**Bài 21: Validation trong Laravel (Phần 2)**

Phần trước mình đã giới thiệu với mọi người 2 cách validation form trực tiếp trên Controller rồi, nhưng các cách đó đều được cho là không hay vì bạn sẽ phải lặp lại việc validate ở rất nhiều nơi, và điều đó thì đi ngược lại với nguyên tắc lập trình cụ thể ở đây là **DRY**(Don't Repeat Yourself). Và một framework được cho là siêu mạnh như Laravel thì nó đều có cách khắc phục cả.

**1, Tạo Request với FormRequest.**

-Để tạo một FormRequest trong Laravel thì cũng có 2 cách là tạo bằng tay và bằng lệnh mà cái gì nó hỗ trợ rồi thì tội gì mà phải tạo bằng tay :D.

-Cú Pháp tạo Formrequest:

php artisan make:request RequestName

-Trong đó: **RequestName** là Tên Request mà bạn muốn tạo.

VD: Mình tạo một LoginRequest.

php artisan make:request LoginRequest

-Nếu như thành công thì bạn sẽ nhận được thông báo:

created Request successfully

Ngay sau đó bạn truy cập vào **app/Http/Requests** sẽ thấy có một file **LoginRequest.php** có nội dung như sau:

<?php

namespace App\Http\Requests;

use Illuminate\Foundation\Http\FormRequest;

class LoginRequest extends FormRequest

{

/\*\*

\* Determine if the user is authorized to make this request.

\*

\* @return bool

\*/

public function authorize()

{

return false;

}

/\*\*

\* Get the validation rules that apply to the request.

\*

\* @return array

\*/

public function rules()

{

return [

//

];

}

}

Chú ý: Tất cả các file Request phải được đặt trong folder Requests.

**2, Cấu hình Request.**

**Bật tắt Validate.**

-Để bật tắt chức năng validation Request thì các bạn chỉ cần cấu hình giá trị trả về của authorize function:

+TRUE: nếu muốn validation.

+FALSE: nếu không muốn validation.

public function authorize()

{

return true;

}

**Cấu hình chuỗi pattern.**

-Trong FormRequest để cấu hình chuỗi pattern các bạn phải viết trong hàm **rules**

public function rules()

{

return [

//pattern content

];

}

-Và nội dung cú pháp thì hoàn toàn giống với [phần trước](http://toidicode.com/validation-trong-laravel-44.html).

-Chú ý: Tên Hàm là mặc định không được thay đổi.

**VD:**

public function rules()

{

return [

'name'=>'required|max:20',

'age'=>'required|integer|max:20',

];

}

**Thay đổi nội dung báo lỗi.**

-Để thay đổi nội dung lỗi hiển thị thì mọi người cần phải tạo một hàm có name **messages()**và dữ liệu trả về là một mảng chứa nội dung báo lỗi vẫn theo cú pháp của [phần trước](http://toidicode.com/Validation-Trong-Laravel-43).

VD:

public function messages()

{

return [

'required'=>':attribute Không được để trống',

'max'=>':attribute Không được quá :max ký tự',

];

}

-Chú ý: Thay đổi tên Hàm sẽ không hoạt động.

**Thay đổi tên input.**

-Và để thay đổi tên cho input thì mọi người cũng cần phải tạo thêm một hàm có tên **attributes()** và dữ liệu trả về là một mảng chứa tên mới.

VD:

public function attributes(){

return [

'name'=>'Tên',

'age'=>'Tuổi',

];

}

-Chú ý: Thay đổi tên Hàm sẽ không hoạt động.

**3, Sử dụng lớp FormRequest trong Controllers.**

-Để sử dụng lớp FormRequest trong Controllers thì trước hết các bạn phải gọi **namespace** của **Request** đó với cú pháp:

use App\Http\Requests\RequestName;

+Trong đó:RequestName là tên Class Request mà bạn muốn sử dụng.

VD: Mình muốn gọi LoginRequest ở VD trên:

use App\Http\Requests\LoginRequest;

-Tiếp đó hàm nào các bạn muốn sử dụng validation thì chỉ việc ánh xạ nó vào hàm với cú pháp:

public function name(RequestName $request)

{

//

}

 +**Trong đó:**

* **name:** là tên hàm của các bạn.
* **RequestName**: là tên class Request mà các bạn vừa khai báo namespace của nó.
* **$request:**Là biến mà các bạn muốn ánh xạ vào.

**VD:**

public function store(LoginRequest $request)

{

return $request->all();

}