ToDo&Co

Audit de qualité et de performance

Auteur : MONTORO Stéphane

Date : 11/2022

Sommaire

Introduction

Contexte du projet

ToDo&Co est une startup dont le cœur de métier est une application permettant de gérer ses tâches quotidiennes. L'entreprise vient tout juste d'être montée, et l'application a dû être développée à toute vitesse pour permettre de montrer à de potentiels investisseurs que le concept est viable.

Objectifs du projet

ToDo&Co à récemment obtenu une levée de fond pour le développement de l'entreprise, elle a donc choisi d'engager un développeur afin de s'occuper de l'amélioration de l'application.

Plusieurs objectifs ont été demandé :

- Implémentation de nouvelles fonctionnalités.
- Identification et correction d'anomalies.
- Implémentation de tests automatisés.
- Établir une documentation et un audit de qualité et de performance.

Procédé

L'audit de performance et de qualité sera réalisé en plusieurs étapes.

1. Dans un 1er temps, il est nécessaire d'établir un « état des lieux » de l'application en pratiquant une analyse manuelle puis automatique de l'application d'un point de vue fonctionnelle puis technique afin de détecter les différents comportements présents dans l'application et incohérence de code existant.
2. Une fois cela fait, des correctifs et améliorations seront par la suite apportés à l'application afin de réduire la dette technique puis d'implémenter les nouvelles fonctionnalités demandées.
3. Lorsque les améliorations apportées, la dette technique réduite, et l'application entièrement fonctionnelle, un second audit sera réaliser afin d'établir le niveau d'amélioration de l'application apporté en termes de qualités et de performance.

État des lieux du projet

Cet état des lieux est une analyse préliminaire du projet qui a pour but de mesurer la dette technique et les différentes anomalies présente dans l'application avant d'effectuer de quelconque changement.

Analyse technique

Afin de réaliser une analyse technique fidèle à l'application, l'analyse technique a été réalisée dès la récupération du projet.

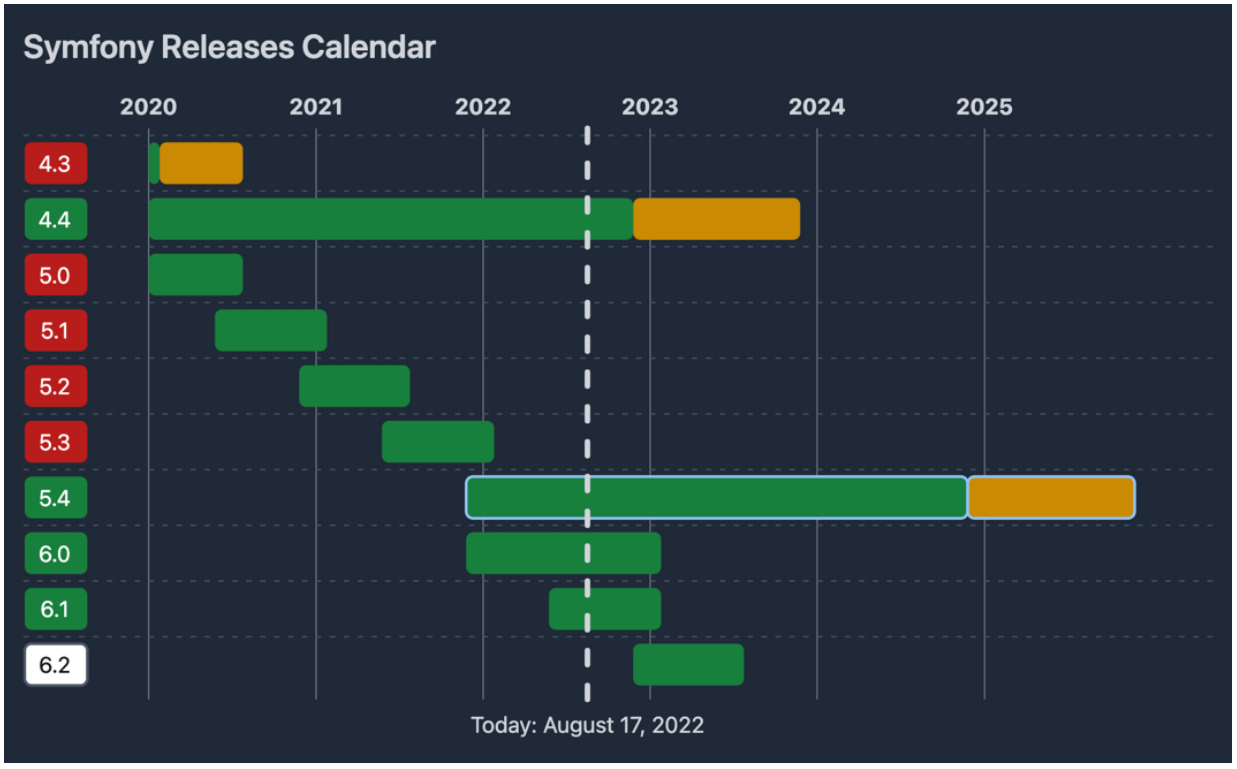
Cela a permis de démontrer un environnement technique obsolète avec une version 3.1.6 de Symfony qui ne fonctionnait pas et de très nombreux composant qui était totalement obsolète.

