

LẬP TRÌNH WEB PHP NÂNG CAO

GV: Trần Thanh Tuấn



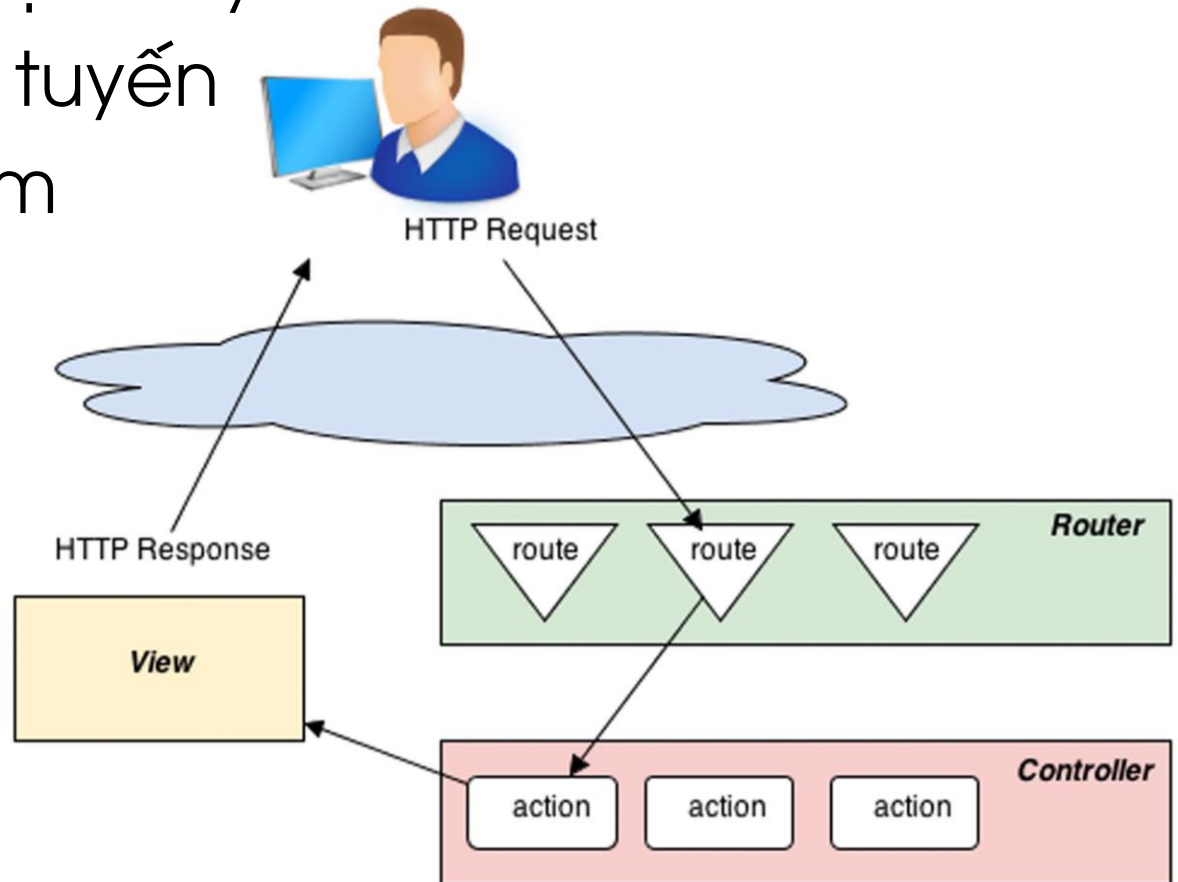
Laravel

Nội dung

- **Routing (Định tuyến)**
- Controller
- View

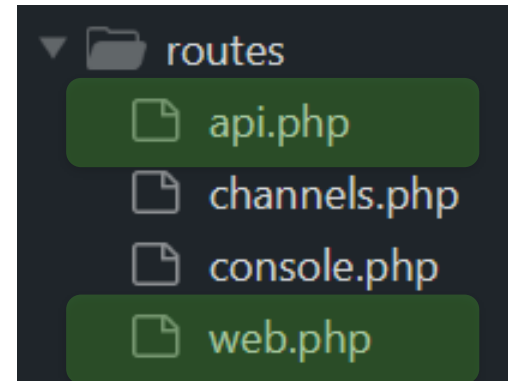
Routing (Định tuyến)

- Cơ bản về định tuyến
- Tham số trong định tuyến
- Định danh định tuyến
- Định tuyến nhóm



