Landry Pierre, Siaud Mathieu, Boudier Theo, Guay Alexis

Projet symfony - EuroPlant

Lancer le project

Environnement recommandé

- PHP: version 8 minimum
- Symfony CLI : version 5 minimum
- Composer : version 2 minimum
- Un SGBD (mysql, postgresql, etc…)

Installation

```
git clone git@github.com:Linkale/ESGI_Symfony.git
cd ESGI_Symfony
composer install
nano .env.local
# écrire la ligne suivante avec vos propres données
DATABASE_URL="mysql://test:test@127.0.0.1:3306/europlant?serverVersion=5
.7&charset=utf8mb4"
enregistrer et quitter
php bin/console doctrine:database:create
php bin/console doctrine:migrations:migrate
php bin/console doctrine:fixtures:load
symfony server:start
```

Bundles utilisés

DoctrineFixturesBundle x Faker

Le bundle data-fixture va permettre de créer un ensemble de données qui permet d'avoir un environnement de développement proche d'un environnement de production avec des fausses données dans notre base de données.
Quant au bundle Faker, il complémente data-fixture, car il permet de générer aléatoirement des données.
Objectifs :

- Générer des données
- Pouvoir supprimer et ajouter des données en une ligne
- Gagner du temps

SecurityBundle

Le bundle Security permet à une application qui l'utilise de mettre en place les questions liées à la sécurité telles que l'authentification et l'autorisation dans le cadre Symfony.
Objectifs :

- Authentification de l'utilisateur
- Gérer les différentes autorisation
- Sécuriser une application

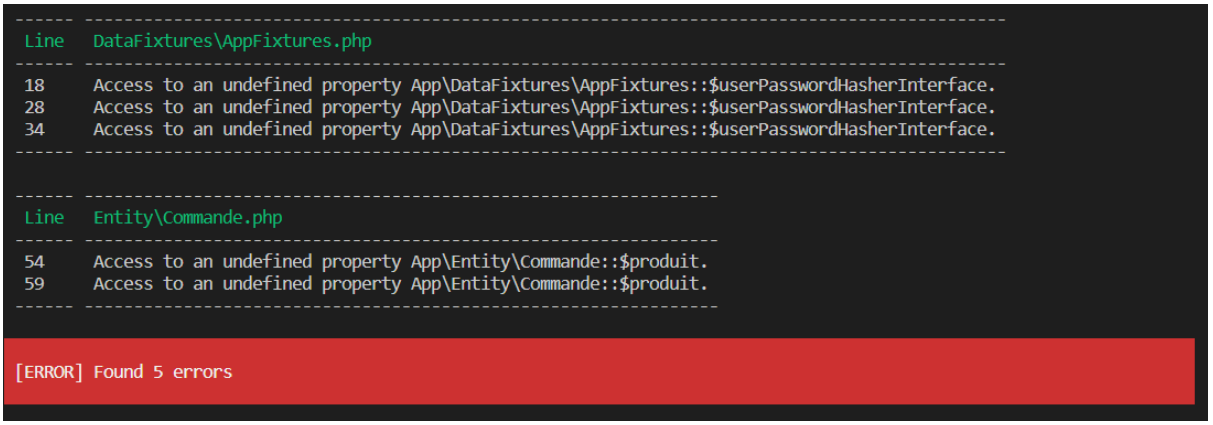
Qualité du code

Pour assurer une qualité de code optimale nous avons vérifié le code à l'aide de phpstan.

PHPStan

PHPStan analyse l'ensemble de votre base de code et recherche les erreurs éventuelles. PHPStan à la possibilité d'être exécuter sur une machine et dans CI pour éviter que ces erreurs n'atteignent les clients en production.

Voici le résulat après avoir lancé PHPStan sur l'application :



Les 5 erreurs ne sont pas des erreurs bloquantes car elles correspondent seulement à des recommandations sur le code.

