

# Création d'un mini blog avec Symfony

## Contexte :

Vous avez franchi le cap ! Le monde du développement web avec PHP Symfony s'offre à vous, et il est essentiel de vous démarquer pour séduire vos futurs employeurs ou clients en un clin d'œil. En tant que développeur PHP Symfony, il est temps de mettre en avant vos compétences en créant un blog à votre image.

L'objectif de cet examen est de développer un blog en utilisant la dernière version de Symfony, avec plusieurs types d'utilisateurs et des fonctionnalités spécifiques à chaque rôle. L'utilisation d'un Template Bootstrap est autorisée pour l'interface.

## Types d'utilisateurs et leurs droits :

### 1. Administrateur :

- Gestion des articles : L'administrateur peut ajouter, modifier et supprimer des articles.
- Gestion des utilisateurs : L'administrateur peut voir la liste des utilisateurs inscrits, valider ou désactiver un compte utilisateur.
- Gestion des commentaires : L'administrateur peut approuver ou désapprouver des commentaires (**facultatif**).

### 2. Utilisateur connecté :

- Accès aux pages publiques : L'utilisateur peut voir la page d'accueil et la liste des articles.
- Détails d'un article : L'utilisateur peut voir les détails de chaque article.
- Ajouter un commentaire : Si l'utilisateur est connecté, il peut ajouter un commentaire sur un article.
- Voir et modifier son profil : L'utilisateur peut consulter et modifier ses informations personnelles (**facultatif**).

### 3. Visiteur :

- Accès limité aux pages publiques : Le visiteur peut voir la page d'accueil, la liste des articles et les commentaires associés, mais ne peut pas ajouter de commentaires ni accéder aux fonctionnalités réservées aux utilisateurs connectés.

## Technologies :

- Symfony (dernière version)
- Bootstrap pour l'interface
- Utilisation de GitHub pour la gestion du code source

## Étapes de l'examen :

### 1. Mise en place du projet Symfony :

- Créer un projet Symfony avec la dernière version.

- Installer les dépendances nécessaires (ex. : Symfony Security pour la gestion des rôles).
- Mettre en place une base de données avec les entités suivantes : **User**, **Post**, **Comment**, et **Category**.
- Configurer le projet pour utiliser Bootstrap et un Template adapté.

## 2. Création des entités :

- Voici les propriétés des classes que vous pouvez utiliser pour la conception de votre blog :
  - **User (Utilisateur)**
    - ◆ id : integer (Identifiant unique)
    - ◆ email : string (Adresse e-mail, unique)
    - ◆ password : string (Mot de passe haché)
    - ◆ roles : array (Liste des rôles, ex: ROLE\_USER ou ROLE\_ADMIN)
    - ◆ firstName : string (Prénom de l'utilisateur)
    - ◆ lastName : string (Nom de l'utilisateur)
    - ◆ profilePicture : string (URL de la photo de profil, **facultatif**)
    - ◆ createdAt : datetime (Date de création du compte)
    - ◆ updatedAt : datetime (Date de dernière mise à jour, **facultatif**)
  - **Post (Article)**
    - ◆ id : integer (Identifiant unique)
    - ◆ title : string (Titre de l'article)
    - ◆ content : text (Contenu de l'article)
    - ◆ publishedAt : datetime (Date de publication)
    - ◆ picture : string (URL de l'image de l'article)
  - **Category (Catégorie)**
    - ◆ id : integer (Identifiant unique)
    - ◆ name : string (Nom de la catégorie)
    - ◆ description : text (Description de la catégorie, **facultatif**)
  - **Comment (Commenaire)**
    - ◆ id : integer (Identifiant unique)
    - ◆ content : text (Contenu du commentaire)
    - ◆ createdAt : datetime (Date de création du commentaire)
    - ◆ status : string (Statut : validé, en attente, supprimé) (**facultatif**)

## 3. Gestion des rôles :

- Implémenter deux rôles : ROLE\_ADMIN, ROLE\_USER.
- Configurer Symfony Security pour gérer l'authentification et les permissions en fonction des rôles des utilisateurs.

## 4. Explication des relations :

- User à Post : Un utilisateur (administrateur) peut avoir plusieurs posts (articles) mais chaque post appartient à un seul utilisateur (OneToMany de User vers Post).

- Post à Category : Chaque post appartient à une seule catégorie, mais une catégorie peut contenir plusieurs posts (ManyToOne de Post vers Category).
- User à Comment : Un utilisateur peut écrire plusieurs commentaires, mais chaque commentaire appartient à un seul utilisateur (OneToMany de User vers Comment).
- Post à Comment : Un post peut avoir plusieurs commentaires, mais chaque commentaire appartient à un seul post (OneToMany de Post vers Comment).

## 5. Mise en place de l'authentification :

- Implémenter un formulaire d'inscription et de connexion pour les utilisateurs.
- Gérer les routes d'inscription, connexion et déconnexion.
- Mettre en place un système de validation des comptes par l'administrateur si nécessaire (facultatif).

## 6. Utilisation de GitHub :

- Héberger le projet sur GitHub.
- Utiliser des commits fréquents pour suivre l'évolution du projet.

## 7. Tests et validation :

- Tester chaque fonctionnalité (ajout/modification/suppression d'articles, ajout de commentaires, gestion des profils utilisateurs).
- Vérifier les permissions selon les rôles (admin, utilisateur connecté, visiteur).
- Corriger les erreurs et améliorer la qualité du code.

## Critères d'évaluation :

- Fonctionnalité complète et correcte des différentes pages du blog.
- Mise en œuvre de la gestion des rôles et permissions.
- Qualité du code et respect des bonnes pratiques Symfony.
- Utilisation efficace de Git et gestion du code source.
- Design de l'interface avec Bootstrap, fonctionnalité responsive.