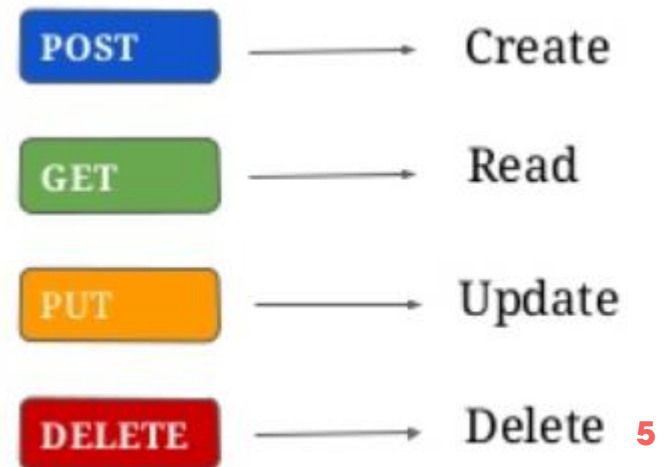
Cơ bản về định tuyến

- Định tuyến trong Laravel:
 - HTTP Request → Đoạn code tương ứng để xử lý (*Controller*)
 - Các route được định nghĩa trong các file định tuyến trong thư mục **routes**
 - **web.php**
 - **api.php**



Cơ bản về định tuyến

- Các cú pháp định tuyến cơ bản:
 - `Route::get($uri, $callback);`
 - `Route::post($uri, $callback);`
 - `Route::put($uri, $callback);`
 - `Route::patch($uri, $callback);`
 - `Route::delete($uri, $callback);`
 - `Route::options($uri, $callback);`



Cơ bản về định tuyến

- Các cú pháp định tuyến cơ bản:
 - Sử dụng 1 định tuyến để phản hồi nhiều HTTP Verb

```
Route::match(['get', 'post'], '/', function () {  
    //  
});
```

```
Route::any('/', function () {  
    //  
});
```

Cơ bản về định tuyến

- Các cú pháp định tuyến cơ bản:
 - Các định tuyến **POST**, **PUT**, **DELETE** được định nghĩa trong **web.php** cần có thêm trường **CSRF** (*Cross Site Request Forgery*) **Token**

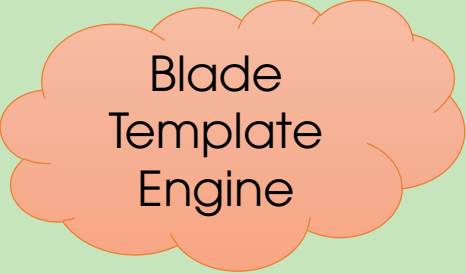
```
<form method="POST" action="/profile">  
    @csrf  
    ...  
</form>
```

Cơ bản về định tuyến

- Các cú pháp định tuyến cơ bản:
 - **Lưu ý:** HTML Form không hỗ trợ **PUT**, **PATCH**, **DELETE**

```
<form action="/foo/bar" method="POST">  
  <input type="hidden" name="_method" value="PUT">  
  <input type="hidden" name="_token" value="{{ csrf_token() }}">  
</form>
```

```
<form action="/foo/bar" method="POST">  
  @method('PUT')  
  @csrf  
</form>
```

The logo for the Blade Template Engine, featuring the text "Blade Template Engine" inside an orange cloud-like shape.

Blade
Template
Engine

Cơ bản về định tuyến

- Định tuyến chuyển hướng:

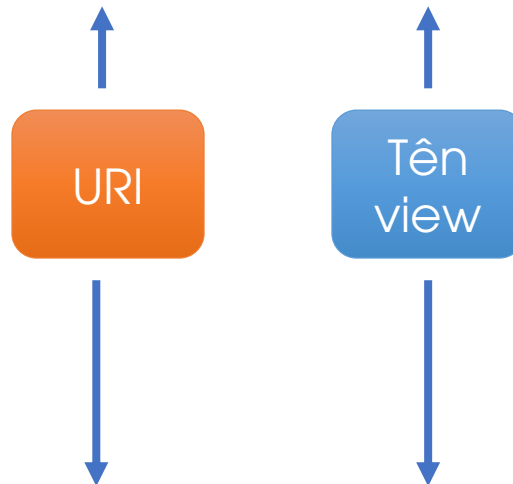
- `Route::redirect('/here', '/there');` // status code 302
- `Route::redirect('/here', '/there', 301);` // status code 301
- `Route::permanentRedirect('/here', '/there');` // status code 301



Cơ bản về định tuyến

- Định tuyến khung nhìn:

