Projet 8 : Le système d'authentification et d'autorisation

- Le fichier qui gère la sécurité de votre application s'appelle `security.yml` et le chemin pour y accéder et le configurer est le suivant : `app/config/security.yml`

- Les zones qui nécessitent une authentification sont cachés derrière un firewall

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

- Dans le code ci-dessous on voit que toutes les pages du site sont cachées derrière le firewall `main`. Ce dernier dirige tous les utilisateurs vers la page de connexion suivante : `/login`. L'authentification est gérée par la classe SecurityController qui a le chemin suivant `Appbundle\Controller\SecurityController.php`

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

- L'entrée `access control` détermine les niveaux d'authentification pour les pages, on y voit que les pages avec le chemins `/login` et `/users/create` autorise une authentification anonyme (accessible à tous) tandis que le reste des pages nécessitent d'avoir au minimum un rôle utilisateur (c-a-d avoir créer un compte utilisateur)

- Quand un utilisateur saisi ses données d'authentifications, il faudrait vérifier si ses informations correspondent à un utilisateur qui existent dans la base de données.
  Afin de récupérer les utilisateurs enregistrés dans la base de données, on utilise un `user provider`

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

  Ci-dessus on voit que notre provider c'est l'ORM doctrine et que la classe qui permet l'hydratation de l'objet c'est `AppBundle :User`, on voit aussi que le champ d'identification est la propriété `username`.

  L'algorithme d'encodage des mots de passe des utilisateurs est déterminé grâce à l'entrée `encoders` :

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

- Les rôles peuvent être hiérarchisées afin d'offrir des autorisations d'accès différentes :

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

Les autorisations peuvent être imposés dans les contrôleurs et dans les templates.

Finalement le fichier `security.yml` est connecté principalement au fichier `.env` qui contient l'url de la base de donnée ainsi que `AppBundle\Entity\User` qui permet de transformer les lignes récupérés de la base de données en objets