Il a fallu placer le projet dans une version Symfony 3.4 pour rendre le projet fonctionnel.

Cela m'a permis d'effectuer les divers analyse technique et fonctionnel.

A titre informatif, afin d'assurer une pérennité au développement et maintenance de l'application, il est recommandé d'utiliser une version stable du framework Symfony appelée LTS (Long Term Support).

Actuellement, la dernière version LTS est la version 5.4. (Voir photo ci-dessous)



Source : https://symfony.com/releases

L'essentiel des premiers correctifs apporté à l'application ont eu pour but de faire évoluer le projet initial vers une version LTS de Symfony, donc la version 5.4 afin de bénéficier des améliorations et fonctionnalités du framework.

L'application possédant beaucoup de dépendance et de composant obsolète, et après quelque mise à jour effectué de la version 3.4 à 4.0 puis 4.0 à 4.4, il a été déterminé qu'il était techniquement beaucoup plus profitable à l'application d'être recréée dans une version 5.4 de Symfony afin de permettre à l'application de faire peau neuve. Pour garder l'originalité du projet, tout le code métier et visuel de l'application a été conservé.

Cela a permis à l'application de ne plus garder de dépendances/composants obsolète.

Qualité du code

Analyse manuelle

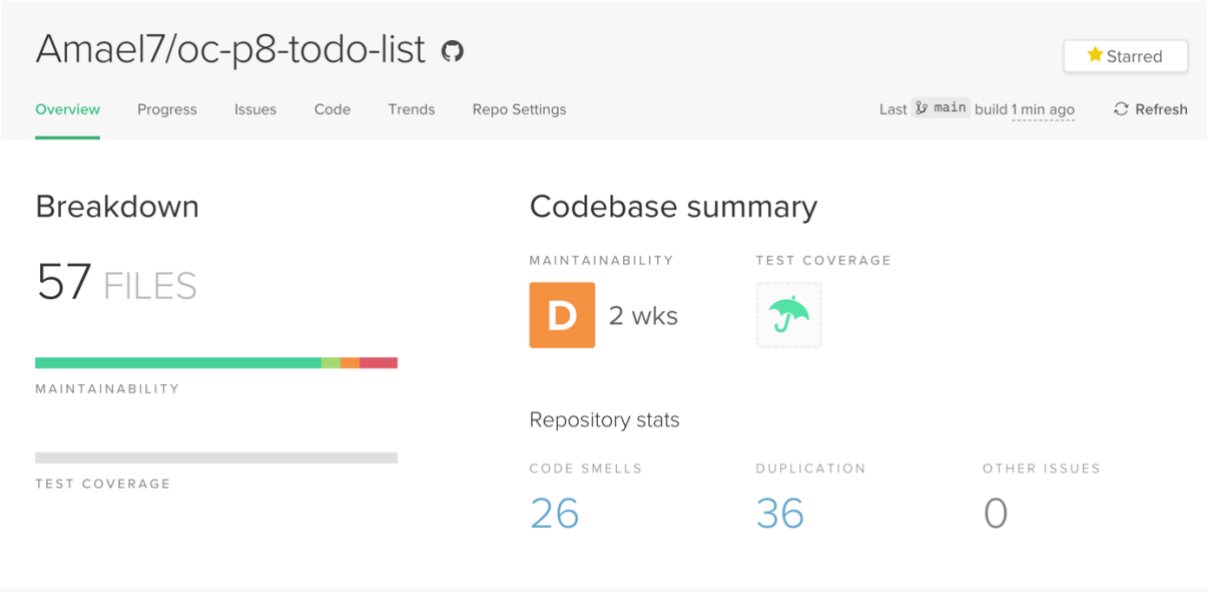
Une 1ère lecture du code a permis d'observer une architecture de type MVC **(Model-View-Controller)** en adéquation avec les bonnes pratiques du framework Symfony. On retrouve néanmoins une **architecture obsolète** due à une version ancienne de Symfony, avec notamment l'ensemble du code métier présent dans un sous-dossier AppBundle du dossier src. Dans les versions plus récentes, le code métier est situé directement dans le dossier src sous le **namespace App**.