- `Route::view('/welcome', 'welcome');`



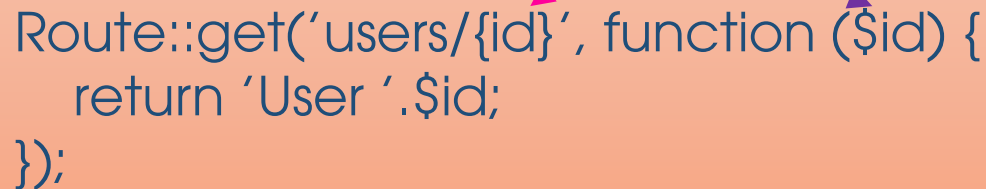
- `Route::view('/welcome', 'welcome', ['name' => 'Taylor']);`

Tham số trong định tuyến

- Cần lấy các đoạn (*segment*) trong **URI**

- `http://domain/users/1`

```
Route::get('users/{id}', function ($id) {  
    return 'User '.$id;  
});
```



- `http://domain/posts/1/comments/5`

```
Route::get('posts/{post}/comments/{comment}', function ($postId,  
    $commentId) {  
    //  
});
```



Tham số trong định tuyến

- Tên tham số trong định tuyến phải:
 - Nằm trong cặp dấu **{ }**
 - Chỉ bao gồm các ký tự **Alphabet**
 - Không sử dụng ký tự gạch ngang – (có thể sử dụng ký tự gạch dưới **_** để thay thế)
- Giá trị các tham số trong route được truyền vào các tham số trong phương thức callback **theo đúng thứ tự**
- Tên tham số trong route và tên tham số trong phương thức callback **không liên quan** gì nhau
 - *{post} vs \$postID, {comment} vs \$commentID*

Tham số trong định tuyến

- Tham số tùy chọn (Optional Parameter):
 - Đặt dấu **?** ở sau tên tham số
 - Khai báo giá trị mặc định cho tham số tương ứng trong phương thức callback

```
Route::get('users/{name?}', function ($name = null) {  
    return $name;  
});
```

```
Route::get('users/{name?}', function ($name = 'John') {  
    return $name;  
});
```

Định danh định tuyến

- Sử dụng phương thức **name** để đặt tên cho định tuyến

```
Route::get('user/profile', function () {  
    //  
})->name('profile');
```

```
Route::get('user/profile', 'UserProfileController@show')->name('profile');
```

Định danh định tuyến

- Sử dụng hàm toàn cục `route()` để tạo URL hoặc chuyển hướng với tên định tuyến

```
// tạo URL...  
$url = route('profile');
```

```
// chuyển hướng...  
return redirect()->route('profile');
```

Định danh định tuyến

- Truyền tham số vào định danh định tuyến
 - Các tham số được truyền vào như là đối số thứ 2 của hàm route()

```
Route::get('user/{id}/profile', function ($id) {  
    //  
})->name('profile');
```

```
$url = route('profile', ['id' => 1]);
```


Nhóm định tuyến

- Nhóm định tuyến cho phép chia sẻ các thuộc tính (namespace, middleware...)
- Sử dụng phương thức `Route::group`

Nhóm định tuyến

- Nhóm middleware:
 - Gán middleware vào tất cả route trong nhóm
 - Sử dụng phương thức **middleware** trước khi định nghĩa nhóm
 - Các middleware được thực thi theo thứ tự trong mảng được truyền vào

```
Route::middleware(['first', 'second'])->group(function () {  
    Route::get('/', function () {  
        // Uses first & second Middleware  
    });  
  
    Route::get('user/profile', function () {  
        // Uses first & second Middleware  
    });  
});
```

Nhóm định tuyến

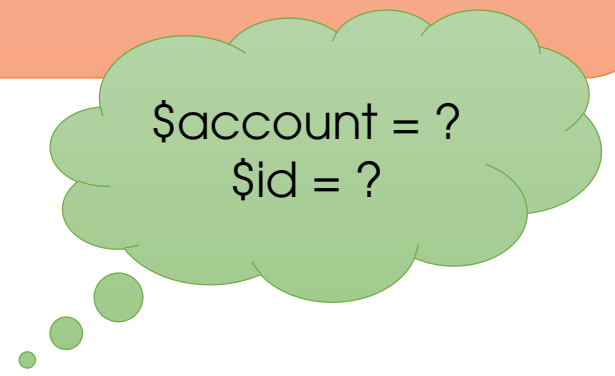
- Nhóm Namespace:
 - Sử dụng phương thức `namespace` để gom nhóm các Controller có cùng PHP namespace

```
Route::namespace('Admin')->group(function () {  
    // Controllers Within The "App\Http\Controllers\Admin" Namespace  
});
```

Nhóm định tuyến

- Định tuyến miền con (sub-domain):
 - Sử dụng phương thức **domain** trước khi định nghĩa nhóm

```
Route::domain('{account}.myapp.com')->group(function () {  
    Route::get('user/{id}', function ($account, $id) {  
        //  
    });  
});
```



\$account = ?
\$id = ?

