TP2 – Routes, contrôleurs et vues

2BACK - Développement backend Symfony

Objectifs du TP

Réinvestir les notions vues dans le cours : création d'un projet, routes, contrôleurs, réponses et premières vues Twig.

Comprendre la structure MVC appliquée à Symfony.

Préparer les prochaines séances sur Twig avancé, les formulaires et Doctrine.

Développer une première mini-application dynamique **de ton choix**.

Prérequis

Avoir Symfony CLI et Composer installés.

Avoir un environnement de développement fonctionnel (VS Code, PHP ≥8.1).

Connaître les bases de PHP orienté objet et HTML.

Étape 1 – Créer ton projet Symfony

1. Ouvre un terminal et place-toi dans ton dossier de travail.
2. Crée une nouvelle application
3. Entre dans ton dossier (`cd mon-projet`)
4. Lance le serveur local à l'aide de la commande appropriée.

💡 Vérifie que la page d'accueil Symfony s'affiche sur http://localhost:8000

Étape 2 – Découvrir la structure du projet

Avant de coder, explore l'arborescence :

- `config/` → fichiers de configuration
- `src/Controller/` → tes futurs contrôleurs
- `templates/` → tes futurs templates Twig
- `public/` → fichiers accessibles publiquement
- `var/` et `vendor/` → fichiers temporaires et dépendances

Note, sur papier, un coin les répertoires que tu utiliseras le plus aujourd'hui.

Étape 3 – Créer ta première route

Crée une nouvelle **classe contrôleur** `src/Controller/HomeController.php`.

Implémente une **méthode** `index()` associée à la **route racine** (`/`) et un **nom**.

Ce **contrôleur** doit afficher un simple **texte de bienvenue** écrit en **HTML** pur.

Lance ton **serveur** et rends-toi sur la **route racine** : le message de bienvenue s'affiche !

💡 Quelle **commande** te permet de **lister toutes les routes disponibles** ? Lance-la !

Étape 4 – Créer une nouvelle page

Crée une **nouvelle méthode** `about()` dans la classe contrôleur associée à la route `/about`, chargée d'afficher le template Twig `home/about.html.twig` :

Crée le dossier et le fichier Twig correspondant :

`templates/home/about.html.twig`

Ajoute un contenu simple :

- Un titre `h1`
- Un paragraphe `p` avec un message de bienvenue sur l'application

💡 Recharge le navigateur sur la route `/about` pour vérifier que la page s'affiche correctement.

Étape 5 – Ajouter une route dynamique

Dans la classe contrôleur, crée une nouvelle méthode `hello()` recevant une chaîne de caractères `$name` en paramètre, associée à une route `hello/{name}` (il s'agit d'un paramètre de route).

Envoyer cette valeur à un template Twig `home/hello.html.twig` en passant la première lettre en majuscule à l'aide de la fonction PHP `ucfirst()`.

Crée le template `home/hello.html.twig` affichant :

- Un titre H1 « `Bonjour {{ name }}` 👋 »
- Teste ton code sur les **routes** suivantes :
  `/hello/Chris`
  `/hello/Symfony`

💡 Observe le passage de paramètre entre la route et le template.

Étape 6 – Ajouter un contenu aléatoire

Crée une route `/random` qui affiche une **citation aléatoire** parmi au moins quatre stockées dans un tableau PHP :

```php
$quotes = [
    'Le code est poésie.',
    'Symfony simplifie la complexité.',
    'Toujours tester, jamais supposer.',
    'Refactoriser, c’est aimer son futur soi.'
];
```

Crée le template associé affichant la citation du jour.

💡 Recharge plusieurs fois : le contenu change !

Étape 7 – Navigation entre les pages

Ouvre le template Twig `templates/base.html.twig` (créé automatiquement par Symfony).

Ajoute un petit menu :

```
<nav>
    <a href="{{ path('app_home') }}">Accueil</a> |
    <a href="{{ path('app_about') }}">À propos</a> |
    <a href="{{ path('app_random') }}">Citation</a>
</nav>
<hr>
```

Fais en sorte que le menu soit affiché dans l'entête de chacune des pages.

💡 Prépare-toi : la prochaine séance portera sur la mise en forme et la factorisation des vues avec Twig !

Étape 8 – Défi bonus (pour les plus rapides)

Crée une route `/api/random` qui retourne la citation au format **JSON**.

Crée une route `/redirect` redirigeant automatiquement vers `/random`.

Crée une route `/error` qui lève une **erreur 404 personnalisée**.

Étape 9 – Questions de réflexion

Quelles sont les différences entre une **route YAML** et une **route** par **attribut PHP8** ?

Quelle **commande** permet de lister toutes les **routes** disponibles ?

Quelles **méthodes** permettent de **rediriger** un utilisateur vers une autre page ?

Quelle est la différence entre `Response` et `JsonResponse` ?

Des questions ?

chris.chevalier@ecole-hexagone.com