Implémentation de l'authentification

User entity

L'utilisateur est représenté par la classe AppBundle Entity User. Une contrainte d'unicité est appliquée à l'attribut email afin de ne pas avoir de doublons.

```php
<?php

namespace App\Entity;

use App\Repository\UserRepository;
use Doctrine\Common\Collections\ArrayCollection;
use Doctrine\Common\Collections\Collection;
use Doctrine\ORM\Mapping as ORM;
use Symfony\Component\Security\Core\User\PasswordAuthenticatedUserInterface;
use Symfony\Component\Security\Core\User\UserInterface;

#[ORM\Entity(repositoryClass: UserRepository::class)]
#[ORM\UniqueConstraint(name: 'UNIQ_IDENTIFIER_EMAIL', fields: ['email'])]
class User implements UserInterface, PasswordAuthenticatedUserInterface
{
```

The security file

La sécurité concernant l'authentification est configurée dans le fichier config/packages/security.yaml. Pour plus d'informations https://symfony.com/doc/6.4/security.html

Providers

Un providers nous permettra d'indiquer où se trouve l'information que nous voulons utiliser pour authentifier l'utilisateur, nous disons que nous allons récupérer les utilisateurs par le biais de l'entité Utilisateur dont la propriété nom d'utilisateur sera utilisé pour authentifier sur le site.

```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:
    # used to reload user from session & other features (e.g. switch_user)
    app_user_provider:
      entity:
        class: App\Entity\User
        property: username
```

Firewall

Un pare-feu est conçu pour empêcher un utilisateur non authentifié d'accéder à certaines parties du site. Pour s'authentifier, un formulaire accessible à la route de connexion est utilisé :

```yaml
firewalls:
  dev:
    pattern: ^/(_(profiler|wdt)|css|images|js)/
    security: false
  main:
    lazy: true
    provider: app_user_provider
    form_login:
      # "app_login" is the name of the route created previously
      login_path: login
      check_path: login
      username_parameter: username
      password_parameter: password
```

Securing URL patterns (access_control)

La façon la plus simple de sécuriser une partie de votre application est de sécuriser un modèle d'URL entier dans security.yaml.

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

Hierarchical Roles

Au lieu de donner plusieurs rôles à chaque utilisateur, vous pouvez définir des règles d'héritage de rôle en créant une hiérarchie de rôle :

```yaml
role_hierarchy:
  ROLE_ADMIN: ROLE_USER
```