<http://tttuan.myapp.com/user/1>

Nhóm định tuyến

- Định tuyến tiền tố:
 - Sử dụng phương thức **prefix** để xác định tiền tố trong mỗi route

```
Route::prefix('admin')->group(function () {  
    Route::get('users', function () {  
        // trùng khớp với URL "/admin/users"  
    });  
});
```

Nhóm định tuyến

- Tiền tố định danh:
 - Sử dụng phương thức **name** để xác định tiền tố định danh trong mỗi tên route trong nhóm

```
Route::name('admin')->group(function () {  
    Route::get('users', function () {  
        // route sẽ được gán với tên "admin.users"  
    })->name('users');  
});
```

Bài tập

1. Viết route cho các URL sau

- (GET) `http://domain.com/quan-tri/linh-vuc`
- (POST) `http://domain.com/quan-tri/linh-vuc/them-moi`
- (GET) `http://domain.com/quan-tri/linh-vuc/1`
- (DELETE) `http://domain.com/quan-tri/linh-vuc/1/xoa`
- (GET) `http://domain.com/quan-tri/cau-hoi`
- (POST) `http://domain.com/quan-tri/cau-hoi/them-moi`
- (GET) `http://domain.com/quan-tri/cau-hoi/11`
- (DELETE) `http://domain.com/quan-tri/cau-hoi/11/xoa`

Bài tập

1. Viết route cho các URL sau

- (GET) <http://domain.com/quan-tri/linh-vuc>

```
Route::get('quan-tri/linh-vuc', function () {  
});
```

- (POST) <http://domain.com/quan-tri/linh-vuc/them-moi>

```
Route::post('quan-tri/linh-vuc/them-moi', function () {  
});
```

- (GET) <http://domain.com/quan-tri/linh-vuc/1>

```
Route::get('quan-tri/linh-vuc/{id}', function ($id) {  
});
```


Bài tập

1. Viết route cho các URL sau

- (DELETE) <http://domain.com/quan-tri/linh-vuc/1/xoa>

```
Route::delete('quan-tri/linh-vuc/{id}/xoa', function ($id) {  
});
```

- **Viết tương tự cho các URL sau:**
- (GET) <http://domain.com/quan-tri/cau-hoi>
- (POST) <http://domain.com/quan-tri/cau-hoi/them-moi>
- (GET) <http://domain.com/quan-tri/cau-hoi/11>
- (DELETE) <http://domain.com/quan-tri/cau-hoi/11/xoa>

Bài tập

1. Viết route cho các URL sau

```
Route::get('quan-tri/linh-vuc', function () {
});
Route::post('quan-tri/linh-vuc/them-moi', function () {
});
Route::get('quan-tri/linh-vuc/{id}', function ($id) {
});
Route::delete('quan-tri/linh-vuc/{id}/xoa', function ($id) {
});
Route::get('quan-tri/cau-hoi', function () {
});
Route::post('quan-tri/cau-hoi/them-moi', function () {
});
Route::get('quan-tri/cau-hoi/{id}', function ($id) {
});
Route::delete('quan-tri/cau-hoi/{id}/xoa', function ($id) {
});
```

Bài tập

2. Gom nhóm các route trong câu 1

```
Route::get('quan-tri/linh-vuc', function () {
});
Route::post('quan-tri/linh-vuc/them-moi', function () {
});
Route::get('quan-tri/linh-vuc/{id}', function ($id) {
});
Route::delete('quan-tri/linh-vuc/{id}/xoa', function ($id) {
});
Route::get('quan-tri/cau-hoi', function () {
});
Route::post('quan-tri/cau-hoi/them-moi', function () {
});
Route::get('quan-tri/cau-hoi/{id}', function ($id) {
});
Route::delete('quan-tri/cau-hoi/{id}/xoa', function ($id) {
});
```

