|  |  |
| --- | --- |
|  | Année Universitaire : 2023-2024  **Applications Web Avancées** |
| **Atelier 5 : Authentification – Spatie**  **Gestion des rôles et permissions** | |

**Objectifs**

Le but de cet atelier est d’installer, configurer, personnaliser le package Spatie pour gérer d'authentification dans Laravel.

Le package spatie permet de créer des ACL dans laravel 9. Il permet de définir comment attribuer un rôle à un utilisateur, d’attribuer une permission à un utilisateur et d’affecter une permission à un rôle.

**Prérequis logiciel**

* Projet Laravel créé.
* Création de la base de données
* Gestion du CRUD d’un modèle
  1. **Installation des packages avec composer**

1. Créer un nouveau projet Laravel. **composer create-project laravel/laravel authSpatie**
2. Créer la base de données « authentification »
3. Installer Spatie : **composer require spatie/laravel-permission**
4. Installer le package de collecte de formulaires:

**composer require laravelcollective/html**

1. installer les dépendances : **npm install**
2. Lancer la commande**: npm run dev**
3. Publier PermissionServiceProvider et Obtenir le fichier de configuration dans config/permission.php et les fichiers de migration.

**php artisan vendor:publish --provider="Spatie\Permission\PermissionServiceProvider"**

1. Exécuter les migrations
   1. **Migration des produits**
2. Lancer la commande suivante

|  |
| --- |
| php artisan make :migration create\_products\_table |

Puis ajoutez le code source suivants à la migration des produits**.**

|  |
| --- |
| public function up()      {          Schema::create('products', function (Blueprint $table) {              $table->id();              $table->string('name');              $table->text('detail');              $table->timestamps();          });      } |

Ensuite lancer la commande suivante

|  |
| --- |
| php artisan migrate |

* 1. **La création des modèles**

Dans cette étape, nous devons créer un modèle pour la table User et Product, donc si vous avez un nouveau projet, vous avez le modèle User, il suffit de remplacer le code et d'autres que vous devez créer.

**Product.php**

|  |
| --- |
| protected $fillable = [          'name', 'detail'      ]; |

**User.php**

|  |
| --- |
| protected $fillable = [          'name',          'email',          'password',      ]; |

* 1. **Ajouter les middlewares**

Le package Spatie fournit son middleware intégré, ce qui nous permet de l'utiliser simplement et de l'afficher comme ci-dessous. Donc, nous devons ajouter le middleware dans le fichier Kernel.php de cette façon :

|  |
| --- |
| protected $routeMiddleware = [          …….          'role' => \Spatie\Permission\Middlewares\RoleMiddleware::class,          'permission' => \Spatie\Permission\Middlewares\PermissionMiddleware::class,          'role\_or\_permission' => \Spatie\Permission\Middlewares\RoleOrPermissionMiddleware::class,      ]; |

* 1. **Authentification**

nous allons créer une authentification pour notre application. Lancer la commande suivante pour créer une authentification.

|  |
| --- |
| composer require laravel/ui  npm install  npm run build |

* 1. **Creation des routes**

En fait, nous allons créer des routes pour le module des utilisateurs, le module des produits et le module des rôles. Nous avons également utilisé un middleware pour nos routes. Il suffit d'ajouter la route comme indiqué ci-dessous.

|  |
| --- |
| **….**  Route::group(['middleware' => ['auth']], function() {      Route::resource('roles', RoleController::class);      Route::resource('users', UserController::class);      Route::resource('products', ProductController::class);  }); |

* 1. Ajouter les contrôleurs

Créez un contrôleur dans app/Http/Controllers/ comme indiqué ci-dessous :