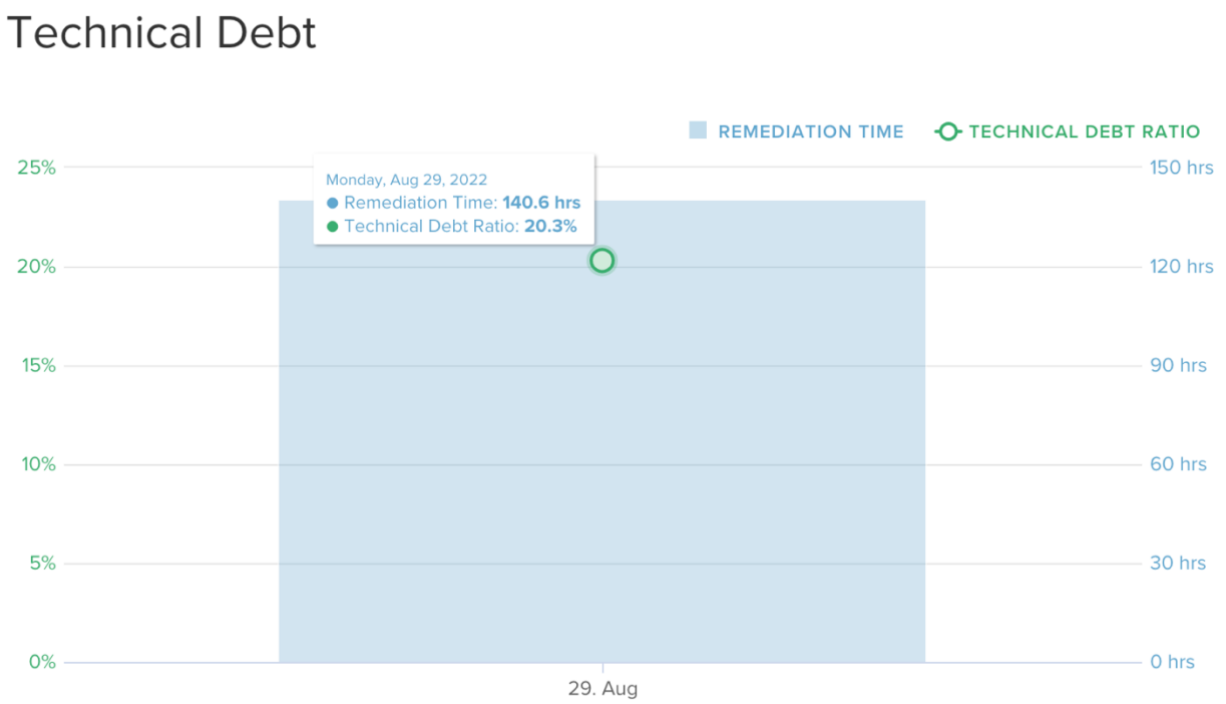
Analyse automatique

La qualité du code a été évaluée par l'outil **CodeClimate** qui permet de remonter différents problèmes pouvant être liés à la performance, les bonnes pratiques de développement, la sécurité, le style du code.

Cet outil d'analyse a été associé au repository GitHub sur lequel les modifications seront apportées afin d'effectuer un suivi à chaque mise à jour du code de l'application au cours du développement de celui-ci

L'analyse CodeClimate préliminaire est présentée ci-dessous :





L'analyse CodeClimate préliminaire démontre une note de **grade D** pour l'application à l'état initial, cela reflète un certain nombre d'anomalies présente dans l'application.

Des refactorisations permettant une diminution de la complexité de certains fichiers, ainsi que l'arrêt de l'utilisation de méthodes et variables non conseillées permettront entre autres de réduire significativement le nombre d'erreurs remontées.

