78. Crea tu propio middleware en Laravel para controlar el acceso administrativo

Middleware

Todo middleware (App\Middleware) tiene un método llamado **handle()** que siempre es llamado cuando el middleware se ejecuta. A través de la petición y el Closure, el método decide si va a dar continuidad a la petición o no.

El middleware es entonces como **una compuerta** que determina si procede con la petición (deja que continúe hacia otro middleware) o suspende todo (respuesta 403, 404, mensaje error, etc).

En cmd:

```
php artisan make:middleware CheckIfAdmin
```

En App\Http\Middleware\CheckIfAdmin:

```
    public function handle(Request $request, Closure $next)
    {
        // Si el usuario no es nulo y además es admin, se continúa
        // (se abre la compuerta, por así decirlo)
        if($request->user() != null && $request->user()->isAdmin()) {
           return $next($request);
        }
        // En otro caso:
        abort(403);
    }
```

En App\Providers\RouteServiceProvider:

Por ahora, antes de ir al kernel, **nos inventamos el nombre del middleware en la función de mapeo**. Sin embargo, tendremos que acordarnos para relacionarlo con el Middleware al que queremos llamar.

Lo llamamos `is.admin`.

```
        protected function mapPanelRoutes() {
                Route::prefix('panel')
                        ->middleware(['web', 'auth', 'is.admin'])
                        // Para mejor organización
                        ->namespace($this->namespace)
                        ->group(base_path('routes/panel.php'));
        }
```

En App\Http\Middleware\Kernel.php:

> **Están ordenados alfabéticamente**
>
> Si vamos a poner uno nuevo, estaría bien seguir el orden, ¿no?

```
...
    protected $routeMiddleware = [
        'auth' => \App\Http\Middleware\Authenticate::class,
        'auth.basic' => \Illuminate\Auth\Middleware\AuthenticateWithBasicAuth::class,
        'cache.headers' => \Illuminate\Http\Middleware\SetCacheHeaders::class,
        'can' => \Illuminate\Auth\Middleware\Authorize::class,
        'guest' => \App\Http\Middleware\RedirectIfAuthenticated::class,
        // Aquí está el wapo
        'is.admin' => \App\Http\Middleware\CheckIfAdmin::class,
        'password.confirm' => \Illuminate\Auth\Middleware\RequirePassword::class,
        'signed' => \Illuminate\Routing\Middleware\ValidateSignature::class,
        'throttle' => \Illuminate\Routing\Middleware\ThrottleRequests::class,
        'verified' => \Illuminate\Auth\Middleware\EnsureEmailIsVerified::class,
    ];
...
```

Prueba

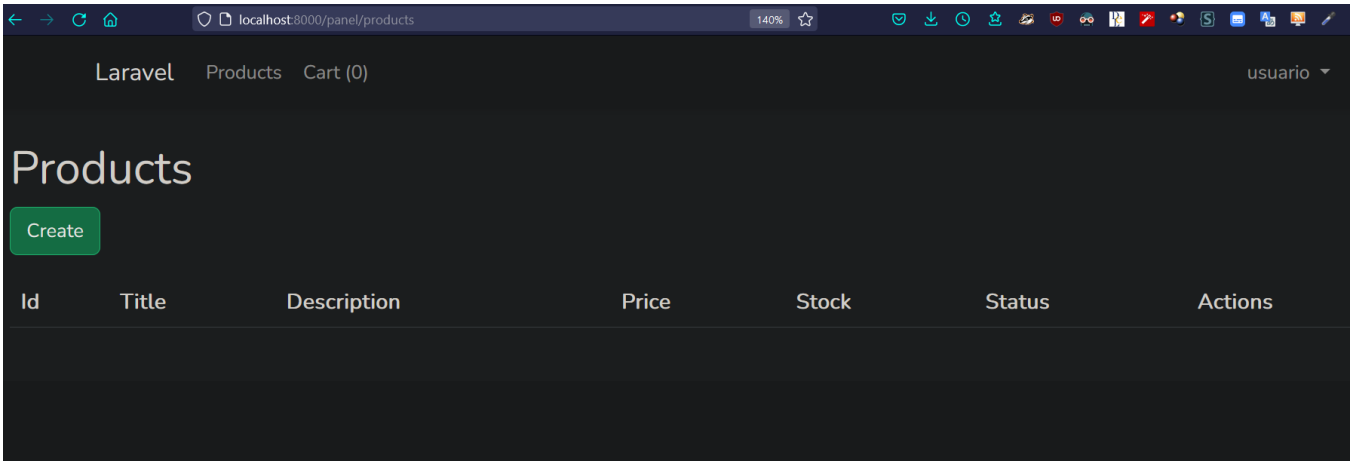
```
>>> $user = App\Models\User::find(21)
=> App\Models\User {#4553
     id: "21",
     name: "usuario",
     email: "usuario@gmail.com",
     email_verified_at: null,
     #password: "$2y$10$d8EwouOCREnLfXaUpl90ZOkEJpV43yCHJHTipc04zg75XdHAS/t.C",
     admin_since: null,
     #remember_token: null,
     created_at: "2023-01-15 18:03:59",
     updated_at: "2023-01-15 18:03:59",
   }

>>> $user->forceFill(['admin_since' => now()])
=> App\Models\User {#4553
     id: "21",
     name: "usuario",
     email: "usuario@gmail.com",
     email_verified_at: null,
     #password: "$2y$10$d8EwouOCREnLfXaUpl90ZOkEJpV43yCHJHTipc04zg75XdHAS/t.C",
     admin_since: "2023-01-15 18:06:42",
     #remember_token: null,
     created_at: "2023-01-15 18:03:59",
     updated_at: "2023-01-15 18:03:59",
   }

>>> $user->isAdmin()
=> true

>>> $user->save()
=> true

// Yyy ahora si vamos a localhost:8000, iniciamos sesión con usuario y pulsamos productos,
debería aparecer una pantalla tal que así (no he usado el seeder, por eso no hay productos,
sé que soy subnormal):
```


localhost:8000/panel/products
140%
Laravel
Products
Cart (0)
usuario
Products
Create
Id
Title
Description
Price
Stock
Status
Actions