Phần 3 – Association

A/Tạo model và migration cho author

php artisan make:model Author --migration

database/migrations/\*create\_authors\_table.php. Mở nó lên và thêm các thuộc tính

public function up()

{

Schema::create('authors', function (Blueprint $table) {

$table->increments('id');

$table->string('name');

$table->string('email');

$table->timestamps();

});

Schema::table('articles', function (Blueprint $table) {

$table->integer('author\_id')->unsigned()->index()->default(1);

$table->foreign('author\_id')

->references('id')

->on('authors');

});

}

Đồng thời chúng ta sẽ update cả bảng articles để tạo foreign key (khóa ngoại) tham chiếu tới bảng authors luôn. Mặc định khóa ngoại này nếu không được thiết lập thì sẽ là 1.

php artisan migrate

B/Thêm danh sách các thuộc tính có thể mass assignment đồng thời xây dựng Relationship

Mở model app/Author.php ra và thêm mảng $fillable và method articles như sau:

class Author extends Model

{

protected $fillable = ['name', 'email'];

public function articles()

{

return $this->hasMany('App\Article');

}

}

Sau đó là model app/Article.php:

class Article extends Model

{

protected $fillable = ['title', 'content', 'author\_id'];

public function author()

{

return $this->belongsTo('App\Author');

}

}

C/Sử dụng Artisan Tinker để tạo mới author và update dữ liệu

php artisan tinker

$author = App\Author::create(['name' => 'Author Name', 'email' => 'email@domain.local']);

App\Article::query()->update(['author\_id' => App\Author::first()->id]);

D/Ứng dụng Laravel 5.2 – Association Controller

php artisan make:controller AuthorsController

Mở app/Http/Controllers/AuthorsController.php lên và update lại code:

<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;

use App\Http\Requests;

use App\Author;

class AuthorsController extends Controller

{

public function index()

{

return Author::all();

}

public function store(Request $request)

{

return Author::create($request->all());

}

public function show(Author $author)

{

return $author;

}

public function update(Request $request, Author $author)

{

$author->update($request->all());

return $author;

}

public function destroy(Author $author)

{

return (string) $author->delete();

}

}

+Add route:

Route::resource('authors', 'AuthorsController');

E/Tạo mới authors request để validate dữ liệu

Cũng giống như phần validate dữ liệu ở phần article, chúng ta tạo mới AuthorRequest như sau:

php artisan make:request AuthorRequest

Mở app/Http/Requests/AuthorRequest.php lên và update lại code:

public function authorize()

{

return true;

}

public function rules()

{

return [

'name' => 'required',

'email' => 'required|email'

];

}

Sau đó chúng ta cần update lại 2 method store và update trong AuthorsController để sử dụng Request vừa tạo. Mở app/Http/Controllers/AuthorsController.php ra và sửa lại:

namespace App\Http\Controllers;

use Illuminate\Http\Request;

use App\Http\Requests\AuthorRequest;

use App\Http\Controllers\Controller;

use App\Author;

class AuthorsController extends Controller

...

public function store(AuthorRequest $request)

...

public function update(AuthorRequest $request, Author $author)

...

}

Thêm author\_id trở thành required trong article validation

Mở app/Http/Requests/ArticleRequest.php ra và update lại rules cho nó:

public function rules()

{

return [

'title' => 'required|min:3',

'content' => 'required',

'author\_id' => 'required',

];

}

Bây giờ khi tạo mới một article thì trường author\_id trở thành bắt buộc