Tests : couverture du code de l'application

Un rapport de couverture du code de l'application initiale a été généré par le biais de PHPUnit et a révélé une absence quasiment totale de tests (rapport de couverture ci-dessous).

	Code Coverage								
	Lines			Functions and Methods			Classes and Traits		
Total		0.00%	0 / 167		0.00%	0 / 34		11.11%	1 / 9
Controller		0.00%	0 / 88		0.00%	0 / 12		0.00%	0 / 4
Entity		0.00%	0 / 60		0.00%	0 / 20		0.00%	0 / 2
Form		0.00%	0 / 19		0.00%	0 / 2		0.00%	0 / 2
AppBundle.php		100.00%	0 / 0		100.00%	0 / 0		100.00%	1 / 1

Legend

Low: 0% to 50% **Medium:** 50% to 90% **High:** 90% to 100%

Generated by PHP_CodeCoverage 2.2.4 using PHP 7.4.30 and PHPUnit 4.8.36 at Sat Sep 10 14:31:51 UTC 2022.

Analyse fonctionnelle

L'analyse fonctionnel a été réalisée après la récupération du projet initial et une mise à jour vers une version 3.4 de Symfony, car la version d'origine de Symfony du projet était totalement obsolète et rendait l'application totalement inutilisable. La mise à jour était donc nécessaire afin de pouvoir pratiquer une analyse fonctionnelle de l'application.

Fonctionnalités

Au travers des différents schémas UML générés, de l'analyse fonctionnel de l'application, des anomalies ont pu ainsi être détecté.

Voici une liste des anomalies trouvé :

- Lien de l'en-tête vers la page d'accueil non fonctionnel.
- Lien « Consulter la liste des tâches terminées » non fonctionnel.
- Toutes les tâches (à faire et faites) apparaissent dans la page suivant le lien « Consulter la liste des tâches à faire ».
- Lorsque l'on valide une tâche comme étant « Marquer comme faites » ou « Marquer comme non terminée », le message de succès affiché et le même dans les deux cas « Superbe ! La tâche à été marquée comme faite ».
- Manque de validation concernant le mot de passe.

Qualité expérience utilisateur (UX/UI)

Cette analyse se focalise sur l'expérience utilisateur lors de l'utilisation de l'application et dénote de nombreux points pouvant être améliorer.

Liste des points améliorable :

- Visuel non attractif
- Pas de titres de pages
- Aucune page d'accueil pour les utilisateurs non authentifiés, une redirection vers la page de login serait plus cohérente.
- Mélange d'anglais et de français présent dans l'application
- Page d'erreur standards et non retravailler
- Pas de confirmation pour la suppression des tâches
- Navigation dans l'application limitée car peu de lien présent entre les pages (liste des utilisateurs, lien vers une page des tâches terminées, etc…)

Synthèse

Points d'amélioration identifiés et demandé par ToDo & Co et plan d'action correspondant :

- Choisir un rôle pourl'utilisateur :
  - Implémentation des rôles user et admin.
  - Possibilité de modifier le rôle d'un utilisateur lors de la modification de celui-ci par un administrateur.
- Une tâche doit être rattachée à un utilisateur :
  - Mise à jour de la structure de la bade de données afin d'intégrer une relation entre les tables Users et Tasks.
  - Assigné l'utilisateur authentifié comme étant l'auteur lors de la création de la tâche.
  - L'auteur de la tâche ne dois pas être modifiable.
  - Rattacher les tâches existantes a un utilisateur anonyme.
- Suppression des tâches par l'auteur seulement :
  - Seul l'auteur de la tâche est autorisé à la supprimé.
  - Seul un administrateur est autorisé à supprimé des taches provenant d'un auteur anonyme.
- Implémentation de tests :
  - Implémentation des tests unitaires avec PHPUnit.
  - Implémentation des tests fonctionnels avec PHPUnit.
  - Rapport de couverture de code supérieur à 70%.

Points d'amélioration identifiés lors de l'état des lieux dont la mise en place est nécessaire au bon fonctionnement de l'application :

- Version du framework Symfony Obsolète :
  - Migration vers une version LTS récente.
- Obsolescence des composants utilisés :
  - Mise à jour des dépendances.
- Code non documenté :
  - Documenter le code afin de faciliter la compréhension et la maintenance par d'autre développeurs.
- Implémentation de tests :
  - Implémentation de tests dans l'application afin d'assurer une bonne évolutivité de l'application dans le temps.

