

P8 : TODO LIST

Documentation technique Authentification Symfony 5

SYSTÈME D'AUTH. SYMFONY

Le composant de Sécurité de Symfony est un système qui permet de gérer plus ou moins facilement la sécurité des applications web. Par sécurité il faut comprendre la gestion de l'authentification mais aussi la gestion des accès aux ressources selon le profil de chaque utilisateur.

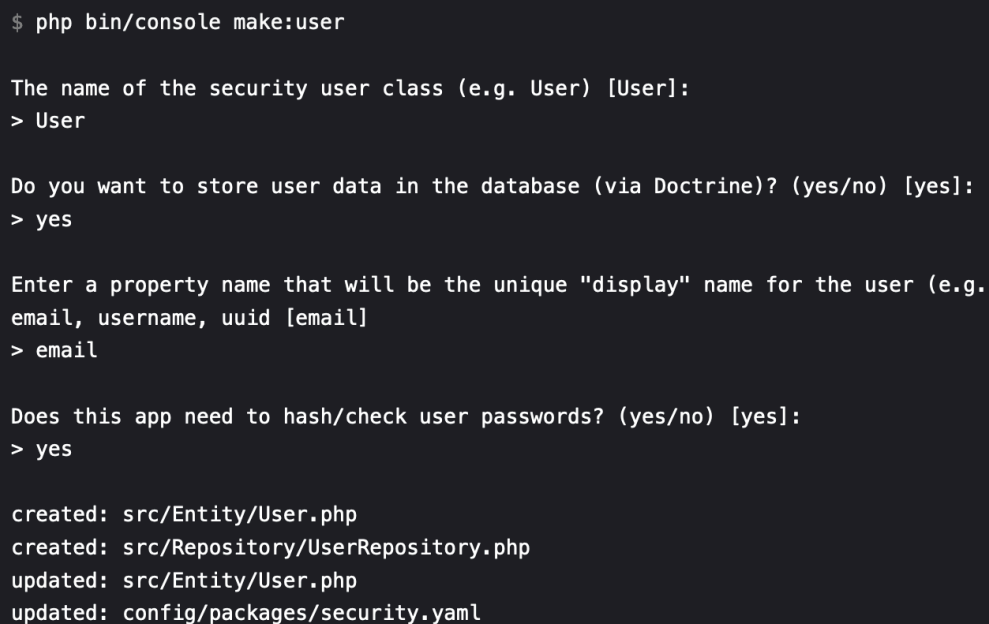
IMPLÉMENTATION DE L'AUTH.

L'application est accessible par tous. Mais il faut s'enregistrer et s'authentifier pour en bénéficier. Nous allons vous décrire le processus depuis la création de l'entité utilisateur jusqu'à la gestion se trouvant dans le fichier de configuration « security.yml

ENTITÉ USER

L'utilisateur est représenté par la classe User. Une contrainte d'unicité est appliquée sur les attributs "email" et "username" afin d'éviter un doublon. Cette classe User implémente la UserInterface. Elle hérite donc des fonctions de cette classe qui sont essentielles à la gestion des utilisateurs.

Grace au CLI de Symfony nous pouvons créer en ligne de commande l'entité User avec la génération de la config souhaitée.



```
$ php bin/console make:user

The name of the security user class (e.g. User) [User]:
> User

Do you want to store user data in the database (via Doctrine)? (yes/no) [yes]:
> yes

Enter a property name that will be the unique "display" name for the user (e.g.
email, username, uuid) [email]
> email

Does this app need to hash/check user passwords? (yes/no) [yes]:
> yes

created: src/Entity/User.php
created: src/Repository/UserRepository.php
updated: src/Entity/User.php
updated: config/packages/security.yaml
```

SECURITY.YML

C'est dans ce fichier que la sécurité est entièrement configurée. Il comprend :

- encoders** : encodage pour le mot de passe
- provider** : le comment, la classe et la propriété qui sera utilisée l'authentification.
- firewalls** : définis les différents authentifications possibles.
- access_control** : le gestionnaire des accès/routes.
- role_hierarchy** : mise en place d'une hiérarchie des rôles.

Pour plus d'informations se rendre dans la documentation de Symfony

[LINK] : <https://symfony.com/doc/current/security.html>

GESTION DE LA SÉCURITÉ

Grace à notre configuration « access_control » nous pouvons utiliser dans nos contrôleurs les annotations afin d'interdire ou non l'accès d'une route. Nous voyons que la route AdminController se retrouve avec un `@IsGranted()`. Cela permet d'identifier quelque rôle pourra afficher cette page.

```
1  // src/Controller/AdminController.php
2  // ...
3
4  + use Sensio\Bundle\FrameworkExtraBundle\Configuration\IsGranted;
5
6  + /**
7  +  * Require ROLE_ADMIN for *every* controller method in this class.
8  +  *
9  +  * @IsGranted("ROLE_ADMIN")
10 +  */
11  class AdminController extends AbstractController
12  {
13  +  /**
14  +  * Require ROLE_ADMIN for only this controller method.
15  +  *
16  +  * @IsGranted("ROLE_ADMIN")
17  +  */
18  public function adminDashboard()
19  {
20      // ...
21  }
22  }
```

A s'avoir que l'accès peut être fait aussi du côté du template twig

```
1 {% if is_granted('ROLE_ADMIN') %}  
2     <a href="...">Delete</a>  
3 {% endif %}
```

Pour plus d'informations se rendre dans la documentation de Symfony

[LINK] : <https://symfony.com/doc/current/security.html>