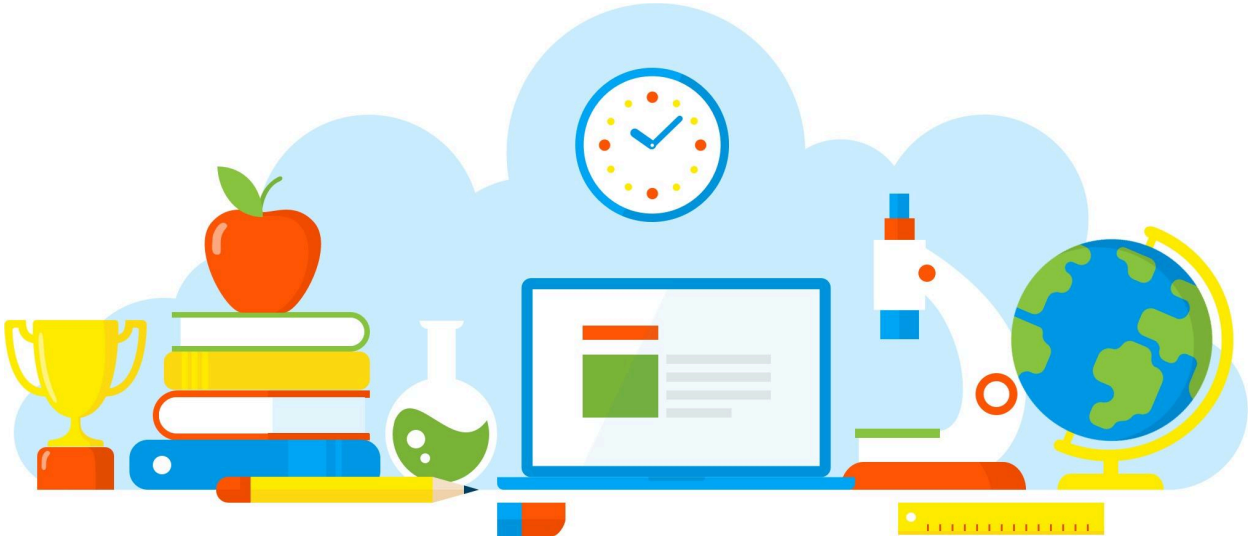

Implémentation de l'authentification

Documentation technique

Filâtre Killian
21/06/2024

Sommaire :

Introduction

L'authentification dans Symfony 6.4 permet de sécuriser l'accès aux parties de votre application. Cette documentation vous guide à travers les étapes nécessaires pour

mettre en place l'authentification en utilisant Symfony 6.4.

Pré-requis

- Symfony 6.4 installé
- Composer installé
- Une base de données configurée

Création de l'entité User

En premier lieu, avant de pouvoir authentifier un utilisateur, nous avons besoin de créer une entité User qui comprendra les informations de l'utilisateur.

```php
class User implements UserInterface, PasswordAuthenticatedUserInterface
{
    1 usage
    #[ORM\Id]
    #[ORM\GeneratedValue]
    #[ORM\Column]
    private ?int $id = null;

    2 usages
    #[ORM\Column(length: 255)]
    private ?string $email = null;

    2 usages
    #[ORM\Column(length: 255)]
    private ?string $username = null;

    2 usages
    #[ORM\Column(length: 255)]
    private ?string $password = null;

    2 usages
    #[ORM\Column(length: 255)]
    private array $roles = [];

    /** @var Collection<int, Task> ...*/
    5 usages
    #[ORM\OneToMany(targetEntity: Task::class, mappedBy: 'id_user', orphanRemov
    private Collection $tasks;
```

Cette entité User a quelques méthodes que Symfony doit utiliser pour la gestion de la sécurité, telles que getRoles(), getSalt(), et eraseCredentials().

SecurityBundle

SecurityBundle est le composant principal responsable de l'authentification et de l'autorisation dans Symfony. SecurityBundle utilise une combinaison d'encodeurs de mots de passe, de providers de l'utilisateur, de firewalls, et de règles d'accès pour gérer l'authentification et l'autorisation.

Configuration du fichier de sécurité

La config du SecurityBundle se fait dans le fichier config/packages/security.yaml :

Voici un exemple :

```yaml
security:
    # https://symfony.com/doc/current/security.html#registering-the-user-hashing-passwords
    password_hashers:
        Symfony\Component\Security\Core\User\PasswordAuthenticatedUserInterface: 'auto'
    # https://symfony.com/doc/current/security.html#loading-the-user-the-user-provider
    providers:
        app_user_provider:
            entity:
                class: App\Entity\User
                property: username
    firewalls:
        dev:
            pattern: ^/(_(profiler|wdt)|css|images|js)/
            security: false
        main:
            lazy: true
            provider: app_user_provider
            form_login:
                login_path: app_login
                check_path: app_login
                always_use_default_target_path: true
                default_target_path: /
                enable_csrf: true
            logout:
                path: app_logout
                target: app_login
```

Le provider de l'utilisateur est défini pour charger des utilisateurs à partir de l'entité User, en utilisant le username comme identifiant. Le firewall est configuré pour utiliser le provider app_user_provider, avec une protection par formulaire pour l'authentification.

Création du formulaire de connexion

Nous devons ensuite créer un formulaire de connexion avec un route /login via la commande : php bin/console make:auth

```php
class SecurityController extends AbstractController
{
    /**
     * Handles the login functionality.
     *
     * @param AuthenticationUtils $authenticationUtils The authentication utility.
     * @return Response The response object.
     */
    killiadmin
    #[Route(path: '/login', name: 'app_login')]
    public function login(AuthenticationUtils $authenticationUtils): Response
    {
        // Get the login error if there is one
        $error = $authenticationUtils->getLastAuthenticationError();

        // Last username entered by the user
        $lastUsername = $authenticationUtils->getLastUsername();

        return $this->render( view: 'security/login.html.twig', [
            'last_username' => $lastUsername,
            'error' => $error,
        ]);
    }
```

Création du système d'authentification

L'authentification est gérée par des authenticators (Guard Authenticators). Un authenticator est une simple classe PHP qui fait trois choses :

- Lire les informations d'authentification du message de requête (c'est-à-dire du formulaire de connexion).
- Valide ces informations
- Gère une réponse réussie ou échouée à la suite de l'authentification (succès ou échec).

Conclusion

Ce document fournit un aperçu de base de l'implémentation de l'authentification dans Symfony 6.4. Pour plus d'informations, veuillez consulter la documentation officielle de la sécurité Symfony - https://symfony.com/doc/6.4/security.html