Course - Laravel Framework

Phát hành

Khi phát hành ứng dụng Laravel, có vài điều cần phải thực hiện để giúp cho dự án của bạn chạy hiệu quả nhất có thể.

Tags: Phát hành

Cấu hình yêu cầu

- Php7.3
- BCMath PHP Extension
- Ctype PHP Extension
- Fileinfo PHP Extension
- JSON PHP Extension
- Mbstring PHP Extension
- OpenSSL PHP Extension
- PDO PHP Extension
- Tokenizer PHP Extension
- XML PHP Extension

NginX

Bạn có thể sử dụng server nginx để deploy ứng dụng của bạn. Bạn không nên ánh xạ đến thư mục root của dự án mà thay vào đó bạn nên ánh xạ vào thư mục public hoặt thư mục có cấp độ tương ứng. Bạn có thể download, khởi động nginx.

Khi đang trong thư mục nginx, hãy chạy lệnh như sau,

```
nginx
```

Trong lúc đang chạy nginx, có thể áp dụng option `-s` với các signal như

1. `quit` - để thoát nginx
2. `reload` - để tải lại cấu hình nginx
3. `reopen` - để mở lại nginx

Ví dụ, để thoát nginx, hãy gõ lệnh như sau,

```
nginx -s quit
```

Hãy sử dụng cấu hình sau cho ứng dụng Laravel của bạn. Mở tập tin *conf/nginx.conf* trong thư mục của nginx lên và dán

```
server {
    listen 80;
    server_name example.com;
    root /srv/example.com/public;
```

```
    add_header X-Frame-Options "SAMEORIGIN";
    add_header X-Content-Type-Options "nosniff";

    index index.php;

    charset utf-8;

    location / {
        try_files $uri $uri/ /index.php?$query_string;
    }

    location = /favicon.ico { access_log off; log_not_found off; }
    location = /robots.txt  { access_log off; log_not_found off; }

    error_page 404 /index.php;

    location ~ \.php$ {
        fastcgi_pass unix:/var/run/php/php7.4-fpm.sock;
        fastcgi_param SCRIPT_FILENAME $realpath_root$fastcgi_script_name;
        include fastcgi_params;
    }

    location ~ /\.(?!well-known).* {
        deny all;
    }
}
```

Nếu đang trên windowsOS thì hãy chạy thêm PHP Fast CGI trong thư mục của php,

```
 set PHP_FCGI_MAX_REQUESTS=0
```

```
 php-cgi.exe -b 127.0.0.1:9000
```

Và sửa lại tập tin cấu hình như sau,

```
server {
     listen 80;
```

```
    server_name localhost;
    root E:\src\example-app\public;

    add_header X-Frame-Options "SAMEORIGIN";
    add_header X-Content-Type-Options "nosniff";

    index index.php;

    charset utf-8;

    location / {
        try_files $uri $uri/ /index.php?$query_string;
    }

    location = /favicon.ico { access_log off; log_not_found off; }
    location = /robots.txt  { access_log off; log_not_found off; }

    error_page 404 /index.php;

    location ~ \.php$ {
        root           E:\src\example-app\public;
        fastcgi_pass   127.0.0.1:9000;
        fastcgi_index  index.php;
        fastcgi_param  SCRIPT_FILENAME  $document_root$fastcgi_script_name;
        include        fastcgi_params;
    }

    location ~ /\.(?!well-known).* {
        deny all;
    }
}
```

Tối ưu hóa

Tối ưu hóa autoloader

Khi deploy ứng dụng thành thành phẩm, thì hãy dùng lệnh sau để tối ưu hóa autoloader của composer.

```
composer install --optimize-autoloader --no-dev
```

> Khi ứng dụng được lưu thành trên repository, thì hãy kèm theo tập tin *composer.lock*, nếu hiện diện trong thư mục dự án thì các package kiểu dependency sẽ được cài đặt nhanh hơn cho ứng dụng.

Tăng tốc tải cấu hình

Khi phát hành ứng dụng của bạn trên môi trường production, hãy chạy lệnh Artisan `config:cache`.

```
php artisan config:cache
```

Lệnh này sẽ giúp compile toàn bộ các tập tin cấu hình của dự án thành một tập tin lưu trữ duy nhất, nhờ đó có thể bỏ qua nhiều lượt tải tập tin bởi framework khi tải các giá trị cấu hình của ứng dụng.

> **Chú ý:** Khi bạn gọi lệnh `config:cache` thì hãy đảm bảo là chỉ sử dụng helper `env` cho các tập tin cấu hình của bạn. Vì nó sẽ không tải lại tập tin *.env*, nên khi dùng helper `env` cho tập tin này sẽ trả lại giá trị là `null`.

Tăng tốc tải route

Khi ứng dụng của bạn có nhiều route, thì bạn nên sử dụng lệnh Artisan route:cache,

```
php artisan route:cache
```

Lệnh này sẽ gom hàng trăm route trong ứng dụng của bạn vào trong một tập tin duy nhất nhằm tối ưu hóa việc đăng ký route cho hàng trăm route của ứng dụng.

Tăng tốc tải view

Khi triển khai ứng dụng của bạn trên môi trường production, hãy chạy lệnh `view:cache`.

```
php artisan view:cache
```

Lệnh này sẽ compile sẵn một lần tất cả các template Blade để chúng không phải compile lại.