**Bài thực hành 14**

**Bảo mật Api với laravel Sanctum**

***Mục tiêu***

* Cài đặt cấu hình Sanctum
* Sử dụng API và Ajax & cách truyền token trên header

**Chú ý:**

* **Hoàn thành labguide 13**

***Thực hành theo các bài tập***

# Bài 1: Cài đặt cấu hình Laravel Sanctum

**Lưu ý** đối với phiên bản Laravel 9 thì Sunctum đã có sãn

Ngoài ra chúng ta có thể cài đặt theo các bước sau

**Bước 1:** mở cmd tại thư mục dự án và chạy lệnh sau

composer require laravel/sanctum

**Bước 2:** Sau khi cài đặt xong thì chạy lệnh sau

php artisan vendor:publish --provider="Laravel\Sanctum\SanctumServiceProvider"

Lệnh trên sẽ copy file migration vào thư mục database/migrations và file cấu hình config/sanctum.php

**Tham khảo nội dung file migration**

2019\_12\_14\_000001\_create\_personal\_access\_tokens\_table.php

File trên sẽ tạo bảng personal\_access\_tokens, xem code của file mgration này

<?php

use Illuminate\Database\Migrations\Migration;

use Illuminate\Database\Schema\Blueprint;

use Illuminate\Support\Facades\Schema;

class CreatePersonalAccessTokensTable extends Migration

{

    /\*\*

     \* Run the migrations.

     \*

     \* @return void

     \*/

    public function up()

    {

        Schema::create('personal\_access\_tokens', function (Blueprint $table) {

            $table->id();

            $table->morphs('tokenable'); // 2 cột tokenable\_type và tokenable\_id

            $table->string('name');

            $table->string('token', 64)->unique();

            $table->text('abilities')->nullable();

            $table->timestamp('last\_used\_at')->nullable();

            $table->timestamps();

        });

    }

    /\*\*

     \* Reverse the migrations.

     \*

     \* @return void

     \*/

    public function down()

    {

        Schema::dropIfExists('personal\_access\_tokens');

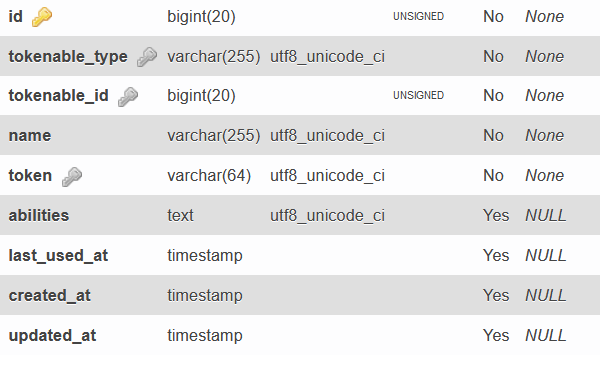
    }

}

**Bước 3:** Chạy lệnh **php artisan migrate** để tạ bảng

php artisan migrate

Bảng sau khi tạo có cấu trúc



Bước 4: Kiểm tra file **app/Http/Kernel.php**, tìm đến đoạn code và bảo đảm như sau

protected $middlewareGroups = [

    'web' => [

        \App\Http\Middleware\EncryptCookies::class,

        \Illuminate\Cookie\Middleware\AddQueuedCookiesToResponse::class,

        \Illuminate\Session\Middleware\StartSession::class,

        // \Illuminate\Session\Middleware\AuthenticateSession::class,

        \Illuminate\View\Middleware\ShareErrorsFromSession::class,

        \App\Http\Middleware\VerifyCsrfToken::class,

        \Illuminate\Routing\Middleware\SubstituteBindings::class,

    ],

    'api' => [

        \Laravel\Sanctum\Http\Middleware\EnsureFrontendRequestsAreStateful::class,

        'throttle:api',

        \Illuminate\Routing\Middleware\SubstituteBindings::class,

    ],

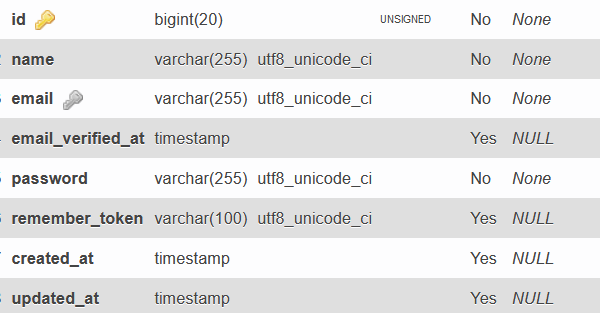
];