Bài tập

2. Gom nhóm các route trong câu 1

```
Route::prefix('quan-tri')->group(function () {  
    Route::get('linh-vuc', function () {  
    });  
    Route::post('linh-vuc/them-moi', function () {  
    });  
    Route::get('linh-vuc/{id}', function ($id) {  
    });  
    Route::delete('linh-vuc/{id}/xoa', function ($id) {  
    });  
    Route::get('cau-hoi', function () {  
    });  
    Route::post('cau-hoi/them-moi', function () {  
    });  
    Route::get('cau-hoi/{id}', function ($id) {  
    });  
    Route::delete('cau-hoi/{id}/xoa', function ($id) {  
    });  
});
```

Bài tập

2. Gom nhóm các route trong câu 1

```
Route::prefix('quan-tri')->group(function () {  
    Route::prefix('linh-vuc')->group(function () {  
        Route::get('/', function () {  
        });  
        Route::post('them-moi', function () {  
        });  
        Route::get('{id}', function ($id) {  
        });  
        Route::delete('{id}/xoa', function ($id) {  
        });  
    });  
    Route::prefix('cau-hoi')->group(function () {  
        Route::get('/', function () {  
        });  
        Route::post('them-moi', function () {  
        });  
        Route::get('{id}', function ($id) {  
        });  
        Route::delete('{id}/xoa', function ($id) {  
        });  
    });  
});
```

Nội dung

- Routing (Định tuyến)
- **Controller**
- View

Controller

- Controller:
 - Hạn chế việc xử lý logic tại các tập tin định tuyến
 - Gom nhóm các yêu cầu xử lý logic có liên quan với nhau thành một Lớp (Class)
 - Các Controller được lưu trữ trong thư mục **app\Http\Controllers**

Controller

```
<?php
```

```
namespace App\Http\Controllers;
```

```
use App\User;
```

```
use App\Http\Controllers\Controller;
```

```
class UserController extends Controller
```

```
{
```

```
    /**
```

```
     * Show the profile for the given user.
```

```
     *
```

```
     * @param int $id
```

```
     * @return View
```

```
     */
```

```
    public function show($id)
```

```
    {
```

```
        return view('user.profile', ['user' => User::findOrFail($id)]);
```

```
    }
```

```
}
```

```
Route::get('users/{id}', 'UserController@show');
```


Controller

- Tạo controller
 - **Cách 1:** Tạo tập tin PHP theo mẫu ví dụ trước vào trong thư mục `app\Http\Controllers`
 - **Cách 2:** Sử dụng artisan

- Tạo controller rỗng

```
php artisan make:controller <TênController>
```

- Tạo controller với các phương thức **CRUD**

```
php artisan make:controller <TênController> --resource
```

Controller

- Tạo controller
 - **Cách 2:** Sử dụng artisan
 - Tạo controller với các phương thức **CRUD** (*Create, Read, Update, Delete*)

```
php artisan make:controller <TênController> --resource
```

Verb	Action
GET	index
GET	create
POST	store
GET	show
GET	edit
PUT/PATCH	update
DELETE	destroy

Nội dung

- Routing (Định tuyến)
- Controller
- **Cơ bản về View**

View

- View:
 - Chứa mã HTML
 - Được lưu trữ trong thư mục resources/views
 - Sử dụng **Blade Template Engine**

```
<html>
  <body>
    <h1>Hello, {{ $name }}</h1>
  </body>
</html>
```

resources/views/greeting.blade.php

```
Route::get('/', function () {
    return view('greeting', ['name' => 'James']);
});
```

View

- View:
 - Sử dụng dấu chấm “.” để biểu diễn các **thư mục / tập tin lồng nhau** trong thư mục `resources/views`

```
<html>
  <body>
    <h1>Hello, {{ $name }}</h1>
  </body>
</html>
```

resources/views/user/greeting.blade.php

```
Route::get('/', function () {
    return view('user.greeting', ['name' => 'James']);
});
```

View

- Truyền dữ liệu vào view:
 - Truyền mảng kết hợp (key – value)

```
Route::get('/', function () {  
    return view('greeting', ['name' => 'James']);  
});
```

- Bên trong view, sử dụng biến tương ứng với **key** truyền vào để lấy giá trị

```
<html>  
  <body>  
    <h1>Hello, {{ $name }}</h1>  
  </body>  
</html>
```

View

- Truyền dữ liệu vào view:
 - Sử dụng phương thức `with()`

```
Route::get('/', function () {  
    return view('greeting')->with('name', 'James');  
});
```

View

- Truyền dữ liệu vào view:
 - Sử dụng phương thức `compact()` để kết hợp các biến và giá trị của chúng thành mảng 1 chiều

```
Route::get('/', function () {  
    $name = 'James';  
    return view('greeting', compact('name'));  
});
```