Course - Laravel Framework

Cấu hình

Tất cả các tập tin cấu hình cho Laravel framework đều được lưu trữ trong thư mục config, các tập tin cấu hình cho phép cấu hình những thứ như thông tin kết nối CSDL, thông tin server mail, và các cấu hình cốt lõi khác như timezone và encryption key của ứng dụng.

Tags: cau hinh, laravel

Các cấu hình môi trường

Trong cài đặt mới của Laravel, thư mục gốc của ứng dụng có chứa tập tin `.env.example`, trong đó nó có chứa nhiều biến môi trường thông thường. Trong suốt tiến trình cài đặt, tập tin này được sao chép thành tập tin `.env`.

Các giá trị cấu hình thông thường trong .env khi trong server web cục bộ hay trên production server đều sẽ được truy cập bởi các tập tin trong thư mục `config` bằng hàm `env` của Laravel.

Các kiểu biến

Thông thường các biến trong tập tin `.env` được lưu dưới dạng chuỗi, nhưng khi nhận giá trị từ hàm `env()` thì có một số kiểu giá trị khác nhau.

`.env` Value	`env()` Value
`true`	`(bool) true`
`(true)`	`(bool) true`
`false`	`(bool) false`
`(false)`	`(bool) false`
`empty`	`(string) ''`
`(empty)`	`(string) ''`
`null`	`(null) null`
`(null)`	`(null) null`

Nếu bạn cần khai báo một biến môi trường có giá trị chứa khoảng trắng, bạn có thể bọc giá trị trong cặp dấu ngoặc kép.

```
APP_NAME="My Application"
```

Nhận cấu hình môi trường

Tất cả các biến được liệt kê trong tập tin này sẽ được tải vào trong biến tổng bộ `$_ENV` của PHP, khi ứng dụng của bạn nhận một request. Bạn có thể nhận lại các giá trị này bằng helper `env`.

```
'debug' => env('APP_DEBUG', false),
```

Giá trị thứ hai trong helper `env` ở trên là giá trị mặc định sẽ được nhận nếu như không tồn tại biến này trong tập tin `.env`.

Xác nhận môi trường hiện tại

Môi trường hiện tại của ứng dụng của bạn có thể nhận được qua biến `APP_ENV` trong tập tin `.env`, bạn có thể nhận được giá trị này bằng phương thức `environment` trong facade `App`.

```
use Illuminate\Support\Facades\App;

$environment = App::environment();
```

Bạn cũng có thể truyền tham số vào trong phương thức `environment` để xác định môi trường có giống với môi trường đã đưa vào trong tham số hay không. Phương thức này sẽ trả lại là `true` nếu môi trường giống tham số được truyền vào.

```
if (App::environment('local'))
{
  // The environment is local
}

if (App::environment(['local', 'staging']))
{
  // The environment is either local OR staging...
}
```

Truy cập giá trị cấu hình

Bạn có thể dễ dàng truy cập vào các giá trị cấu hình bằng cách sử dụng helper `config` ở bất kỳ nơi nào trong ứng dụng của bạn. Các giá trị có thể được truy cập bằng cú pháp dot. Nó gồm tên của tập tin và tên option mà bạn muốn truy cập. Có thể cho thêm giá trị mặc định vào tham số, để sử dụng khi không tồn tại option cấu hình.

```
$value = config('app.timezone');
```

```php
// Retrieve a default value if the configuration value does not exist...
$value = config('app.timezone', 'Asia/Seoul');
```

Để cấu hình lúc chạy ứng dụng, truyền vào tham số một mảng giá trị như sau,

```php
config(['app.timezone' => 'America/Chicago']);
```

Caching cấu hình

Để cho ứng dụng chạy nhanh hơn trong quá trình deploy, thì bạn sẽ thường cache các tập tin cấu hình của mình vào một tập tin duy nhất và nhờ như vậy, ứng dụng sẽ tải nhanh hơn, dùng tốt nhất lúc deploy ứng dụng của bạn. Hãy chạy lệnh sau.

```
php artisan config:cache
```

Chú ý: một khi bạn dùng lệnh `config:cache` thì bạn phải sử dụng hàm `env` cho các tập tin cấu hình của bạn. Một khi cache cấu hình, thì tập tin `.env` sẽ không được tải lại nữa.

