Phần 7 – Ajax CRUD

http://laravel-php.com/laravel-framework/laravel-co-ban/ung-dung-laravel-5-2-hoan-chinh-part-7/

A/Cài đặt jQuery và Rails UJS

Đầu tiên là cài đặt jQuery và jquery-ujs sử dụng Bower. Chi tiết về Bower các bạn xem lại bài Sử dụng Bootstrap nhé. Còn bây giờ thì mở terminal lên và sử dụng các lệnh sau:

bower install jquery --save

bower install jquery-ujs --save

Nhớ lại ở phần 5 của series khi mà chúng ta cài đặt và sử dụng Gulp, ở trong file gulpfile.js chúng ta có dòng code sau:

mix.scripts(bowerFiles('\*\*/\*.js'), 'public/js/3rd-party.js', '/');

Chúng đã đã mix tất cả các file js cài đặt bằng Bower và compile chúng lại thành 1 file duy nhất là 3rd-party.js. Bây giờ khi thực hiện lệnh cài đặt jquery và jquery-ujs, chúng ta không cần phải sửa đổi lại gulpfile.js nữa mà chạy thẳng lệnh gulp luôn.

B/Xây dựng template cho ajax requests

Đã cài đặt xong thư viện js cần thiết để sử dụng ajax rồi, bây giờ tạo cho nó cái template ở đường dẫn resources/views/layouts/ajax.blade.php với nội dung bá đạo như sau:

@yield('content')

a/Tạo modal partial

Tiếp tục, hãy tạo file modal với đường dẫn resources/views/shared/modal.blade.php với nội dung:

<div class="modal" id="modal">

<div class="modal-dialog">

<div class="modal-content">

<div class="modal-body"></div>

</div>

</div>

</div>

D/Thêm modal và js vào HTML layout

Bây giờ mở resources/views/layouts/html.blade.php lên và include những thành phần mới tạo ra ở trên vào:

@include('shared.modal')

<script type="text/javascript" src='/js/3rd-party.js'></script>

<script type="text/javascript" src='/js/app.js'></script>

</body>

</html>

E/Service provider / Sửa method boot của AppServiceProvider

Mở app/Providers/AppServiceProvider.php lên và update lại code như sau:

...

use Illuminate\Support\ServiceProvider;

use Request;

class AppServiceProvider extends ServiceProvider

{

public function boot()

{

view()->composer('\*', function ($view) {

$remote = Request::ajax() ? true : null;

$layout = $remote ? 'layouts.ajax' : 'layouts.html';

$view->with(compact('remote', 'layout'));

});

}

...

Chúng ta vừa thêm một biến remote để xác định là request gửi đến có phải là ajax hay không. Nếu request đó là ajax thì sử dụng ajax layout, còn không thì sử dụng layout html cũ. Đọc thêm về Views Composer ở Documentation.

Tiếp tục, mở resources/views/articles/index.blade.php và thiết lập data-remote thành true như sau:

...

<h1>Articles</h1>

{!! link\_to\_route('articles.create', 'New Article', null, [

'class' => 'btn btn-primary btn-lg',

'data-remote' => 'true' ]) !!}

...

Cũng tương tự với resources/views/articles/create.blade.php. Tuy nhiên ở đây thì ta sẽ thiết lập data-remote tự động đối với Form tạo mới article:

{!! Form::open(['route' => 'articles.store', 'data-remote' => $remote]) !!}

E/Tạo javascript để load nội dung remote

Đầu tiên hãy tạo một thư mục để chứa các file js:

mkdir resources/assets/js

Trong thư mục js trên, tạo mới file loadRemoteContent.js với nội dung:

$(document).on('ajax:success', '.btn[data-remote]', function(e, data, status, xhr) {

var target = $('#modal');

target.find('.modal-body').html(xhr.responseText);

target.modal('show');

target.on('ajax:success', 'form[data-remote]', function(e, data, status, xhr) {

$('#content').html(xhr.responseText);

target.modal('hide');

});

target.on('ajax:error', 'form[data-remote]', function(e, data, status, xhr) {

target.find('#alert-box')

.show()

.find('ul')

.html(toList(data.responseJSON));

});

});

function toList(messages) {

var converted = '';

$.each(messages, function(key, value)

{

converted += '<li>' + value + '</li>';

});

return converted;

}

F/Thêm js vào gulpfile

Mở gulpfile.js ở thư mục gốc lên và update lại:

...

elixir(function(mix) {

...

mix.scriptsIn('', 'public/js/app.js');

});

Sau đó chạy lệnh:

gulp

để biên dịch tất cả ra file app.js.