# Bài 2: Tạo chức năng đăng ký đăng nhập và tạo mã token

Các Api sẽ tạo

|  |  |  |
| --- | --- | --- |
| **API** | **Http Method** | **Mô tả** |
| **/api/user/login** | POST | Đăng nhập |
| **/api/user/register** | POST | Đăng ký tài khoản |
| **/api/user/logout** | POST | Thoát đăng nhập |
| **/api/user/me** | GET | Thông tin user |

**Bước 1:** Kiểm tra lại bảng users có cấu trúc như sau



**Bước 2:** Tạo các routes phù hợp

Mở file routes/api.pph và khai báo như sau

Route::group(['prefix'=>'user'], function() {

    Route::post('/login', [UserController::class,'login'])->name('user.login');

    Route::post('/register', [UserController::class,'register'])->name('user.register');

    Route::post('/logout', [UserController::class,'logout'])->middleware('auth:sanctum')->name('user.logout');

    Route::get('/me', [UserController::class,'me'])->middleware('auth:sanctum')->name('user.me');

});

**Bước 3:** Tạo UserController trong thư mục app/Http/Controllers/Api

Nội dung file app/HttpControllers/Api/UserController.php gòm các phương thức tương ứng với router như sau

<?php

namespace App\Http\Controllers\Api;

use App\Http\Controllers\Controller;

use Illuminate\Http\Request;

use App\Models\User;

class UserController extends Controller

{

    public function me(Request $request)

    {

        return response()->json([

            'data' => auth()->user(),

            'token' => $request->bearerToken()

        ]);

    }

    public function login(Request $req)

    {

        $data = $req->only('email','password');

        $check = auth()->attempt($data); // kiểm tra đăng nhập

        if ($check) {

            $user = auth()->user(); // lấy về thông tin user trong auth();

            $user->tokens()->delete(); // xóa tất cả các token cũ

            // Tạo token mới

            $token = $user->createToken('Admin-Token')->plainTextToken;

            return response()->json([

                'data' => $user,

                'token' => $token

            ]);

        }

        // báo lỗi nếu thông tin tài khoản không hợp lệ

        return response()->json(['error' => 'Thông tin tài khoản không hợp lệ'] );

    }

    public function register(Request $request)

    {

        $validator = \Validator::make($request->all(), [

            'name' => 'required',

            'email' => 'required|email',

            'password' => 'required',

        ]);

        if($validator->fails()){

            return $this->sendError('Validation Error.', $validator->errors());

        }

        $input = $request->all();

        $input['password'] = bcrypt($input['password']);

        $user = User::create($input);

        return response()->json([

            'data' => $user

        ]);

    }

    public function logout()

    {

        $user = auth('sanctum')->user();

        $user->tokens()->delete(); // xóa hết các token

        auth()->logout();

        return response()->json([

            'data' => null,

            'token' => null

        ]);