Points d'amélioration identifiés lors de l'état des lieux dont la mise en place n'est pas nécessaire au bon fonctionnement de l'application mais pourras faire partie d'une amélioration de l'application :

- Qualité Expérience utilisateur :
  - Amélioration de l'aspect visuel de l'application.
  - Amélioration du responsive.
  - Amélioration de la navigation.
  - Implémentation d'un système de traduction Anglais/Français.
  - Amélioration des pages d'erreurs.
- Sécurité :
  - Activation d'un compte utilisateur par email
  - Mise en place d'une demande de confirmation ainsi que d'une protection CSRF pour la suppression des tâches

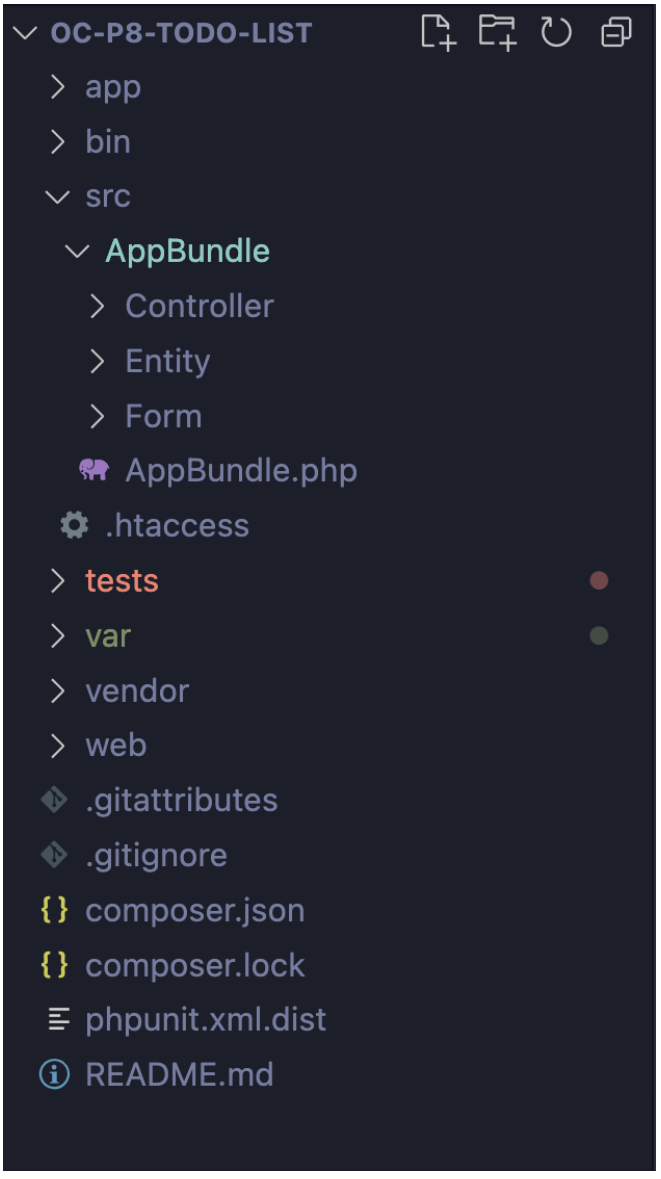
Rapport d'Audit de qualité et de performances

Suite à l'état des lieux de l'application, les mise à jour de la version du projet, de nouvelles fonctionnalités ainsi qu'un certain nombre d'actions correctives réalisé selon le plan d'action déterminé dans la partie précédente. Un audit de qualité et de performances a été réalisé et les résultats sont présentés ci-dessous.