Chế độ Debug

Option `debug` trong tập tin `config/app.php` tương ứng với biến `APP_DEBUG` trong tập tin `.env` sẽ cho biết có bao nhiêu thông tin về lỗi mà sẽ được hiển thị cho người dùng. Đối với môi trường phát triển cục bộ, bạn nên đặt biến môi trường `APP_DEBUG` là `true`. Trong môi trường production, giá trị này nên là `false`, nhằm tránh để lộ các thông tin cấu hình nhạy cảm cho những người dùng cuối.

Chế độ Bảo trì

Khi ứng dụng đang được bảo trì, một màn hình tùy biến sẽ được hiện lên cho mọi yêu cầu được gửi về ứng dụng của bạn. Nó giúp vô hiệu hóa ứng dụng khi đang nâng cấp, bảo trì. Nếu đang trong trạng thái bảo trì `Symfony\Component\HttpKernel\Exception\HttpException` sẽ quăng ra code trạng thái `503`.

Để vào chế độ bảo trì, hãy dùng lệnh Artisan `down` như sau,

```
php artisan down
```

Bạn có thể sử dụng option refresh để tự động refresh page theo thời gian từ giá trị được gán kèm theo, HTTP header refresh sẽ được gửi đến tất cả các response.

```
php artisan down --refresh=15
```

Bạn còn có thể dùng option retry để cài đặt giá trị của HTTP header Retry-After. Tuy nhiên các trình duyệt web đa số đều phớt lờ header này.

```
php artisan down --retry=60
```

Vượt cấp chế độ bảo trì

Trong chế độ bảo trì bạn vẫn có thể vượt cấp để duyệt ứng dụng giống như khi nó chưa trong chế độ bảo trì. Khi đang trong chế độ bảo trì bạn có thể sử dụng option secret để vượt cấp bằng token.

```
php artisan down --secret="1630542a-246b-4b66-afa1-dd72a4c43515"
```

Sau đó, bạn có thể điều hướng đến ứng dụng với URL khớp với token này, và nó sẽ trình bày token cookie bảo trì lên trên trình duyệt.

```
https://example.com/1630542a-246b-4b66-afa1-dd72a4c43515
```

Khi truy cập route ẩn này, bạn sẽ được điều hướng tới route `/` của ứng dụng. Một khi cookie được phát tán trên trình duyệt, thì bạn có thể truy cập ứng dụng của bạn một cách bình thường giống như cả khi nó chưa trong chế độ bảo trì.

Pre-render với chế độ bảo trì

Bạn có thể áp dụng trang thông báo pre-render để bỏ qua việc tải các dependency của ứng dụng bằng option --render.

```
php artisan down --render="errors::503"
```

Redirect với chế độ bảo trì

Trong khi đang trong chế độ bảo trì, Laravel sẽ hiển thị một trang thông báo chung cho tất cả các URL yêu cầu về hệ thống. Tuy nhiên, nếu bạn muốn chuyển hướng người dùng về

một trang cụ thể nào đó thì có thể sử dụng option `redirect`. Ví dụ, bạn muốn chuyển tất cả các yêu cầu về URL duy nhất là `/`, thì dùng lệnh như sau,

```
php artisan down --redirect=/
```

Dừng chế độ bảo trì

Để dừng chế độ bảo trì, dùng lệnh `up` như sau,

```
php artisan up
```

Chú ý: Bạn có thể tùy biến trang thông báo bảo trì bằng cách sửa lại template sau. *resources/views/errors/503.blade.php*