LẬP TRÌNH WEB PHP NÂNG CAO

GV: Trần Thanh Tuấn

Laravel

Nội dung

- Validation

Validation

- Nếu các luật được validate thành công thì sẽ thực hiện các nghiệp vụ một cách bình thường
- ***Ngược lại,*** validate thất bại thì sẽ tạo ra 01 ngoại lệ (Exception) và thông báo lỗi đến cho người dùng một cách tự động. Các lỗi này được đưa vào session.

Validation

- Sử dụng phương thức validate của đối tượng Illuminate\Http\Request

```
$request->validate([
    `ten_dang_nhap' => 'required|max:20',
    `mat_khau' => 'required'
]);
```

Validation

- Hiển thị thông báo lỗi trên view

```
<!-- /resources/views/dang-nhap.blade.php -->

<h1>Đăng Nhập</h1>

@if ($errors->any())
    <div class="alert alert-danger">
        <ul>
            @foreach($errors->all() as $error)
                <li>{{$error }}</li>
            @endforeach
        </ul>
    </div>
@endif
```

Illuminate\Support\
MessageBag

Validation

- Hiển thị thông báo lỗi trên view

```
<!-- /resources/views/dang-nhap.blade.php -->

<h1>Đăng Nhập</h1>

<label for="ten_dang_nhap">Tên đăng nhập</label>

<input id="ten_dang_nhap" type="text"
class="@error(`ten_dang_nhap´) is-invalid @enderror">

@error(`ten_dang_nhap´)
    <div class="alert alert-danger">{{ $message }}</div>
@enderror
```

Kiểm tra lỗi qua "thuộc tính" và thông báo lỗi trong biến $message

Validation

- Nếu validate **Ajax Request** thì sẽ trả về **JSON Response** *(HTTP status code 422)* chứa các lỗi validation

Validation

- Với các kịch bản validation phức tạp hơn thì cần tạo **“form request”**

```
php artisan make:request DangNhapRequest
```

Validation

- Với các kịch bản validation phức tạp hơn thì cần tạo **“form request”**
- Viết các luật validation vào phương thức rules()

```
public function rules()
{
    return [
      `ten_dang_nhap‘ => ‘required|max:20‘,
      `mat_khau‘ => ‘required‘
    ];
}
```

Validation

- Với các kịch bản validation phức tạp hơn thì cần tạo **“form request”**
- “Form request” sẽ được validate trước khi phương thức của controller được “gọi”

```
public function xuLyDangNhap(DangNhapRequest $request)
{
    // Xử lý đăng nhập
}
```

Validation

- Với các kịch bản validation phức tạp hơn thì cần tạo **“form request”**
- Tuỳ chỉnh các thông báo lỗi ==> ghi đè phương thức messages()

```
public function messages()
{
    return [
        `ten_dang_nhap.required‘ => `Chưa nhập tên đăng nhập‘,
        `mat_khau.required‘  => `Chưa nhập mật khẩu‘
    ];
}
```

Validation

- Với các kịch bản validation phức tạp hơn thì cần tạo **"form request"**
- Phương thức authorize(): kiểm tra quyền thao tác của một user đã chứng thực
  - Trả về **false** ==> HTTP status code 403
  - Nếu việc kiểm tra quyền thao tác của user được cài đặt ở **bộ phận** khác thì nên trả về **true**.

```
public function authorize()
{
    return true;
}
```

Validation

- Một số luật validation
  - required
  - numeric
  - nullable
  - min:value
  - max:value
  - digits:value
  - between:min,max
  - email
  - file
  - gt:field
  - gte:field

Validation

- Một số luật validation
  - lt:field
  - lte:field
  - size
  - different:field
  - unique:table,column
  - bail ==> dừng validate các luật còn lại nếu gặp phải lỗi

https://laravel.com/docs/5.8/validation#available-validation-rules

Bài tập

- Tìm hiểu cách sử dụng đối tượng Validator

  https://laravel.com/docs/5.8/validation#manually-creating-validators