**UserController :**

|  |
| --- |
| <?php    namespace App\Http\Controllers;    use Illuminate\Http\Request;  use App\Http\Controllers\Controller;  use App\Models\User;  use Spatie\Permission\Models\Role;  use DB;  use Hash;  use Illuminate\Support\Arr;    class UserController extends Controller  {      /\*\*       \* Display a listing of the resource.       \*       \* @return \Illuminate\Http\Response       \*/      public function index(Request $request)      {          $data = User::orderBy('id','DESC')->paginate(5);          return view('users.index',compact('data'))              ->with('i', ($request->input('page', 1) - 1) \* 5);      }        /\*\*       \* Show the form for creating a new resource.       \*       \* @return \Illuminate\Http\Response       \*/      public function create()      {          $roles = Role::pluck('name','name')->all();          return view('users.create',compact('roles'));      }        /\*\*       \* Store a newly created resource in storage.       \*       \* @param  \Illuminate\Http\Request  $request       \* @return \Illuminate\Http\Response       \*/      public function store(Request $request)      {          $this->validate($request, [              'name' => 'required',              'email' => 'required|email|unique:users,email',              'password' => 'required|same:confirm-password',              'roles' => 'required'          ]);            $input = $request->all();          $input['password'] = Hash::make($input['password']);            $user = User::create($input);          $user->assignRole($request->input('roles'));            return redirect()->route('users.index')                          ->with('success','User created successfully');      }        /\*\*       \* Display the specified resource.       \*       \* @param  int  $id       \* @return \Illuminate\Http\Response       \*/      public function show($id)      {          $user = User::find($id);          return view('users.show',compact('user'));      }        /\*\*       \* Show the form for editing the specified resource.       \*       \* @param  int  $id       \* @return \Illuminate\Http\Response       \*/      public function edit($id)      {          $user = User::find($id);          $roles = Role::pluck('name','name')->all();          $userRole = $user->roles->pluck('name','name')->all();            return view('users.edit',compact('user','roles','userRole'));      }        /\*\*       \* Update the specified resource in storage.       \*       \* @param  \Illuminate\Http\Request  $request       \* @param  int  $id       \* @return \Illuminate\Http\Response       \*/      public function update(Request $request, $id)      {          $this->validate($request, [              'name' => 'required',              'email' => 'required|email|unique:users,email,'.$id,              'password' => 'same:confirm-password',              'roles' => 'required'          ]);            $input = $request->all();          if(!empty($input['password'])){              $input['password'] = Hash::make($input['password']);          }else{              $input = Arr::except($input,array('password'));          }            $user = User::find($id);          $user->update($input);          DB::table('model\_has\_roles')->where('model\_id',$id)->delete();            $user->assignRole($request->input('roles'));            return redirect()->route('users.index')                          ->with('success','User updated successfully');      }        /\*\*       \* Remove the specified resource from storage.       \*       \* @param  int  $id       \* @return \Illuminate\Http\Response       \*/      public function destroy($id)      {          User::find($id)->delete();          return redirect()->route('users.index')                          ->with('success','User deleted successfully');      }  } |

**RoleController**

|  |
| --- |
| <?php    namespace App\Http\Controllers;  use Illuminate\Http\Request;  use App\Http\Controllers\Controller;  use Spatie\Permission\Models\Role;  use Spatie\Permission\Models\Permission;  use DB;    class RoleController extends Controller  {      /\*\*       \* Display a listing of the resource.       \*       \* @return \Illuminate\Http\Response       \*/      function \_\_construct()      {           $this->middleware('permission:role-list|role-create|role-edit|role-delete', ['only' => ['index','store']]);           $this->middleware('permission:role-create', ['only' => ['create','store']]);           $this->middleware('permission:role-edit', ['only' => ['edit','update']]);           $this->middleware('permission:role-delete', ['only' => ['destroy']]);      }        /\*\*       \* Display a listing of the resource.       \*       \* @return \Illuminate\Http\Response       \*/      public function index(Request $request)      {          $roles = Role::orderBy('id','DESC')->paginate(5);          return view('roles.index',compact('roles'))              ->with('i', ($request->input('page', 1) - 1) \* 5);      }        /\*\*       \* Show the form for creating a new resource.       \*       \* @return \Illuminate\Http\Response       \*/      public function create()      {            $permission = Permission::get();          //dd($permission);          return view('roles.create',compact('permission'));      }        /\*\*       \* Store a newly created resource in storage.       \*       \* @param  \Illuminate\Http\Request  $request       \* @return \Illuminate\Http\Response       \*/      public function store(Request $request)      {          $this->validate($request, [              'name' => 'required|unique:roles,name',              'permission' => 'required',          ]);            $role = Role::create(['name' => $request->input('name')]);          $role->syncPermissions($request->input('permission'));            return redirect()->route('roles.index')                          ->with('success','Role created successfully');      }      /\*\*       \* Display the specified resource.       \*       \* @param  int  $id       \* @return \Illuminate\Http\Response       \*/      public function show($id)      {          $role = Role::find($id);          $rolePermissions = Permission::join("role\_has\_permissions","role\_has\_permissions.permission\_id","=","permissions.id")              ->where("role\_has\_permissions.role\_id",$id)              ->get();            return view('roles.show',compact('role','rolePermissions'));      }        /\*\*       \* Show the form for editing the specified resource.       \*       \* @param  int  $id       \* @return \Illuminate\Http\Response       \*/      public function edit($id)      {          $role = Role::find($id);          $permission = Permission::get();          $rolePermissions = DB::table("role\_has\_permissions")->where("role\_has\_permissions.role\_id",$id)              ->pluck('role\_has\_permissions.permission\_id','role\_has\_permissions.permission\_id')              ->all();            return view('roles.edit',compact('role','permission','rolePermissions'));      }        /\*\*       \* Update the specified resource in storage.       \*       \* @param  \Illuminate\Http\Request  $request       \* @param  int  $id       \* @return \Illuminate\Http\Response       \*/      public function update(Request $request, $id)      {          $this->validate($request, [              'name' => 'required',              'permission' => 'required',          ]);            $role = Role::find($id);          $role->name = $request->input('name');          $role->save();            $role->syncPermissions($request->input('permission'));            return redirect()->route('roles.index')                          ->with('success','Role updated successfully');      }      /\*\*       \* Remove the specified resource from storage.       \*       \* @param  int  $id       \* @return \Illuminate\Http\Response       \*/      public function destroy($id)      {          DB::table("roles")->where('id',$id)->delete();          return redirect()->route('roles.index')                          ->with('success','Role deleted successfully');      }  } |

