Document technique : sécurité

Grâce à Symfony, une grande partie des sécurités sont déjà mis en place, notamment sur les injections SQL dans les formulaires qui sont créés grâce à la commande : php bin/console make:form.
Mais il faut tout de même augmenter cette sécurité en effectuant quelques actions comme celle-ci.

Token CSRF

Premièrement, avec la création d'un formulaire de login et de sign up, symfony, de base, génère des token CSRF qui permettent d'authentifier l'utilisateur et ses permissions.

Par exemple au login :

```
<input type="hidden" name="_csrf_token"
       value="{{ csrf_token('authenticate') }}"
>
```

```
return new Passport(
    new UserBadge($username),
    new PasswordCredentials($request->request->get( key: 'password', default: '')),
    [
        new CsrfTokenBadge( csrfTokenId: 'authenticate', $request->request->get( key: '_csrf_token')),
        new RememberMeBadge(),
    ]
);
```

Il vérifie que le token est bien le bon.

Il le fait également lors de certaines actions, comme pour supprimer un événement.
Dans ce cas, lors de la suppression, on rajoute un input hidden avec comme valeur le token CSRF ('delete + id de l'event') :

```twig
<form action="{{ path('app_event_delete', {id: event.id}) }}" method="post">
    <input type="hidden" name="csrf" value="{{ csrf_token('delete' ~ event.id) }}">
    <input type="submit" value="Delete" class="cursor-pointer p-2 bg-custom-brown text-custom-beige rounded-md">
</form>
```

Et ensuite dans le controller, sur la route du delete, il vérifie que le token soit bon:

```php
#[Route('/event/delete/{id}', name: 'app_event_delete')]
public function delete(EntityManagerInterface $em, Event $event, Request $request): Response
{
    if ($this->isCsrfTokenValid( id: 'delete' . $event->getId(), $request->request->get( key: 'csrf'))){
        $em->remove($event);
        $em->flush();
    }

    return $this->redirectToRoute( route: '/');
}
```

Rôle admin

Lorsqu'un utilisateur est considéré comme admin, il peut avoir accès à certaines fonctionnalités comme le delete, create ou edit, que les utilisateurs normaux n'ont pas.
Cela ce fait de plusieurs manières, d'abord la protection d'une route :

```yaml
access_control:
    - { path: ^/event/create, roles: ROLE_ADMIN }
    # - { path: ^/admin, roles: ROLE_ADMIN }
    # - { path: ^/profile, roles: ROLE_USER }
```

En faisant cela, seuls les utilisateurs ayant le rôle admin peuvent accéder à la page de création d'événements.

On peut le faire également comme ceci :

```twig
{% if is_granted('ROLE_ADMIN') %}
    <div>
        <form action="{{ path('app_event_delete', {id: event.id}) }}" method="post">
            <input type="hidden" name="csrf" value="{{ csrf_token('delete' ~ event.id) }}">
            <input type="submit" value="Delete" class="cursor-pointer p-2 bg-custom-brown text-custom-beige rounded-md">
        </form>
        <a href="{{ path('app_event_edit', {id: event.id}) }}">Edit</a>
    </div>
{% endif %}
```

Avec ça, l’utilisateur ne verra les boutons/liens pour supprimer ou éditer ou créer un événement seulement s' il a le rôle admin.

THOMAS DORET-GAISSET
A3 ALT CDI