

EJ15:Multi Autenticación

1. Continuamos desde donde lo dejamos en el ejercicio 14.
2. Primero, debemos crear un nuevo campo en la base de datos que registre si un usuario es administrador o un usuario normal. Para ello, vamos a la carpeta migrations y en la migration del user añadimos un nuevo campo llamado admin. Será un booleano: "true" es administrador y "false" es usuario.

```
public function up()
{
    Schema::create('users', function (Blueprint $table) {
        $table->id();
        $table->string('name');
        $table->string('nick')->unique();
        $table->timestamp('email_verified_at')->nullable();
        $table->string('password');
        $table->rememberToken();
        $table->timestamps();
        $table->boolean('admin');
    });
}
```

Escribimos en la terminal el comando `php artisan migrate:refresh` para que se actualice la base de datos con el campo nuevo introducido

```
PS C:\Users\Marcos\Documents\DAW\DWES\ProyectosLaravel> php artisan migrate:refresh
```

Además, insertamos dos usuarios: uno como administrador y otro como usuario normal para realizar las pruebas.

	id	name	nick	email_verified_at	password	remember_token	created_at	updated_at	admin
<input type="checkbox"/>	1	Marcos	Marc	NULL	\$2y\$10\$73aulkMrKW2mVLeO00oMI.PQ8i3iI3Uwzn3y6PIQ9O...	NULL	2022-12-27 16:45:06	2022-12-27 16:45:06	1
<input type="checkbox"/>	2	Gonzalo	Gonzalo	NULL	\$2y\$10\$We25HhkvAdHjEdLkiRe7 b1 L52MGITbrtNU9hpOZr...	NULL	2022-12-27 17:53:34	2022-12-27 17:53:34	0

3. Creamos un middleware llamado "isAdmin" que se encargará de filtrar las peticiones que provengan de administradores o de usuarios. Si el usuario que realiza la petición es administrador continuará revisando los siguientes middlewares, sino retornara a la página home del usuario.

```

1 reference | 0 implementations
class isAdmin
{
    /**
     * Handle an incoming request.
     *
     * @param \Illuminate\Http\Request $request
     * @param \Closure(\Illuminate\Http\Request): (\Illuminate\Http\Response|\Illuminate\Http\RedirectResponse) $next
     * @return \Illuminate\Http\Response|\Illuminate\Http\RedirectResponse
     */
    0 references | 0 overrides
    public function handle(Request $request, Closure $next)
    {
        if(Auth::check() && Auth::user()->admin == true){
            return $next($request);
        }

        return redirect("home");
    }
}

```

Registramos el middleware en el kernel con el nombre “isAdmin”.

```

4 references
protected $routeMiddleware = [
    'auth' => \App\Http\Middleware\Authenticate::class,
    'auth.basic' => \Illuminate\Auth\Middleware\AuthenticateWithBasicAuth::class,
    'auth.session' => \Illuminate\Session\Middleware\AuthenticateSession::class,
    'cache.headers' => \Illuminate\Http\Middleware\SetCacheHeaders::class,
    'can' => \Illuminate\Auth\Middleware\Authorize::class,
    'guest' => \App\Http\Middleware\RedirectIfAuthenticated::class,
    'password.confirm' => \Illuminate\Auth\Middleware\RequirePassword::class,
    'signed' => \App\Http\Middleware\ValidateSignature::class,
    'throttle' => \Illuminate\Routing\Middleware\ThrottleRequests::class,
    'verified' => \Illuminate\Auth\Middleware\EnsureEmailIsVerified::class,
    'verifiednick' => \App\Http\Middleware\EnsurePhoneIsVerified::class,
    'isAdmin' => \App\Http\Middleware\isAdmin::class,
];

```

4. A continuación, creamos otra vista “homeAdmin” parecida al home del usuario, pero será el home que verá el administrador al loguearse. Escribimos un mensaje diferente para saber que se ha logueado como administrador.

```

@extends('layouts.app')

@section('content')
<div class="container">
    <div class="row justify-content-center">
        <div class="col-md-8">
            <div class="card">
                <div class="card-header">{{ __('Dashboard') }}</div>

                <div class="card-body">
                    @if (session('status'))
                        <div class="alert alert-success" role="alert">
                            {{ session('status') }}
                        </div>
                    @endif

                    {{ __('You are logged in as an admin!') }}
                </div>
            </div>
        </div>
    </div>
</div>
@endsection

```

Vista administrador

```

@extends('layouts.app')

@section('content')
<div class="container">
    <div class="row justify-content-center">
        <div class="col-md-8">
            <div class="card">
                <div class="card-header">{{ __('Dashboard') }}</div>

                <div class="card-body">
                    @if (session('status'))
                        <div class="alert alert-success" role="alert">
                            {{ session('status') }}
                        </div>
                    @endif

                    {{ __('You are logged in!') }}
                </div>
            </div>
        </div>
    </div>
</div>
@endsection

```

Vista usuario

5. En el controlador HomeController, creamos una función que nos redirige a la vista del “homeAdmin”.

```

0 references | 0 overrides
public function adminHome(){
    return view('homeAdmin');
}

```

6. Creamos la ruta “admin/home” que llama a la función “adminHome” del “HomeController”.

```

Route::get('admin/home', [App\Http\Controllers\HomeController::class, 'adminHome'])->name('admin.home')->middleware("isAdmin");

```

A la ruta, le aplicamos el middleware “isAdmin”, de manera que antes de llamar a la función comprobará si el usuario logueado es un administrador.

7. Por último, hay que actualizar el login del LoginController, de manera que cuando el administrador se logee le redirige a su home, y si se logea un usuario corriente al home del usuario. En las vistas de la carpeta Auth, vemos que hay una vista que realiza el login llamada “login.blade.php”. En esta vista, al finalizar la inserción de los datos, redirige a una función del LoginController llamada “login”. Para ello, en el

HomeController creamos una nueva función llamada "login" que sobrescribe la función con el nuevo código que vamos a escribir.

```
<div class="card-body">
  <form method="POST" action="{{ route('login') }}">
    @csrf

0 references | 0 overrides
public function login(Request $request)
{
    $input = $request->all();
    $this->validate($request,[
        'nick' => 'required',
        'password' => 'required'
    ]);

    if(auth()->attempt(array('nick'=>$input['nick'],'password'=>$input['password']))){
        if(auth()->user()->admin == true){
            return redirect()->route('admin.home');
        }else{
            return redirect('home');
        }
    }else{
        return redirect()->route('login');
    }
}
```

Después de realizar la validación, llamamos a la función attempt() de la clase auth que comprobará si el nick y el password introducido son los correctos. En la siguiente condición, comprobará si el campo admin del usuario registrado es igual a true o false. Dependiendo del resultado, redirigirá al home del administrador o al del usuario. Si no se cumple la primera condición, retornará al login inicial.

8. Pruebas:

- Logearse con un usuario normal:

Login

Nick

Gonzalo

Password

.....|


☐ Remember Me

Login

[Forgot Your Password?](#)

Dashboard
You are logged in!

- Loguearse como administrador:

Login	
Nick	Marc
Password 
<input type="checkbox"/> Remember Me	
<input type="button" value="Login"/> Forgot Your Password?	

Dashboard
You are logged in as an admin!