Semaine 19 : réaliser un projet Symfony

Objectif :

Cette semaine vous allez réaliser votre premier vrai projet Symfony et ceci en équipe. Cela vous permettra de mettre en œuvre sur une problématique métier les concepts vus jusqu'à lors mais aussi de comprendre en quoi un framework comme Symfony est un avantage notamment lorsqu'on travaille en équipe.

Compétences acquises :

- Comprendre le modèle requêtes/réponses
- Démarrer un projet Symfony via la CLI
- Gérer le routing de l'application
- Développer des controllers et associer des routes à des méthodes
- Utiliser le moteur de template twig pour l'affichage des vues
- Gérer sa base de données et ses entités avec l'orm Doctrine
- Créer des formulaires et gérer leur soumission
- Gérer ses utilisateurs et la sécurité de l'application
- Valider ses données
- Afficher des messages flash
- Peupler sa base à l'aide de fixtures
- Test son application à l'aide des tests unitaires

Ressources et exercices :

- https://openclassrooms.com/en/courses/4087056-testez-et-suivez-letat-de-votre-application-php/4419436-mise-en-place-dun-outil-pour-implementer-ses-tests-unitaires (très court, à compléter)

- https://symfony.com/doc/current/testing.html

- https://phpunit.readthedocs.io/en/9.3/ (documentation officielle, très technique)

- https://devcenter.heroku.com/articles/deploying-symfony4

Projet à rendre : Une application bancaire Symfony

Si l'évolution de votre application vers un modèle objet avec intégration du pattern MVC a permis de corriger les bugs remontés précédemment et d'améliorer la maintenabilité de l'application, trop soucis de maintenance sont encore présents.

En effet de nombreux développeurs se sont succédés sur le projet, apportant chacun leur style de code et aujourd'hui le code source a perdu en cohérence rendant son évolution difficile. De même des problèmes de performances lors des grosses montées en charge se font sentir.

C'est pour toutes ces raisons que votre chef de projet a décidé de migrer l'application vers sur le framework Symfony.

Rappel des spécifications fonctionnelles :

- L’application n’est accessible qu’aux seuls utilisateurs connectés
- Quand un utilisateur non connecté va sur l’application il est redirigé vers une page de connexion avec un formulaire
- Un utilisateur se connecte à l’aide d’une adresse mail et d’un mot de passe
- Un utilisateur connecté peut se déconnecter
- Une fois connecté, l’utilisateur voit uniquement ses comptes en banque personnels.
- Quand l’utilisateur clique sur un compte en banque, il arrive sur une page dédié au compte où il voit les informations du compte mais aussi les dernières opérations effectuées sur le compte
- Via une page dédiée un utilisateur peut créer un nouveau compte personnel à l’aide d’un formulaire. Une fois créé le compte apparaît sur la page d’accueil. Attention le compte doit respecter les conditions minimum de création de compte (bon type et bon montant)
- L’utilisateur peut effectuer des dépôts ou des retraits sur le compte de son choix. Le montant du compte est alors mis à jour et une nouvelle opération est enregistrée sur le compte.

En plus de ces spécifications, vous essaierez de :
- peupler la base de données à l’aide de fixtures
- valider les données rentrées dans les formulaires à l’aide du validator

Quelques conseils pour la gestion du projet :

- Assurez-vous d’avoir un UseCase ou une arborescence cohérente
- Choisissez un chef de projet
- Prenez le temps de préparer votre Kanban et de découper soigneusement les tâches en estimant leur temps et la date de réalisation
- Créer vous un planning de travail (éventuellement un diagramme de Gant pour assurer l’enchaînement des tâches)
- Réalisez au moins une réunion en début et fin de journée pour faire le point sur le travail fait et à faire
- Réalisez votre schéma de BDD, diagramme de classes et entités + migrations ensemble afin d’avoir une base commune
- N’échangez rien par mail ou clef, tout passe par GitHub
- N’oubliez pas de pull avant de push
- Vérifiez toujours si vous avez récupéré des migrations car il faut les exécuter
- Ne modifiez jamais la BDD manuellement, vous modifiez les entité que vous migrez
- Lancez régulièrement un composer update

Le rendu se fera via Teams avant dimanche soir minuit dans le dossier rendu vous déposerez un fichier txt au nom du chef de projet avec le lien vers votre repository GitHub qui contiendra le projet.

Pour aller plus loin :

- Ecrire quelques tests unitaires simples pour vérifier le fonctionnement de l’application
- Déployer votre application sur le service cloud Heroku afin d’en avoir une version en ligne

- L’utilisateur peut effectuer des transfères d’un compte à l’autre, ce qui crée deux nouvelles opération
- Des messages d’erreur sont éventuellement affichés sur les différents formulaires quand ceux-ci sont mal remplis
- L’utilisateur peut supprimer les comptes qui lui appartiennent
- Sur la page d’accueil, pour chaque compte l’utilisateur voit la dernière opération sur ce compte

- L’utilisateur peut accéder à la page de son profil et mettre à jour ses informations personnels