Landry Pierre, Siaud Mathieu, Boudier Theo, Guay Alexis

Fonctionnalités

Système d'authentification

Les utilisateurs peuvent se connecter à leurs comptes et, suivant leurs droits, accéder à différentes fonctionnalités du site internet. On utilise le bundle SecurityBundle pour la gestion des connexions et des rôles.

Cas d'utilisation

Un client se connecte au site internet.
Un administrateur se connecte au site internet.

Améliorations

Authentification via Google

A l'aide du bundle knpUOAuth2ClientBundle ainsi que tu provider pour Google, on pourrait proposer aux utilisateurs de se connecter via Google pour leur simplifier la vie.

Impacts

Il faudrait au niveau du schéma de données ajouter un attribut google_user_id dans la table User qui permettrait d'associer un compte Google à un User dans notre système.

Gestion des commandes

Les commandes permettent de suivre les demandes des utilisateurs. Lorsqu'un client émet une commande pour un produit, celui-ci peut voir le récapitulatif de celle-ci. Les administrateurs peuvent accéder à toutes les commandes des différents clients, et pour chaque commande il peuvent leur attribuer une facture que le client pourra consulter.

Cas d'utilisation

Un client commande un produit.
Un client consulte ses commandes.
Depuis une commande, un client consulte les détails d'une commande.
Un administrateur consulte la liste des commandes.
Depuis une commande, un administrateur associe une facture à une commande.

Amélioration

Génération automatique des factures

Hormis le système de panier qui sera évoqué plus tard dans ce document, on pourrait proposer en guise d'amélioration, la génération automatique des factures. Actuellement les administrateurs doivent uploader sur notre site internet la facture liée à la commande. Il serait bénéfique de pouvoir générer automatiquement la facture sous format PDF.

Pour mettre en place cette amélioration, il faudra utiliser le bundle KnpSnappyBundle qui nous permettra de générer des fichiers PDF à partir de nos templates twig.

Landry Pierre, Siaud Mathieu, Boudier Theo, Guay Alexis

Gestion des produits

Un catalogue de produits est disponible sur le site internet. Les utilisateurs suivant leurs rôles peuvent effectuer différentes actions vis-à-vis de ces produits.

Cas d'utilisation

Un client regarde les produits proposés sur le site internet.
Un client clique sur un produit affiché dans le catalogue des produits pour voir les détails.
Depuis les détails d'un produit, un client passe une commande.
Un administrateur consulte les produits proposés sur le site internet.
Un administrateur ajoute un produit au catalogue.
Un administrateur sélectionne un produit dans le catalogue pour voir les détails du produit.
Depuis les détails d'un produit, un administrateur modifie un produit.
Depuis les détails d'un produit, un administrateur modifie un produit.

Améliorations

Ajout des photos des produits

Il faudrait proposer au administrateur un champ photo pour y déposer un fichier de type image. On pourrait ensuite afficher cette image pour donner plus envie à nos clients d'acheter nos produits.
Pour mettre en place cette amélioration, il faudra stocker les fichiers que l'utilisateur a envoyés dans le formulaire dans le dossier upload de notre système et ensuite mettre en base de données le path vers cette image dans un attributs image_path de notre table Produit.

Mis en place d'un système de panier et de paiement

On pourrait proposer au client un système de panier pour éviter de faire une commande par produits. Et rajouter un système de paiement pour valider le passage du panier en commande.

Avantages

Avec la mise en place d'un système de paiement à l'aide d'un panier, cela rassurerait les clients qui auront à leur disposition une expérience d'achat similaire à ce qui se fait sur la majeure partie des sites internet actuellement existants.

Inconvénients

Les clients vont maintenant devoir payer.

Impacts

Il faudra adapter notre schéma de base de données pour l'implémentation de cette mise a jours, cette modification entraînera la création et la modification d'une multitude de tables.
Pour la gestion des paiements, on pourrait utiliser l'API de Stripe qui permet de traiter les demandes de paiements avec un grand nombre de couches de sécurité pour éviter d'accepter le paiement d'une carte volée, ou d'une carte avec trop peu de fonds par exemple.