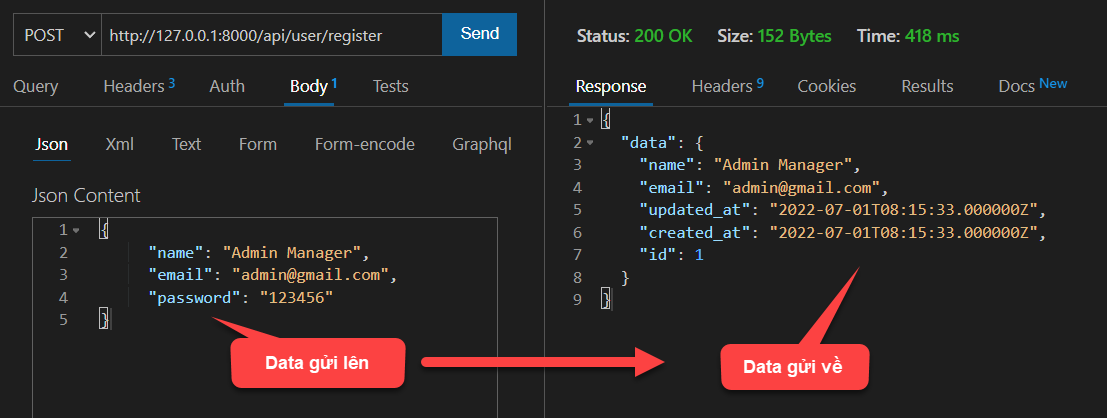
    }

}

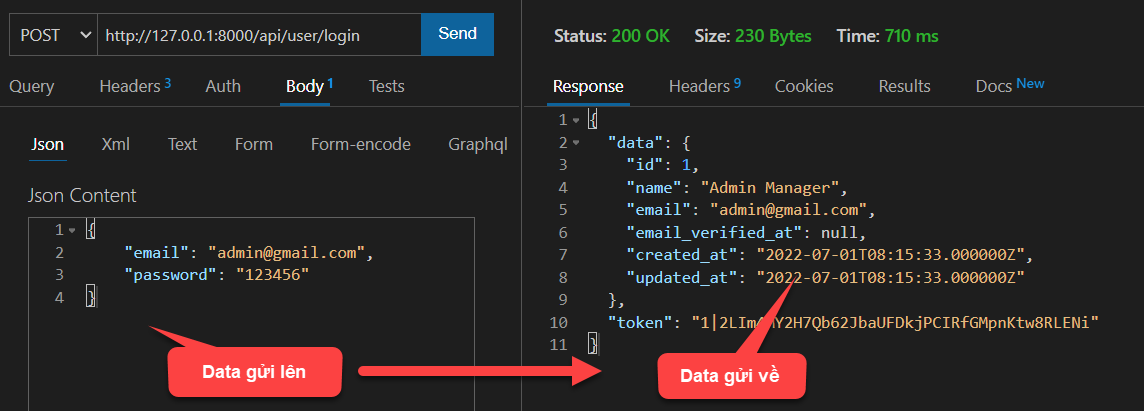
**Bước 3:** Test các Api trên

|  |  |  |
| --- | --- | --- |
| **API** | **Http Method** | **Mô tả** |
| **/api/user/login** | POST | Đăng nhập |
| **/api/user/register** | POST | Đăng ký tài khoản |
| **/api/user/logout** | POST | Thoát đăng nhập |
| **/api/user/me** | GET | Thông tin user |