**ProductController :**

|  |
| --- |
| <?php    namespace App\Http\Controllers;    use App\Models\Product;  use Illuminate\Http\Request;    class ProductController extends Controller  {      /\*\*       \* Display a listing of the resource.       \*       \* @return \Illuminate\Http\Response       \*/      function \_\_construct()      {           $this->middleware('permission:product-list|product-create|product-edit|product-delete', ['only' => ['index','show']]);           $this->middleware('permission:product-create', ['only' => ['create','store']]);           $this->middleware('permission:product-edit', ['only' => ['edit','update']]);           $this->middleware('permission:product-delete', ['only' => ['destroy']]);      }      /\*\*       \* Display a listing of the resource.       \*       \* @return \Illuminate\Http\Response       \*/      public function index()      {          $products = Product::latest()->paginate(5);          return view('products.index',compact('products'))              ->with('i', (request()->input('page', 1) - 1) \* 5);      }        /\*\*       \* Show the form for creating a new resource.       \*       \* @return \Illuminate\Http\Response       \*/      public function create()      {          return view('products.create');      }        /\*\*       \* Store a newly created resource in storage.       \*       \* @param  \Illuminate\Http\Request  $request       \* @return \Illuminate\Http\Response       \*/      public function store(Request $request)      {          request()->validate([              'name' => 'required',              'detail' => 'required',          ]);            Product::create($request->all());            return redirect()->route('products.index')                          ->with('success','Product created successfully.');      }        /\*\*       \* Display the specified resource.       \*       \* @param  \App\Product  $product       \* @return \Illuminate\Http\Response       \*/      public function show(Product $product)      {          return view('products.show',compact('product'));      }        /\*\*       \* Show the form for editing the specified resource.       \*       \* @param  \App\Product  $product       \* @return \Illuminate\Http\Response       \*/      public function edit(Product $product)      {          return view('products.edit',compact('product'));      }        /\*\*       \* Update the specified resource in storage.       \*       \* @param  \Illuminate\Http\Request  $request       \* @param  \App\Product  $product       \* @return \Illuminate\Http\Response       \*/      public function update(Request $request, Product $product)      {           request()->validate([              'name' => 'required',              'detail' => 'required',          ]);            $product->update($request->all());            return redirect()->route('products.index')                          ->with('success','Product updated successfully');      }        /\*\*       \* Remove the specified resource from storage.       \*       \* @param  \App\Product  $product       \* @return \Illuminate\Http\Response       \*/      public function destroy(Product $product)      {          $product->delete();            return redirect()->route('products.index')                          ->with('success','Product deleted successfully');      }  } |

