Laravel Modules

Mô hình MVC của Laravel hỗ trợ rất tốt cho các dự án trung bình và nhỏ, tuy nhiên, với các dự án lớn hoặc cần mở rộng với nhiều người phát triển thì mô hình MVC lại bộc lộ một vài khuyết điểm, đáng kể nhất chính là khó quản lý số số lượng code khi chúng được tăng lên không ngừng.

Để giải quyết vấn đề này, Laravel đã cung cấp cho chúng ta khả năng Module hóa sử dụng mô hình HMVC (Hierarchy - Model - View - Controller). HMVC giúp chúng ta chia từng tính tăng thành các nhóm với cấu trúc MVC và Routing riêng biệt từ đó giúp chúng ta dễ dang quản lý code cũng như khả năng mở rộng trong tương lai.

Khai báo ban đầu và khởi tạo ServiceProvider

Khai báo Folder chứa Modules

Chúng ta sẽ đặt Folder chứa Modules tại thư mục root của ứng dụng và đặt tên là "modules", lưu ý là Folder này không nhất thiết phải đặt tại thư mục root và cũng không nhất thiết có tên là "modules".

Khi tạo xong chúng ta sẽ khai báo trong mục "autoload/psr-4" ở file "composer.json"

```
"autoload": {
    "psr-4": {
        "App\\": "app/",
        "modules\\": "modules/"
    },
    "classmap": [
        "database/seeds",
        "database/factories"
    ]
},
```

Cuối cùng chạy "composer dump-autoload" để hoàn tất khai báo.

Khởi tạo và khai báo ServiceProvider

Chúng ta sẽ tạo file "ModuleServiceProvider.php" đặt tại thư mục "modules" vừa tạo và nó có nội dung như sau:

```
<?php

namespace modules;
```

```
use Illuminate\Support\ServiceProvider;
use File;

class ModuleServiceProvider extends ServiceProvider
{

    public function register() {}

    public function boot(){}

}
```

Để hoàn tất quá trình khai báo thì chúng ta cần một bước cuối cùng nữa đó là khai báo trong "config/app.php" tại mục "providers":

```
'providers' => [
    ...
    /*
    * Custom Service Providers...
    */
    'App\Modules\ServiceProvider',
]
```

Tạo Module cùng với khai báo ServiceProvider tương ứng

Cấu trúc Module cơ bản

```
modules
├── Demo
|   ├── configs
|   |   └── demo.php
|   ├── migrations
|   ├── resources
|   |   └── lang
|   |   └── views
|   ├── routes
|   |   └── routes.php
|   ├── src
|   |   └── Commands
|   |   └── Http
|   |   |   └── Controllers
|   |   |   └── Middlewares
|   |   └── Models
├── Demo2
|   ├── ...
└── ModuleServiceProvider.php
```

Khai báo ServiceProvider để có thể load được toàn bộ modules

```
?php

namespace modules;
use Illuminate\Support\ServiceProvider;
use File;

class ModuleServiceProvider extends ServiceProvider
{

    public function register() {}

    public function boot(){
        // Đăng ký modules theo cấu trúc thư mục
        $directories = array_map('basename', File::directories(__DIR__));
        foreach ($directories as $moduleName) {
            $this->registerModule($moduleName);
        }
    }

    // Khai báo đăng ký cho từng modules
    private function registerModule($moduleName) {
        $modulePath = __DIR__ . "/$moduleName/";
        // Khai báo thành phần ở đây
    }

}
```

Khai báo các thành phần có tại cấu trúc thư mục

Khai báo routes, migrations, langs, views, helpers

Khai báo các thành phần của modules tại "modules/ModuleServiceProvider.php" và nó được đặt tại hàm "registerModule".

```
private function registerModule($moduleName) {
    $modulePath = __DIR__ . "/$moduleName/";

    // Khai báo route
    if (File::exists($modulePath . "routes/routes.php")) {
        $this->loadRoutesFrom($modulePath . "routes/routes.php");
    }

    // Khai báo migration
    // Toàn bộ file migration của modules sẽ tự động được load
    if (File::exists($modulePath . "migrations")) {
        $this->loadMigrationsFrom($modulePath . "migrations");
    }

    // Khai báo languages
    if (File::exists($modulePath . "resources/lang")) {
        // Đa ngôn ngữ theo file php
        // Dùng đa ngôn ngữ tại file php resources/lang/en/general.php : @lang('Demo::general.hello')
```

```
        $this->loadTranslationsFrom($modulePath . "resources/lang", $moduleName);
        // Đa ngôn ngữ theo file json
        $this->loadJSONTranslationsFrom($modulePath . 'resources/lang');
    }

    // Khai báo views
    // Gọi view thì ta sử dụng: view('Demo::index'), @extends('Demo::index'), @include('Demo::index')
    if (File::exists($modulePath . "resources/views")) {
        $this->loadViewsFrom($modulePath . "resources/views", $moduleName);
    }

    // Khai báo helpers
    if (File::exists($modulePath . "helpers")) {
        // Tất cả files có tại thư mục helpers
        $helper_dir = File::allFiles($modulePath . "helpers");
        // khai báo helpers
        foreach ($helper_dir as $key => $value) {
            $file = $value->getPathName();
            require $file;
        }
    }
}
```

configs

Để có thể sử dụng configs thì chúng ta cần khai báo tại function "register”

```
public function register() {
    ...
    // Khai báo configs
    $configFile = [
        'demo' => __DIR__.'/Demo/configs/demo.php',
    ];
    foreach ($configFile as $alias => $path) {
        $this->mergeConfigFrom($path, $alias);
    }
    ...
}
```

Khi khai báo như trên thì chúng ta có thể gọi config tại module như các config bình thường khác

Middlewares

Để có thể sử dụng Middlewares thì chúng ta cần khai báo tại function "register”

```
public function register() {
    ...
    // Khai báo middleare
    $middleare = [
        // 'key' => 'namespace của middleare'
        'demo' => '\modules\Demo\src\Http\Controllers\Middlewares\DemoMiddleware',
    ];
```

```
    foreach ($middleare as $key => $value) {
        $this->app['router']->pushMiddlewareToGroup($key, $value);
    }
    ...
}
```

Commands

Để có thể sử dụng commands thì chúng ta cần khai báo tại function "register”

```
public function register() {
    ...
    // Khai báo commands
    $this->commands([
        // namespace của commands đặt tại đây
        '\modules\Demo\src\Http\Commands\DemoCommand'
    ]);
    ...
}
```