**POST** api/user/register (Đăng ký tài khoản

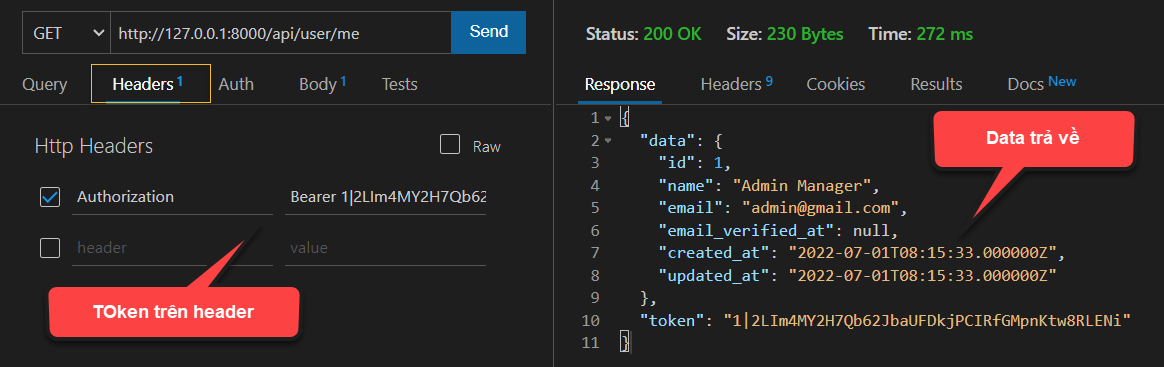


**POST** api/user/login (Đăng nhập tài khoản)



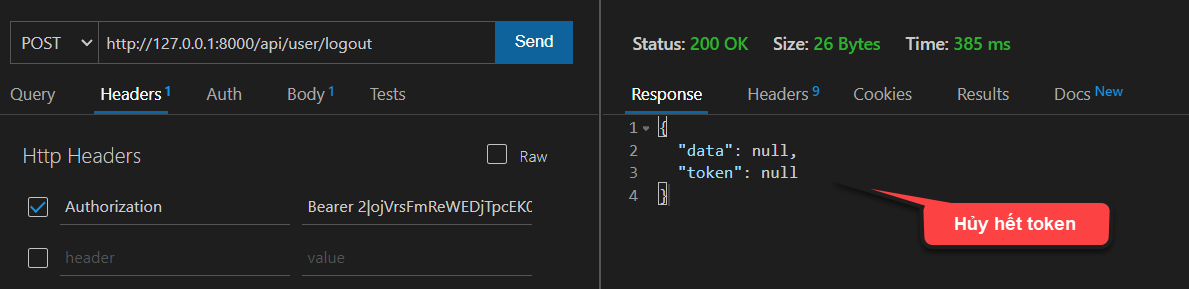
**GET** api/user/me (Thông tin tài khoản)

Api này cần truyền trên header thông tin token đã tạo khi đăng nhập



**POST** api/user/logout (Thoát khỏi tài khoản)

Route này cũng cần truyền vào token



# Bài 2: Bảo mật Api sử dụng mã tone đã tạo theo Sanctum ở trên

**Bước 1:** áp dụng middleware cho các api cần bảo mật

Mở file routes/api.php và sử dụng middleware cho auth:sanctum cho các nhóm api như sau

Route::group(['prefix' => 'categories','middleware' => 'auth:sanctum'], function() {

    Route::get('/', [CategoryController::class,'categories']);

    Route::get('/{id}', [CategoryController::class,'category']);

    Route::post('/', [CategoryController::class,'createCategory']);

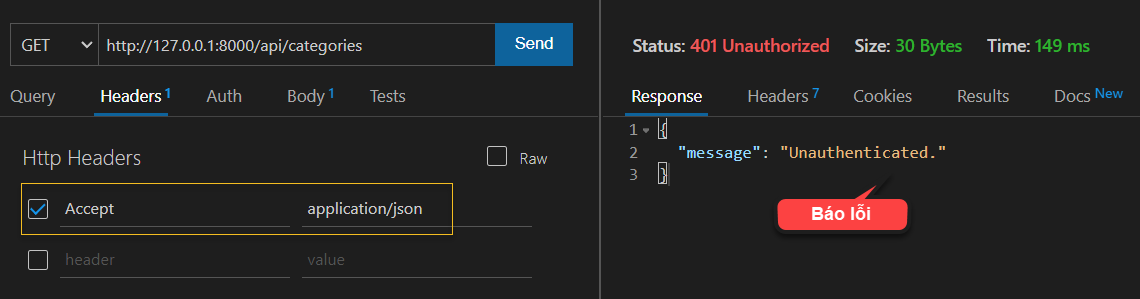
    Route::delete('/{id}', [CategoryController::class,'deleteCategory']);

    Route::put('/{id}', [CategoryController::class,'updateCategory']);