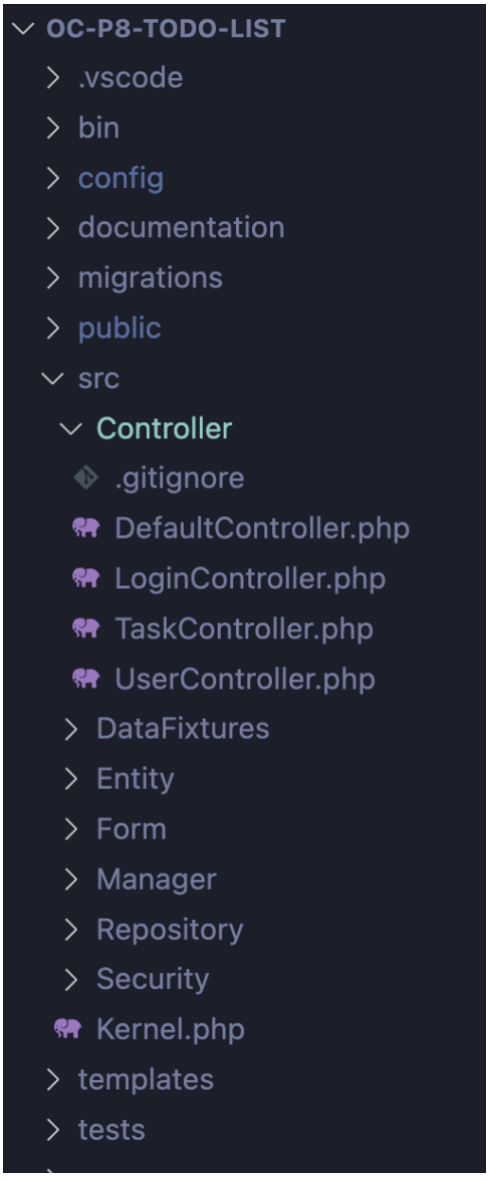
Qualité du code

Architecture

Avant :



Après :



Afin de respecter l'évolution de l'architecture des fichiers des versions récentes de Symfony, un remaniement conséquent a été réalisé.

Le code métier, initialement présent au sein du dossier AppBundle présent dans la route suivante : src/AppBundle est maintenant directement présent au sein de ce même répertoire qui correspond au namespace App\.

L'architecture MVC est conservée, mais la majorité de la logique initialement présente au sein des contrôleurs est maintenant répartie entre des managers (TaskManager et UserManager), dont le rôle est d'effectuer les actions sur les données et faire le lien avec les Repository, en charge de persister les données en base de données.

Les contrôleurs ne conservent donc que la responsabilité de traiter la requête et de renvoyer une réponse adaptée.

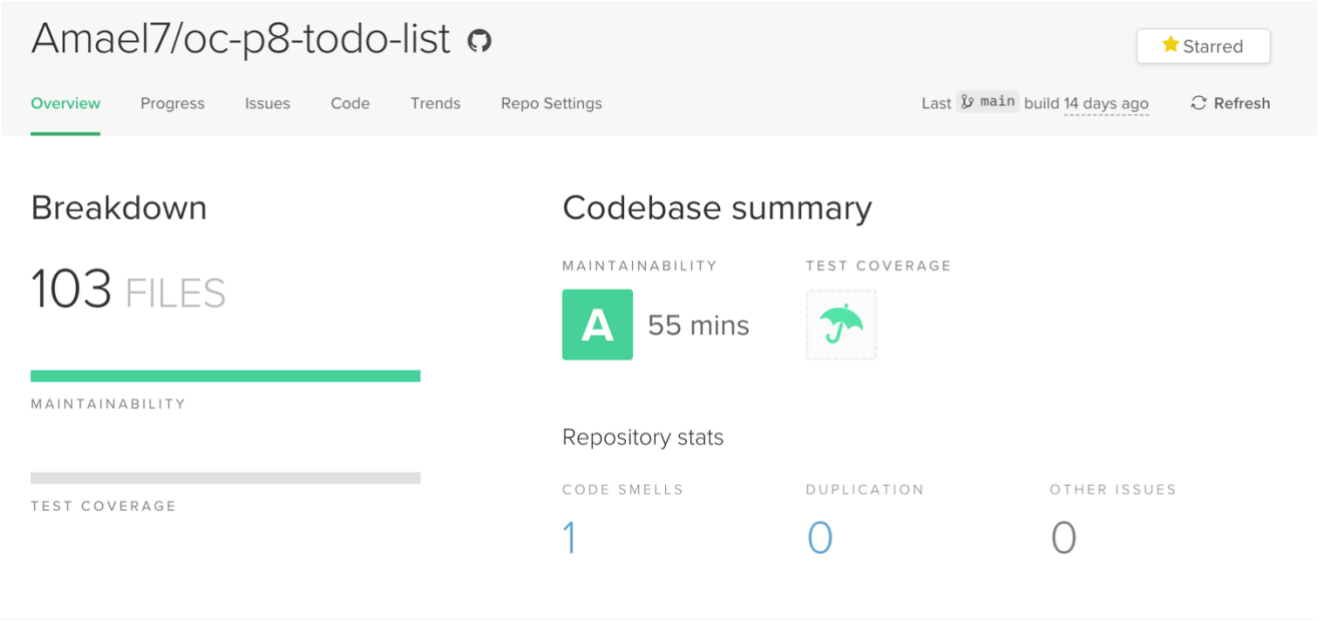
Le code ainsi remodelé est amené à être plus maintenable et évolutif dans le temps.

