Authentification Guide

La sécurité concernant l'authentification est configuré dans le fichier `config/packages/security.yaml`
Vous trouverez plus d'informations concernant ce fichier et ses différentes parties dans la [documentation officielle de Symfony](https://symfony.com/doc/4.2/security.html).

L'entité User
Avant toute de chose, il est nécessaire d'avoir défini une entité qui representera l'utilisateur connecté.
Cette classe doit implémenter l'interface `UserInterface` et donc implémenter les différentes méthodes définis dans celle-ci.
Dans ce cas-ci, cette classe a déjà été implementée et se situe dans la fichier `src/Entity/User.php`.

Les Providers
Un provider va nous permettre d'indiquer où se situe les informations que l'on souhaite utiliser pour authentifier l'utilisateur, dans ce cas-ci, on indique qu'on recupérera les utilisateurs via Doctrine grâce à l'entité User dont la propriété username sera utilisé pour s'authentifier sur le site.
Attention, on peut indiquer ici la classe User car celle-ci implémente l'interface `UserInterface` !
```yaml
# config/packages/security.yaml
providers:
  # used to reload user from session & other features (e.g. switch_user)
  app_user_provider:
    entity:
      class: App\Entity\User
      property: username
```

Password hasher
Le composant PasswordHasher a été introduit dans la version 5.3. Avant cela version, la fonctionnalité de hachage de mot
de passe a été fournie par la sécurité composant.
Avant de hacher les mots de passe, vous devez configurer un hacheur à l'aide de la password_hashers option. Vous devez configurer l' algorithme de hachage et éventuellement quelques options d'algorithme
```yaml
# config/packages/security.yaml
when@test:
  security:
    password_hashers:

      Symfony\Component\Security\Core\User\PasswordAuthenticatedUserInterface:
        algorithm: auto
        cost: 4
        time_cost: 3
        memory_cost: 10
```

Les Firewalls
Un firewall va définir comment nos utilisateurs vont être authentifiés sur certaines parties du site.
Le firewall `dev` ne concerne que le développement ainsi que le profiler et ne devra à priori pas être modifié.
Le firewall `main` englobe l'entiéreté du site à partir de la racine défini via `pattern: ^/`, l'accès y est autorisé en anonyme c-à-d sans être authentifié, on y indique que c'est le provider "app_user_provider" qui sera utilisé.
Afin de s'authenfier, on définit un formulaire de connexion via `form_login:` où sont indiqués le nom des routes correspondant à ce formulaire, la route de vérification du login ainsi que la route vers laquelle l'utilisateur devra être redirigé par défaut après son authentification.
```yaml
# config/packages/security.yaml
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
    logout:
      path: app_logout
```

Les Access_Control
Un access_control va définir les limitations d'accès à certaines parties du site.
Dans ce cas-ci, on indique que :
- L'url /login est accessible sans authentification.
- L'url /admin n'est accessible qu'en étant authentifié avec un utilisateur ayant le rôle "ROLE_ADMIN".
```yaml
# config/packages/security.yaml
access_control:
        -  { path: ^/login, roles: IS_AUTHENTICATED_ANONYMOUSLY }
        -  { path: ^/admin, roles: ROLE_ADMIN }
```