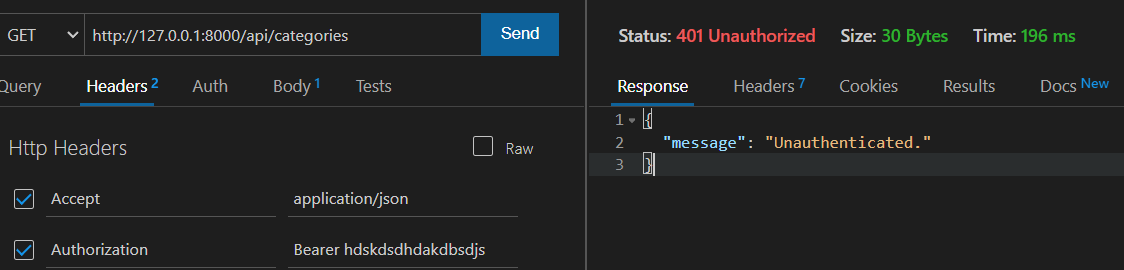
});

Bước 2: Test các api trên với 3 trường hợp

Trường hợp 1: truy cập các api mà không có mã token

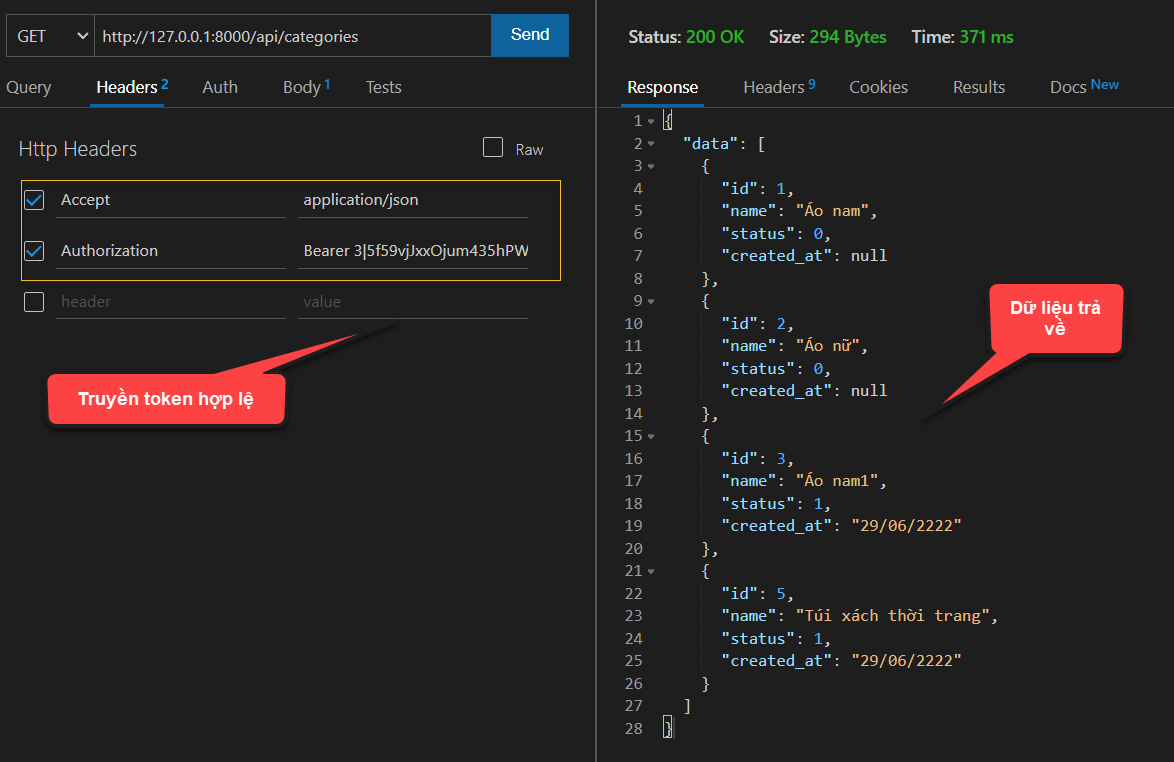


Trường hợp 2: Truyền token không hợp lệ



Trường hợp 3: Truyền đúng mã token (Mã token lấy từ login)

Bearer 3|5f59vjJxxOjum435hPWTmvJwAYNZHEEUwx8MHJUF



# Bài tập làm thêm

Sinh viên thực hiện ajax cho chức năng CRUD của bảng product, customer…