AUTHENTIFICATION

Le système de sécurité est configuré dans app/config/security.yml

Nous pouvons configurer Symfony pour qu'il authentifie nos utilisateurs en utilisant la méthode que nous voulons et pour charger les informations utilisateur à partir de n'importe quelle source. Quel que soit votre besoin, l'authentification est configurée dans security.yml, principalement sous la clé de pare-feu.

Encoder le mot de passe

Ici, nous pouvons choisir l'encodeur que nous voulons pour les mots de passe des utilisateurs, par example, bcrypt, sha12 ou si nous ne voulons rien, nous pouvons définir plaintext.

```
encoders:
    AppBundle\Entity\User: bcrypt
```

Ajouter de la hiérarchie des rôles

Dans la configuration ci-dessus, les utilisateurs qui ont le rôle ROLE_ADMIN aura également le rôle ROLE_USER.

```
role_hierarchy:
    ROLE_ADMIN: [ROLE_USER, ROLE_ADMIN]
    ROLE_USER: ROLE_USER
```

Les pages du site sont limitées à ROLE_USER

```
main:
    anonymous: ~
    pattern: ^/
```

Tout le site est sous ce pare-feu principal

C'est la grande partie de la configuration de sécurité. C'est ici que nous convertira l'accès aux URL. Nous pouvons configurer plusieurs pares-feux.
Nous créons une authentification de formulaire. Nous donnons un chemin pour la méthode de connexion accessible via login/, c'est la méthode loginAction dans le SecurityController.
Nous n'avons pas besoin d'implémenter un contrôleur pour l'Url login_check. Depuis le pare-feu, il va automatiquement intercepter et transformer tout formulaire soumis à cette Url.
Le always_use_default_target_path ignore l'Url précédemment demandée et redirige toujours vers la page par défaut défini sur default_target_path.
Habituellement, nous souhaiterons également que nos utilisateurs puissent se déconnecter. Heureusement, le pare-feu peut gérer cela automatiquement lorsque nous activons le paramètre de configuration de déconnexion.

```
firewalls:
    main:
        anonymous: ~
        pattern: ^/
        form_login:
            login_path: login
            check_path: login_check
            always_use_default_target_path:  true
            default_target_path:  /
        logout_on_user_change: true
        logout: true
```

Contrôle d'accès

Symfony nous offre plusieurs moyens d'appliquer l'autorisation, notamment la configuration access_control dans security.yml, l'annotation @Security et l'utilisation directe d'isGranted sur le service security.authorization_checker.

- Pour sécuriser un modèle d'url complet, nous utilisons access_control ;
- Dans la mesure du possible, utilisons l'annotation @Security;
- Vérifions la sécurité directement sur le service security.authorization_checker chaque fois que nous avons une situation plus complexe.

La section access_control est très puissante, mais elle peut aussi être dangereuse (car elle implique une sécurité) si nous ne comprenons pas comment cela fonctionne. En plus de l'URL, access_control peut correspondre à l'adresse IP, au nom d'hôte et aux méthodes HTTP. Il peut également être utilisé pour rediriger un utilisateur vers la version https d'un modèle d'URL.

Comment fonctionne la sécurité access_control?

Pour chaque requête entrante, Symfony vérifie chaque entrée access_control pour trouver celle qui correspond à la requête en cours. Dès qu'il trouve une entrée access_control correspondante, il s'arrête - seul le premier access_control correspondant est utilisé pour appliquer l'accès. Pour plus information : http://symfony.com/doc/3.4/security/access_control.html

Pour accéder à l'application, l'utilisateur devons être enregistré en tant que ROLE_USER. Donc, pour être un utilisateur, un visiteur doit avoir un compte.

Nous pouvons configurer ici :

IS_AUTHENTICATED_REMEMBERED : Tous les utilisateurs connectés ont ceci, même s'ils sont connectés à cause d'un cookie "Mémorisez-moi". Même si nous n'utilisons pas la fonctionnalité "Mémorisez-moi", nous pouvons l'utiliser pour vérifier si l'utilisateur est connecté.

Ceci est similaire à IS_AUTHENTICATED_REMEMBERED, mais plus fort. Les utilisateurs qui se sont connectés uniquement à cause d'un cookie "Mémorisez-moi" auront IS_AUTHENTICATED_REMEMBERED mais n'auront pas IS_AUTHENTICATED_FULLY

```
access_control:
    - { path: ^/, roles: ROLE_USER }
```

Seuls les utilisateurs avec ROLE_ADMIN peuvent accéder à ^ / users

```
access_control:
    - { path: ^/users, roles: ROLE_ADMIN }
```

Tout le monde peut accéder à la page ^ / login

IS_AUTHENTICATED_ANONYMOUSLY: Tous les utilisateurs (même anonymes) disposent de ceci - ceci est utile lors de la mise en liste blanche des URL pour garantir l'accès.

```
access_control:
    - { path: ^/login, roles: IS_AUTHENTICATED_ANONYMOUSLY }
```

Les utilisateurs sont chargés à partir de la base de données

C'est la partie dans laquelle nous configurons où nos utilisateurs sont chargés. Les utilisateurs sont stockés dans la base de données et chargés par la propriété du nom d'utilisateur.

```
providers:
    doctrine:
        entity:
            class: AppBundle:User
            property: username
```