* 1. **Ajouter les fichiers blade nécessaire**
  2. **Création de seeding pour la permission et admin user**

nous allons maintenant créer un seeder pour les permissions. Pour l'instant, nous avons des permissions fixes et nous créons un seeder comme indiqué ci-dessous, mais vous pouvez ajouter d'autres permissions si vous le souhaitez.

|  |
| --- |
| public function run()      {          $permissions = [             'role-list',             'role-create',             'role-edit',             'role-delete',             'product-list',             'product-create',             'product-edit',             'product-delete'          ];            foreach ($permissions as $permission) {               Permission::create(['name' => $permission]);          }      } |

En suite lancer la commande suivante :

|  |
| --- |
| php artisan make:seeder PermissionTableSeeder  php artisan db:seed --class=PermissionTableSeeder |

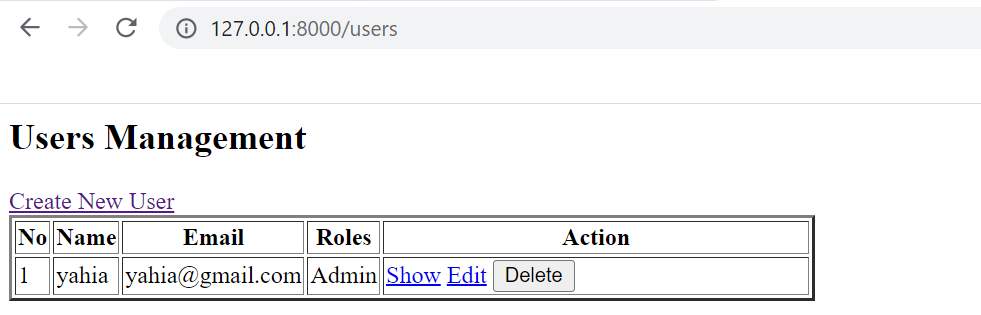
De même on va créer le seeder pour la table adminuser

|  |
| --- |
| public function run()      {          $user = User::create([              'name' => 'yahia',              'email' => yahia@gmail.com',              'password' => bcrypt('123456')          ]);            $role = Role::create(['name' => 'Admin']);            $permissions = Permission::pluck('id','id')->all();            $role->syncPermissions($permissions);            $user->assignRole([$role->id]);      } |

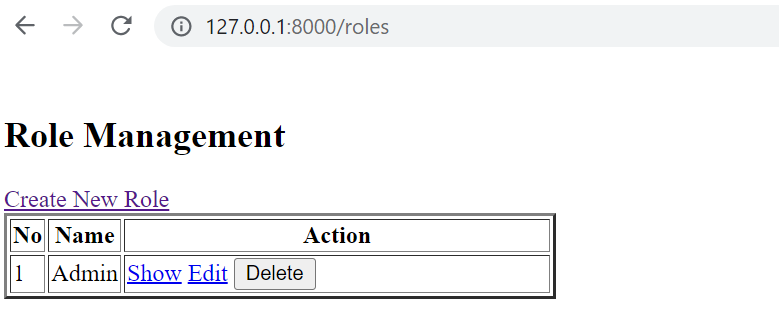
En fin lacer la commande suivante :

|  |
| --- |
| php artisan db:seed --class=CreateAdminUserSeeder |

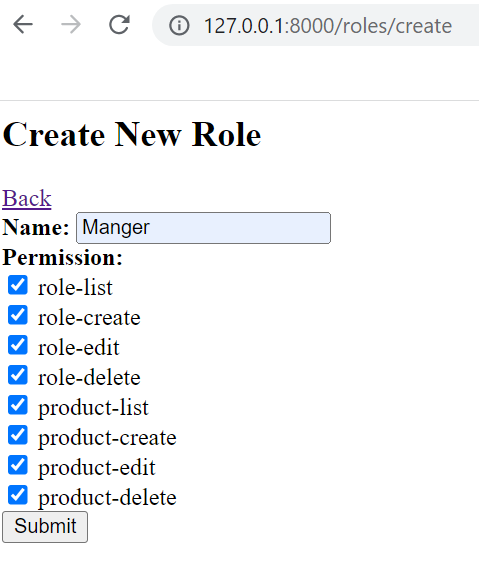
**Gestion utilisateur**



**Gestion des Roles**



**Ajouter un rôle**

****