TP5 : Formulaires et Validation

Création de Formulaires

1. Ouvrez un terminal sous votre projet SF4, puis créer la classe formulaire correspondant à l'entité ***Article*** tapant la commande suivante :

   php bin/console make:form

Suivez l'assistant en fournissant le nom de la classe : ***ArticleType*** et le nom de l'entité : ***Article***

2. La classe form/ArticleType sera créée :

```php
<?php

namespace App\Form;

use App\Entity\Article;
use Symfony\Component\Form\AbstractType;
use Symfony\Component\Form\FormBuilderInterface;
use Symfony\Component\OptionsResolver\OptionsResolver;

class ArticleType extends AbstractType
{
    public function buildForm(FormBuilderInterface $builder, array $option
s)
    {
        $builder
            ->add('nom')
            ->add('prix')
        ;
    }

    public function configureOptions(OptionsResolver $resolver)
    {
        $resolver->setDefaults([
            'data_class' => Article::class,
        ]);
    }
}
```

3. Modifier la fonction new de la classe IndexController.php comme suit :

```
/**
     * @Route("/article/new", name="new_article")
     * Method({"GET", "POST"})
     */
    public function new(Request $request) {
        $article = new Article();
        $form = $this->createForm(ArticleType::class,$article);
        $form->handleRequest($request);
          if($form->isSubmitted() && $form->isValid()) {
          $article = $form->getData();
           $entityManager = $this->getDoctrine()->getManager();
          $entityManager->persist($article);
          $entityManager->flush();
            return $this->redirectToRoute('article_list');
        }
        return $this->render('articles/new.html.twig',['form' => $form-
>createView()]);
    }
```

4. Ajouter le *use* le classe ArticleType au début du fichier :

```
use App\Form\ArticleType;
```

5. Ajouter le Bouton « Créer » à la vue new.htm.twig :

```
{% extends 'base.html.twig' %}

{% block title %}Ajouter Article{% endblock %}
{% block body %}
 {{ form_start(form) }}

 {{ form_widget(form) }}
  <button type="submit" class="btn btn-success">Créer</button>
 {{ form_end(form) }}
{% endblock %}
```

6. Modifier le fichier inc/navbar.htm.twig pour ajouter la route new_article:

```
<a href="{{ path('new_article')}}" class="nav-link">Ajouter article</a>
```

7. Testez votre travail

8. Faire la même chose pour la fonction **edit** :

```
 /**
     * @Route("/article/edit/{id}", name="edit_article")
     * Method({"GET", "POST"})
     */
    public function edit(Request $request, $id) {
        $article = new Article();
$article = $this->getDoctrine()->getRepository(Article::class)->find($id);

        $form = $this->createForm(ArticleType::class,$article);

        $form->handleRequest($request);
        if($form->isSubmitted() && $form->isValid()) {

          $entityManager = $this->getDoctrine()->getManager();
          $entityManager->flush();

          return $this->redirectToRoute('article_list');
        }

        return $this->render('articles/edit.html.twig', ['form' =>
            $form->createView()]);
      }
```

9. Ajouter le Bouton « Modifier » à la vue edit.htm.twig :

```
{{ form_widget(form) }}
  <button type="submit" class="btn btn-success">Modifier</button>
{{ form_end(form) }}
```

10. Testez votre travail

Validation des données

11. Pour créer des contraintes de validation sur les données entrées dans les formulaires, symfony offre des annotations de validation qui seront créées au niveau de l'entité, pour plus de détails vous pouvez consulter la documentation symfony sur la page : https://symfony.com/doc/current/validation.html

12. On voudrait ajouter les contraintes :

    - Le nom d'un article doit comporter au moins 5 caractères et au maximum 50
    - Le prix d'un article ne doit pas être égal à 0

Pour ce faire modifier l'entité Article comme suit :

```php
<?php
namespace App\Entity;
use Doctrine\ORM\Mapping as ORM;
use Symfony\Component\Validator\Constraints as Assert;
/**
 * @ORM\Entity(repositoryClass="App\Repository\ArticleRepository")
 */
class Article
{    /**
     * @ORM\Id()
     * @ORM\GeneratedValue()
     * @ORM\Column(type="integer")
     */
    private $id;
    /**
     * @ORM\Column(type="string", length=255)
     * @Assert\Length(
     *      min = 5,
     *      max = 50,
     * minMessage = "Le nom d'un article doit comporter au moins {{ limit }} caractères",
     * maxMessage = "Le nom d'un article doit comporter au plus {{ limit }} caractères"
     * )
     */
    private $nom;

    /**
```

```
     * @ORM\Column(type="decimal", precision=10, scale=0)
     * @Assert\NotEqualTo(
     *     value = 0,
     *     message = "Le prix d'un article ne doit pas être égal à  0 "
     * )
     */
    private $prix;
...
```

13. Pour desactiver la validation HTML 5 coté client au niveau du navigateur modifier la fonction form_start au niveau du fichier new.html.twig comme suit :

```
{{ form_start(form, {'attr': {'novalidate': 'novalidate'}}) }}
```

14. Tester votre travail