp không thông qua Views (JSON, XML, Download,…)