Création d'une seconde page

Nouvelle vue, nouveau contrôleur

De la même manière que pour l'affiche de notre première, nous allons construire une seconde page.

Imaginons que nous voulons construire une page profil.

Nous allons donc construire un nouveau contrôleur, avec une méthode permettant l'affichage de cette vue, ainsi qu'un nouveau template :

- Contrôler (/Controller/ProfilController.php)

```php
<?php

namespace App\Controller;

use Symfony\Bundle\FrameworkBundle\Controller\AbstractController;
use Symfony\Component\HttpFoundation\Response;
use Symfony\Component\Routing\Annotation\Route;
/**
 * @Route("/profil")
 * @method render(string $string, array $array)
 */
class ProfilController extends AbstractController
{
    /**
     * @Route("/detail", name="profil")
     */
    public function detail() :Response
    {
        // affichage de la page d'accueil
        return $this->render('profil/detail.html.twig');
    }
}
```

- Template (/Template/profil/detail.html.twig)

```twig
{% extends 'base.html.twig' %}
 {% block body %}
     <div class="container">
         <div class="row mt-5">
             <div class="col-sm-8 offset-sm-2">
                 <h1>Vos informations</h1>

             </div>
         </div>
     </div>
 {% endblock %}
```

La structure du template et du contrôleur est identique à ce qu'on a vu précédemment, les seuls changements majeurs sont dans le contrôleur :

- le nom du contrôleur et de la classe sont évidemment différents.
- nous avons ajouté une route au contrôleur pour le différencier du 'HomeController'.
- le nom et le chemin de la méthode sont également différents, afin de les repérer facilement.

Il nous reste plus qu'à mettre un lien qui nous permettrai de naviguer entre les 2 pages.

Pour cela nous allons dans un premier temps personnaliser la navbar :

- on ajoute un lien pour aller sur la page d'accueil

```
 <a class="navbar-brand" href="{{ path('home') }}">Accueil</a>
```

- on ajoute un lien pour aller sur la page 'profil'

```
<a class="nav-link" href="{{ path('profil') }}">Profil</a>
```

- on en profite pour enlever ce dont nous n'avons pas besoin, pour obtenir au final :

```
<nav class="navbar navbar-expand-lg navbar-light bg-light">
    <a class="navbar-brand" href="{{ path('home') }}">Accueil</a>
    <button class="navbar-toggler" type="button" data-toggle="collapse"
data-target="#navbarSupportedContent" aria-controls="navbarSupportedCon
tent" aria-expanded="false" aria-label="Toggle navigation">
        <span class="navbar-toggler-icon"></span>
    </button>

<div class="collapse navbar-collapse" id="navbarSupportedContent">
    <ul class="navbar-nav mr-auto">
        <li class="nav-item active">
            <a class="nav-link" href="{{ path('profil') }}">Profil</a>
        </li>
        <li class="nav-item">
            <a class="nav-link" href="#">Link</a>
        </li>
    </ul>
</div>
</nav>
```

- 'path' indique le nom de la route à cibler (dans les annotations, 'name="profil" ').

Maintenant que nous arrivons à naviguer entre 2 pages, il serait intéressant d'afficher des données qui viennent du contrôleur sur la vue.

Faire transiter des données du contrôleur à la vue

Passer un tableau

Dans 'ProfilController', dans la méthode 'detail', créons un tableau qui contiendra les informations à faire transiter :

```
$info = ['Loper', 'Dave', 'daveloper@code.dom', '01/01/1970'];
```

Pour afficher les informations contenues dans ce tableau, il suffit de le passer en second paramètre de la méthode 'render()', via un tableau d'options :

```
return $this->render('profil/detail.html.twig', [
         'informations' => $info
      ]);
```

Notre contrôleur aura donc cette structure :

```
// Controller/ProfilController.php

  /**
     * @Route("/detail", name="profil")
     */
    public function detail() :Response
    {
$info = ['Loper', 'Dave', 'daveloper@code.dom', '01/01/1970'];

        // affichage de la page d'accueil
        return $this->render('profil/detail.html.twig', [
            'informations' => $info
        ]);
    }
```

Affichons ces informations dans un tableau à l'aide d'une boucle for :

```
<!-- profil/detail.html.twig -->

 <ul>
    {% for info in informations %}
        <li>{{ info }}</li>
    {% endfor %}
</ul>
```

Passer un tableau associatif

Pour passer un tableau associatif du contrôleur vers la vue, le procédé reste le même, seul la méthode d'affichage va changer :

```
// Controller/ProfilController.php

    /**
     * @Route("/detail", name="profil")
     */
    public function detail() :Response
    {
$info = ['lastname' => 'Loper', 'firstname' => 'Dave', 'email' => 'davelope
r@code.dom', 'birthdate' => '01/01/1970'];

        // affichage de la page d'accueil
        return $this->render('profil/detail.html.twig', [
            'informations' => $info
        ]);
    }

<!-- profil/detail.html.twig -->
 <table class="table table-light table-hover">
     <thead>
```

```
    <tr>
        <th>Nom</th>
        <th>Prénom</th>
        <th>Date de naissance</th>
        <th>Email</th>
    </tr>
    </thead>
    <tbody>
    <tr>
        <td>{{ informations.lastname }}</td>
        <td>{{ informations.firstname }}</td>
        <td>{{ informations.birthdate }}</td>
        <td>{{ informations.email }}</td>
    </tr>
    </tbody>
</table>
```

On se sert ici du nom du tableau, avec la clé associative associée à la valeur que l'on veut afficher.