Atelier 1 Laravel: Introduction à Laravel

Objectif

L'objectif de cette étude de cas est de vous faire manipuler Laravel en vous concentrant sur la configuration des routes, la gestion des réponses, des redirections, et l'utilisation des variables d'environnement, sans inclure l'utilisation de vues (Blade) ni de contrôleurs.

---

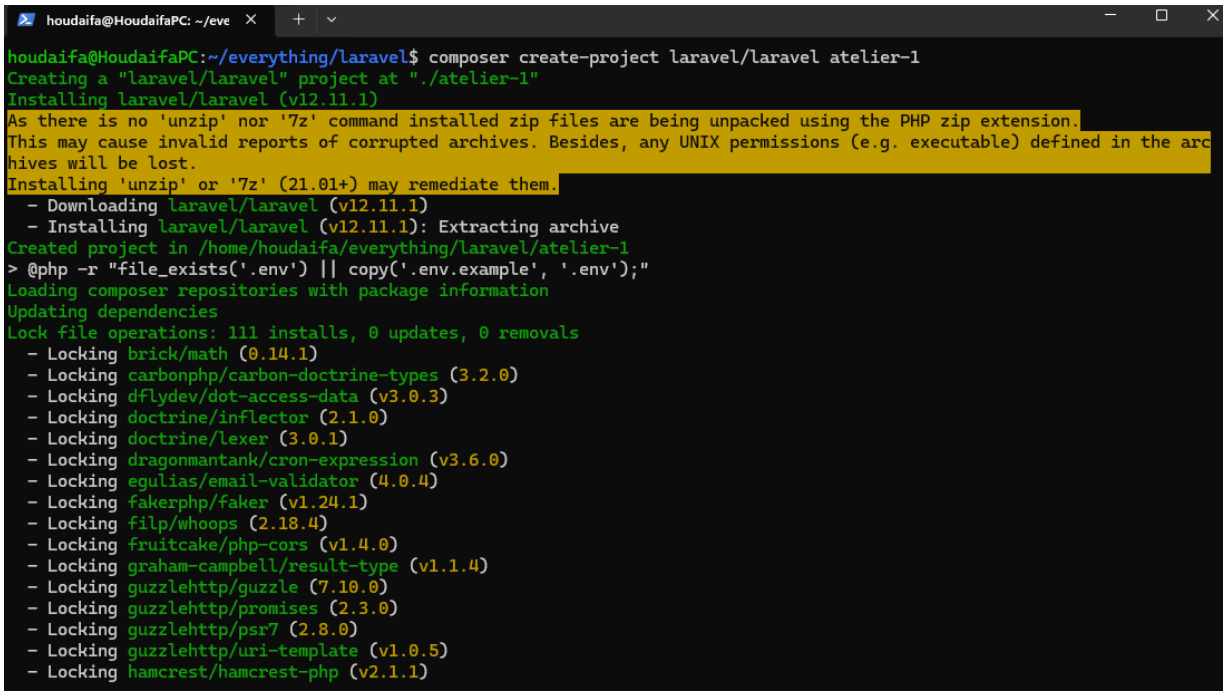
1. Créer un projet Laravel

Objectif : Créer un projet Laravel de base pour démarrer.

Étapes :

1. Ouvrez votre terminal et allez dans le répertoire où vous souhaitez créer le projet.
2. Exécutez la commande suivante pour créer un projet Laravel (remplacez NomDuProjet par le nom que vous souhaitez donner à votre projet) :

composer create-project --prefer-dist laravel/laravel NomDuProjet



3. Une fois le projet créé, naviguez dans le répertoire du projet avec la commande :

cd NomDuProjet

4. Démarrez le serveur de développement Laravel : php

3. Créer et tester des routes dans Laravel

Objectif : Apprendre à définir des routes simples, dynamiques et préfixées sans utiliser de contrôleurs.

3.1 Routes de base

Dans le fichier routes/web.php, ajoutez une route qui retourne une chaîne de texte simple.

web.php

routes > web.php

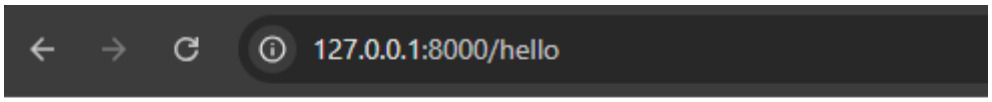
```php
<?php

use Illuminate\Support\Facades\Route;

Route::get('/', function () {
    return view('welcome');
});

Route::get('/hello', function (){
    return 'Hello, laravel!';
});

```

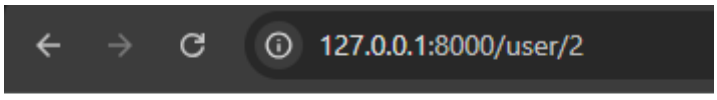


Hello, laravel!