Documentation

Afin d'améliorer la compréhension du code existant par des personnes extérieures, toutes les méthodes du code métier ont été documentées.

CodeClimate

Alors que l'analyse préliminaire de CodeClimate indiquait un grade D, les remaniements du code ont permis une réduction de la dette, avec l'obtention d'un badge grade A, et la correction d'un certain nombre d'anomalies



Tests: couverture du code de l'application

L'analyse préliminaire ayant mis en évidence une absence quasiment totale de tests pour l'application, Des tests unitaires, fonctionnels et d'intégration pour les fonctionnalités ont été implémenté. Une fois les corrections des anomalies et l'application entièrement testée et fonctionnelle.

Voici le rapport de couverture de test :

/Users/stephanemontoro/code/Amael7/OpenClassrooms/oc-p8-todo-list/src / (Dashboard)

	Code Coverage								
	Lines			Functions and Methods			Classes and Traits		
Total		81.68%	156 / 191		92.42%	61 / 66		80.00%	12 / 15
Controller		100.00%	52 / 52		100.00%	13 / 13		100.00%	4 / 4
DataFixtures		0.00%	0 / 33		0.00%	0 / 4		0.00%	0 / 2
Entity		100.00%	47 / 47		100.00%	28 / 28		100.00%	2 / 2
Form		100.00%	10 / 10		100.00%	3 / 3		100.00%	2 / 2
Manager		100.00%	23 / 23		100.00%	8 / 8		100.00%	2 / 2
Repository		100.00%	14 / 14		100.00%	6 / 6		100.00%	2 / 2
Security		83.33%	10 / 12		75.00%	3 / 4		0.00%	0 / 1
Kernel.php		n/a	0 / 0		n/a	0 / 0		n/a	0 / 0

Legend

Low: 0% to 50% Medium: 50% to 90% High: 90% to 100%

Un taux de couverture de 70% était demandé et une couverture supérieure à 80% a était implémentée afin de couvrir la majorité du code métier de l'application et d'assurer une certaine stabilité et maintenabilité dans les développements futurs.

Performance de l'application

Les performances de l'application ont été réalisé avec l'outil d'analyse **Blackfire**.

Une analyse initiale de l'application a été réalisée juste après la mise en place d'une première version fonctionnel de l'application en version 3.4 de Symfony puis une seconde analyse fut réalisée à la suite des améliorations de l'application afin d'offrir une comparaison Avant/Après des performance de l'application.

Les comparatifs ont été réalisé sur diverse routes de l'application en GET et une route en POST,

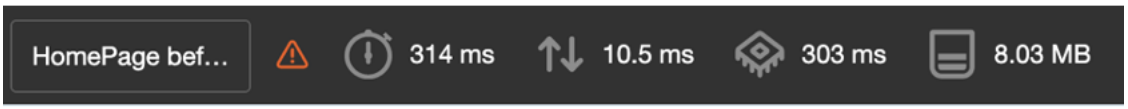
A savoir :

- La page d'accueil (/)
- La page liste des utilisateur (/users)
- La page des tâches (/tasks pour la version initiale et /tasks/todo pour la version finale)
- Action login en POST

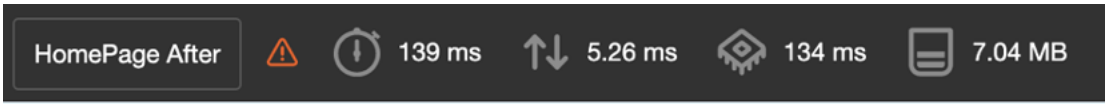
Les figure ci-dessous illustre les comparatifs des analyses Blackfire réalisées avant et après les améliorations, en termes de performance grâce à deux mesures : le temps de chargement de la page et la mémoire allouée. Comme on peut le constater, le temps de chargement des différentes pages a été au minimum divisé par 2 grâce aux optimisations, passant d'un temps de chargement d'environ supérieur à 300ms à un temps de chargement inférieur d'environs 150/170ms, confirmant une meilleure réactivité de l'application à l'issue des améliorations.

