Bucket-List : données et Doctrine

Fiche TP du module 5

Solution

- Configurer la base de données dans les fichiers `.env` et `.env.local`
- Créer la base de données avec :

  php bin/console doctrine:database:create
- Générer l'entité `Wish` avec make dans l'invite de commande, avec les propriétés demandées

```php
<?php

namespace App\Entity;

use App\Repository\WishRepository;
use Doctrine\ORM\Mapping as ORM;

/**
 * @ORM\Entity(repositoryClass=WishRepository::class)
 */
class Wish
{
    /**
     * @ORM\Id
     * @ORM\GeneratedValue
     * @ORM\Column(type="integer")
     */
    private $id;

    /**
     * @ORM\Column(type="string", length=250)
     */
    private $title;

    /**
     * @ORM\Column(type="text", nullable=true)
     */
    private $description;

    /**
     * @ORM\Column(type="string", length=50)
     */
    private $author;

    /**
     * @ORM\Column(type="boolean")
     */
    private $isPublished;

    /**
     * @ORM\Column(type="datetime")
     */
```

```
    private $dateCreated;

    // ... getters/setters ...
}
```

- Mettre à jour la base de données avec `doctrine:schema:update --force`
- Ajouter des données manuellement dans la table wish depuis PHPMyAdmin
- Dans `WishController`, dans la méthode `list` :
    - récupérer les idées avec la méthode `$repo->findBy()`
    - passer les idées à Twig avec le 2e argument de la fonction `render()`

```
public function list(WishRepository $wishRepository): Response
{
    // récupère les Wish publiés, du plus récent au plus ancien
    $wishes = $wishRepository->findBy(['isPublished' => true], ['dateCreated' => 'DESC']);

    return $this->render('wish/list.html.twig', [
        // les passe à Twig
        "wishes" => $wishes
    ]);
}
```

- Dans `list.html.twig` :
    - utiliser une boucle pour afficher les idées une par une
    - ajouter une balise `<a>` autours de chaque idée, menant à la page détails
    - pour générer la bonne URL dans le href, utiliser la fonction `path()` avec ses 2 arguments

```
{% block body %}
    <section class="wishes-list">
        <h2>All wishes!</h2>

        {% for wish in wishes %}
            <article>
                <a href="{{ path('wish_detail', {id: wish.id}) }}" title="View this wish in details">{{ wish.title }}</a>
            </article>
        {% endfor %}
    </section>
{% endblock %}
```

- Dans `WishController`, dans la méthode `detail` :
    - récupérer l'idée en fonction de son identifiant avec la méthode `$repo->find()`
    - passer l'idée à Twig

```
public function detail(int $id, WishRepository $wishRepository): Response
{
    // récupère ce wish en fonction de l'id présent dans l'URL
```

```
    $wish = $wishRepository->find($id);

    // s'il n'existe pas en bdd, on déclenche une erreur 404
    if (!$wish){
        throw $this->createNotFoundException('This wish do not exists! Sorry!');
    }

    return $this->render('wish/detail.html.twig', [
        "wish" => $wish
    ]);
}
```

- Dans `detail.html.twig` :
    - inutile de faire une boucle
    - afficher tous les détails de l'idée

```
{% block body %}
    <h2>{{ wish.title }}</h2>
    {# on utilise le filtre date() pour convertir l'objet en chaîne #}
    <p class="wish-credit">Created by {{ wish.author }} on {{ wish.dateCreated | date('Y-m-d') }}</p>
    {# le filtre nl2br convertie les sauts de ligne du texte en balises <br> #}
    <div>{{ wish.description | nl2br }}</div>
{% endblock %}
```

- Modification de style dans app.css :

```
main a {
    color: #8c6956;
}

.wishes-list article {
    margin-bottom: 0.5rem;
}

.wish-credit {
    font-size: 0.9rem;
    font-style: italic;
}
```