**Installation Steup**

**1)Créer un projet symfony:**

symfony new devwell --version="7.1.\*"

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**2)Installer les dépandances:**

**Installer Twig pour les templates :**

composer require symfony/twig-bundle

**Installer Doctrine (ORM) pour la gestion de la base de données :**

composer require symfony/orm-pack

**Installer Symfony Security Bundle pour la gestion des utilisateurs et de l’authentification :**

composer require symfony/security-bundle

**Installer Symfony Form et Validator pour la gestion des formulaires et des validations :**

composer require symfony/form symfony/validator

**Installer Symfony Asset Bundle pour la gestion des assets (CSS, JS, images) :**

composer require symfony/asset

**Installer PHPUnit pour les tests :**

composer require --dev phpunit/phpunit

**Installer le Profiler Pack pour le débogage :**

composer require symfony/profiler-pack

**Installer Symfony Maker Bundle pour la génération de code :**

composer require symfony/maker-bundle --dev

composer require symfony/maker-bundle --dev –with-all-dependencies

**2bis)Installer les dépandances:**

**Tailwind :**

composer require symfonycasts/tailwind-bundle

php bin/console tailwind:init

php bin/console tailwind:build --watch

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**3)Configuration de la base de données :**

Dans le fichier .env, configure ta connexion SQL (par exemple, pour MySQL) :

DATABASE\_URL="mysql://root:@127.0.0.1:3306/devwell\_db"

**Créer la base de données :**

php bin/console doctrine:database:create

### ****4) Configurer le routing et les contrôleurs de base :****

1)Créer un controller et sa créer la twig en automatique

php bin/console make:controller HomeController

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

### ****5) Test de fonctionnement :****

1. **Lancer le serveur Symfony :** Pour vérifier que tout fonctionne, lance le serveur Symfony en utilisant la commande suivante :
2. symfony server:start

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**6) Créer son repo git et le connecter**

Et le connecter à son repo github

git init

git add .

git commit -m "first commit"

git branch -M main

git remote add origin https://github.com/ilyaasAS/devwell.git

git push -u origin main

Mettez à jour régulièrement

git add .

git commit -m "Home"

git push

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Etape 1 : Créer l’inscription**

**Créez l'entité** User en utilisant la commande Symfony suivante :

php bin/console make:entity User

firstName (string)

lastName (string)

email (string)

password (string)

### 2 : Créer un formulaire pour l'inscription

1. **Utilisez la commande Symfony pour générer un formulaire** :

php bin/console make:form UserType

**3 ) Créer le contrôleur pour l'inscription**

php bin/console make:controller RegistrationController

et sa créer la twig

php bin/console doctrine:migrations:diff

php bin/console doctrine:migrations:migrate

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Etape 2 : Créer l’Authentification**

2. Créer l’authenticator

Générez un authenticator avec Symfony Maker pour gérer la connexion :

php bin/console make:auth

3. Créer le formulaire de connexion

templates/security/login.html.twig

LoginType.php

4. Créer le formulaire de connexion : configuer le security.yaml

https://main--zen-jepsen-04a701.netlify.app/posts/6.symfony6-authentification.article

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Etapes 3 : Faire ma NavBar**

Créer un dossier et le composant :

components/navbar.html.twig

base.html.twig :

<!-- navbar -->

        {% include 'components/navbar.html.twig' %}

#### **JavaScript**

Ajoutez ce script juste avant la fermeture du tag </body> pour activer/désactiver l'affichage du menu mobile lorsque vous appuyez sur le bouton burger.

<script>

    // Récupérer le bouton et le menu

    const menuToggle = document.getElementById('menu-toggle');

    const mobileMenu = document.getElementById('mobile-menu');

    // Ajouter un gestionnaire d'événement pour le bouton

    menuToggle.addEventListener('click', () => {

        // Basculer la classe `hidden` pour afficher/masquer le menu

        mobileMenu.classList.toggle('hidden');

    });

</script>

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Etapes 4: Faire mon Footer :**

Créer un dossier et le composant :

components/footer.html.twig

base.html.twig :

<!-- Footer -->

        {% include 'components/footer.html.twig' %}

faire les pages du footer :

php bin/console make:controller CGVController

php bin/console make:controller CGUController

php bin/console make:controller MentionLegaleController

php bin/console make:controller ContactControlle

et sa créer la twig

attention au nom des routes

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Etapes 5: Faire ma page profil:**

php bin/console make:controller ContactControlle

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Etapes 6: Faire ma page Admin :**

### ****1. Création des entités****

#### a. ****Créer l'entité Product****

Pour créer l'entité Product avec des propriétés telles que name, price, stock, description et une relation avec la catégorie (category), tu peux utiliser la commande suivante :

php bin/console make:entity Product

Lors de l'exécution de cette commande, Symfony te demandera d'ajouter les propriétés suivantes :

* **name** (type : string)
* **price** (type : float)
* **stock** (type : integer)
* **description** (type : text)
* **category** (relation ManyToOne avec l'entité Category)

Exemple pour la relation :

* **category** (ManyToOne, avec l'entité Category)

#### b. ****Créer l'entité Category****

Pour créer l'entité Category, tu peux utiliser cette commande :

php bin/console make:entity Category

Lorsque tu l'exécutes, tu ajouteras une propriété :

* **name** (type : string)

Ensuite, une relation **OneToMany** sera ajoutée dans l'entité Product pour référencer la catégorie.

2. **Génération du CRUD pour chaque entité**

a. **Générer le CRUD pour Product**

php bin/console make:crud Product

Cela générera :

* Le contrôleur ProductController
* Des vues Twig pour l'ajout, la modification, l'édition et la suppression des produits
* Le formulaire pour le produit

b. **Générer le CRUD pour Category**

php bin/console make:crud Category

Cela générera :

* Le contrôleur CategoryController
* Des vues Twig pour l'ajout, la modification, l'édition et la suppression des catégories
* Le formulaire pour la catégorie

c. **Générer le CRUD pour User**

php bin/console make:crud User

Cela générera :

* Le contrôleur UserController
* Des vues Twig pour l'ajout, la modification, l'édition et la suppression des utilisateurs
* Le formulaire pour l'utilisateur

### 3. ****Créer la page Admin pour gérer ces entités****

Maintenant que nous avons les CRUD pour chaque entité, tu peux créer une interface d'administration.

#### a. ****Créer le contrôleur AdminController****

Crée un contrôleur pour l'administration des produits, catégories et utilisateurs.

php bin/console make:controller AdminController

Cela créera un contrôleur AdminController et un fichier de vue par défaut.

#### b. ****Contrôleur AdminController****

Modifie ton contrôleur AdminController pour inclure les actions nécessaires pour gérer les entités **Product**, **Category**, et **User**.

#### c. ****Vues Twig****

Crée les vues Twig pour afficher l'interface d'administration dans templates/admin/.

* **index.html.twig** : Page principale de l'admin qui liste les produits, catégories et utilisateurs.
* **product/new.html.twig** : Formulaire d'ajout d'un produit.
* **category/new.html.twig** : Formulaire d'ajout d'une catégorie.
* **user/new.html.twig** : Formulaire d'ajout d'un utilisateur.