3.2 Route dynamique avec des paramètres

Ajoutez une route qui accepte un paramètre dynamique dans l'URL.

```php
Route::get('user/{id}', function ($id){
    return "User Id: $id";
});
```

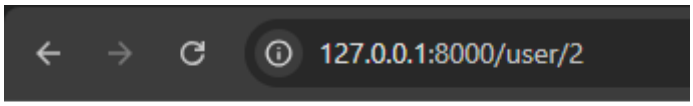


User Id: 2

3.3 Route avec un paramètre optionnel

Ajoutez une route qui accepte un paramètre optionnel.

```php
Route::get('user/{id}/{group?}', function ($id, $group = 'default') {
    return "User ID: {$id}, Group: {$group}";
});
```



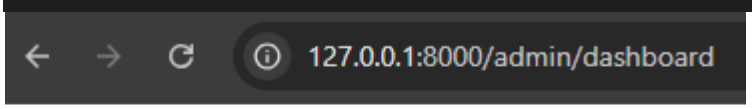
User ID: 2, Group: default

3.4 Route préfixée

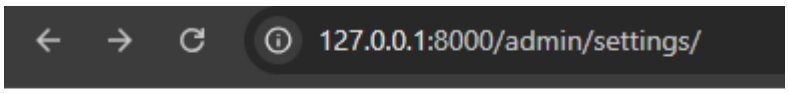
Ajoutez une route préfixée pour un groupe d'URLs sous un même préfixe (admin).

```php
Route::prefix('admin')->group(function () {
    Route::get('dashboard', function () {
        return 'Admin Dashboard';
    });

    Route::get('settings', function () {
        return 'Admin Settings';
    });
});
```



Admin Dashboard



Admin Settings

---

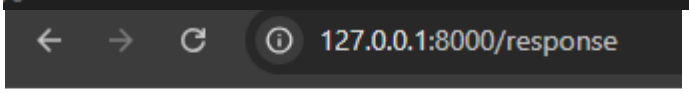
4. Gérer les réponses et redirections

Objectif : Apprendre à retourner des réponses simples, des redirections et des réponses JSON sans vue ni contrôleur.

4.1 Retourner une réponse simple

Ajoutez une route qui retourne une réponse simple.

```php
Route::get('/response', function () {
    return "This is a simple response.";
});
```

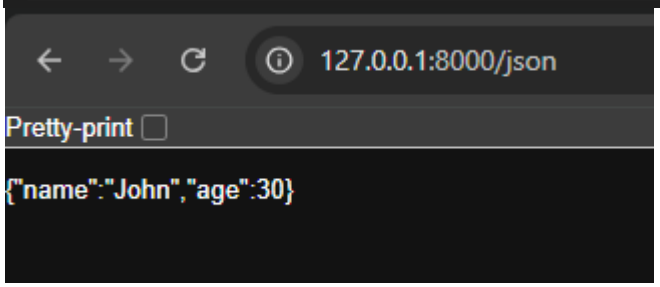


This is a simple response.

4.3 Retourner une réponse JSON

Ajoutez une route qui retourne une réponse JSON, utile pour les APIs.

```php
Route::get('/json', function(){
    return response()->json(['name' => 'John', 'age' => 30]);
});
```



---

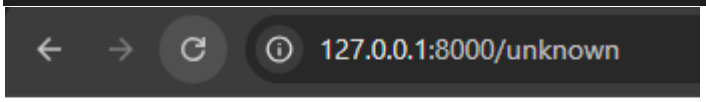
5. Gérer les erreurs 404

Objectif : Apprendre à gérer les erreurs 404 avec la méthode fallback.

5.1 Route Fallback

Dans le fichier routes/web.php, ajoutez une route fallback pour gérer les erreurs 404.

```php
Route::fallback(function () {
    return response("Page Not Found", 404);
});
```

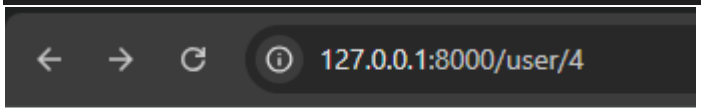


Page Not Found

---

6. Routes avec des conditions

```php
Route::get('/user/{id}', function ($id) {
return "User ID: $id";
})->where('id', '[0-9]+');
```



User ID: 4