HomePage (/) :

Avant :

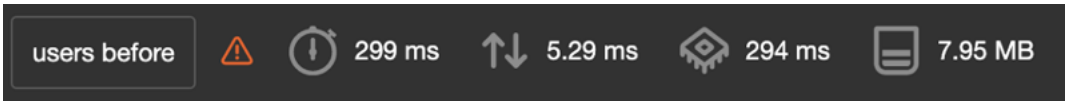


Aprés :

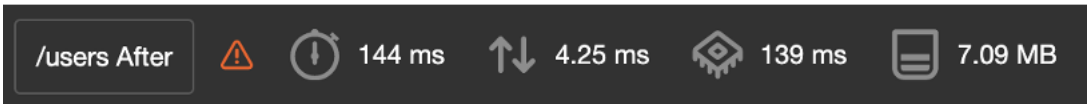


Liste des utilisateurs (/users) :

Avant :

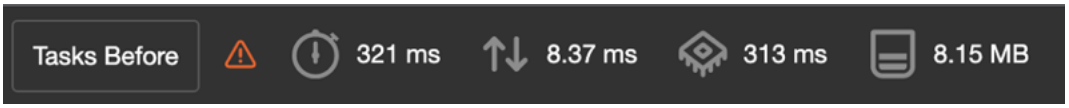


Après :

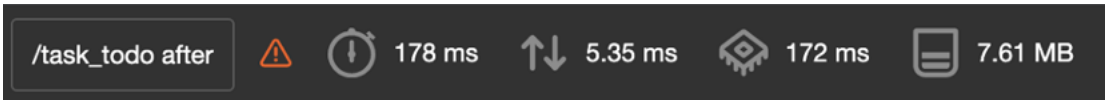


Liste des tâches (/task puis /task_todo) :

Avant :

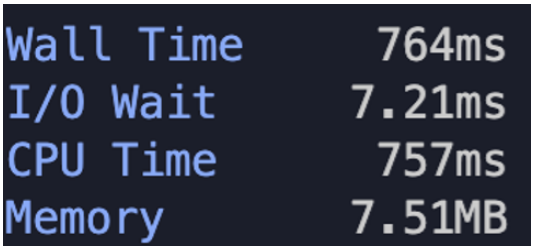


Après :

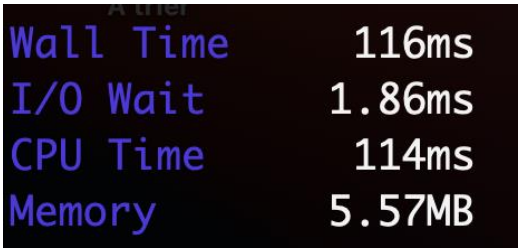


Login POST :

Avant :



Après :



Bilan et plan d'amélioration

Au commencement du projet d'amélioration de l'application ToDo&Co, l'état des lieux de l'application a démontré une dette technique ainsi qu'un certain nombre d'anomalies, d'une part spécifiée par le client et d'autre part révélé par une analyse technique et fonctionnelle.

Après toutes l'implémentation de correctifs et de nouvelles fonctionnalités demandées par le client et un audit de qualité et de performance réalisé, cela à démontré une amélioration significative de la qualité du code au travers de CodeClimate, du taux de couverture de code par des tests unitaires et fonctionnels avec PHPUnit ainsi que de la performance de l'application au travers de Blackfire

La qualité de l'application peut cependant toujours être amélioré au travers de divers pistes d'améliorations entrevue tout au long du projet.

Améliorations globales :

- Améliorer l'esthétique de l'application et le responsive design
- Traduction : mettre en place un système de traduction pour l'ensemble des pages.

Fonctionnalités :

- Améliorer la sécurité de l'application en ajoutant une vérification de l'email lors de l'inscription et en laissant la possibilité à l'utilisateur de modifier son mot de passe même s'il n'est pas administrateur.
- Améliorer la gestion des tâches : Pagination, filtres, Délai de réalisation de la tâche, Indice de progression de l'accomplissement de la tâche, possibilité de commenter/partager les tâches.
- Possibilité de supprimer un utilisateur, dans la limite d'avoir toujours au moins un compte administrateur actif.
- Améliorer la sécurité de l'application en ajoutant une confirmation de suppression de

tâche, ainsi qu'une protection CSRF.

Performance :

- Tests de performance régulier grâce à l'outil Blackfire, afin de vérifier les performances dans le temps et s'assurer de l'absence ralentissement ou de bug de performance au cours des développements.