ToDo & Co

Audit de performance et de qualité du code

Auteur : Luigi GANDEMER

Table des matières

1. Contexte

1.1 Présentation

L'application **Todolist** appartient à la société ToDo & Co. L'application a dû être développée à toute vitesse pour permettre de montrer à de potentiels investisseurs le **MVP (Minimum Viable Product)**.

1.2 Mise à jour des versions

L'application était initialement sur une version de Symfony qui n'est plus maintenu et un environnement assez anciens.

Technologies initiales :
- Symfony 3.1
- PHP 5.6.40
- Composer 2.1
- MySQL 5.6.51
- phpMyAdmin 4.9.11

Technologies après mis à jour :
- Symfony 6.4
- PHP 8.2.0
- Composer 2.5.7
- MySQL 8.0.31
- phpMyAdmin 5.2.0

1.3 Informations supplémentaires

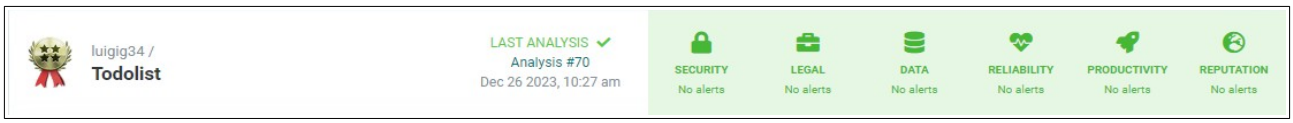
Toutes les informations fournis dans l'analyse de performance sont basés sur le **Profiler** de Symfony et l'outil **Lighthouse** de Google.

Des captures d'écran du **Profiler** et de **Lighthouse** seront aussi fournis dans un dossier « SCREENSHOTS » dans le repository GitHub.

2. Qualité du code

L'analyse de la qualité du code à été faites via l'outil **SymfonyInsight** lié au repository GitHub.

Le projet à actuellement un score parfait concernant la qualité du code (médaille de platine).

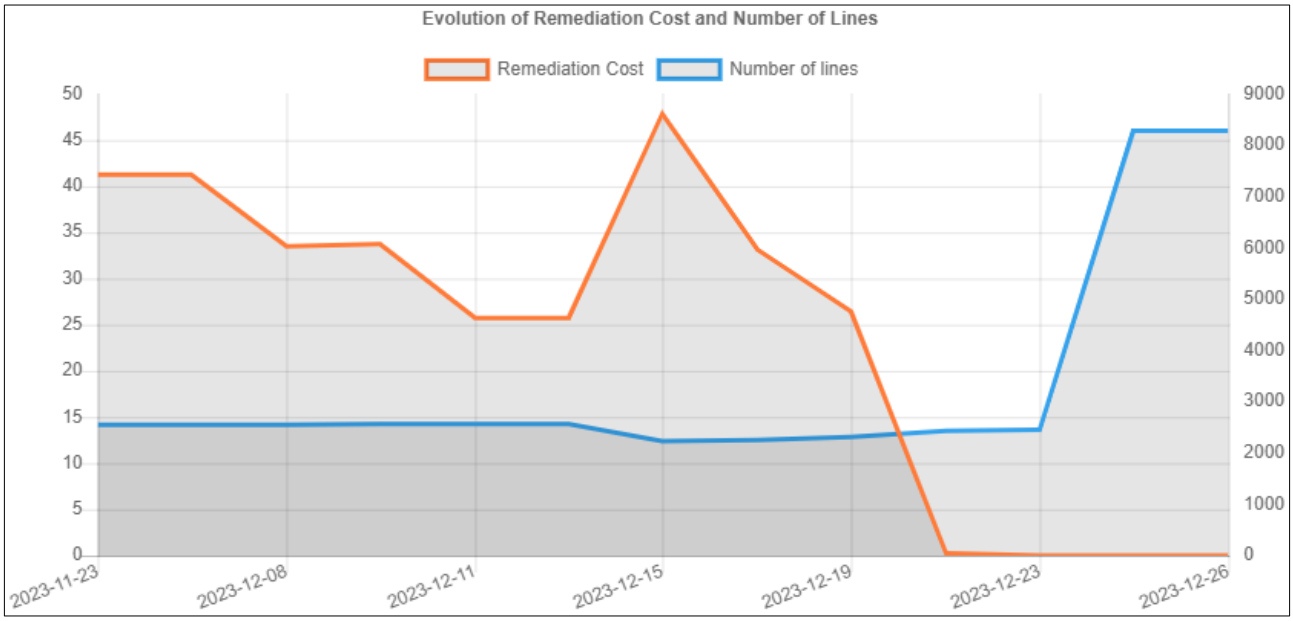


Le graphique ci-dessous montre l 'évolution du coût estimé pour corriger les problèmes dans le code (coût de remédiation) et le nombre total de lignes de code dans le projet.

Quand la ligne orange monte, cela signifie que le coût pour améliorer le code augmente.

La ligne bleu qui monte indique le nombre de lignes de code dans le projet.

Ce graphique sert à comprendre et améliorer la qualité du code au fil du temps.



On peut conclure que le coût actuel pour améliorer le code du projet est estimé à 0 car la qualité actuel du code est parfaite.

3. Performances

3.1 Solution mis en place

Les solutions mis en place pour améliorer la performance de l'application sont les suivantes :

- **Migrer sur Symfony 6.4 :** Cette version de Symfony est une LTS (Long Term Support) donc elle est toujours maintenu par les développeurs de Symfony.

- **Migrer sur PHP 8.2.0 :** Cette version de PHP est la plus récente actuellement, elle offre de meilleures performances, une sécurité renforcée, et introduit de nouvelles fonctionnalités.

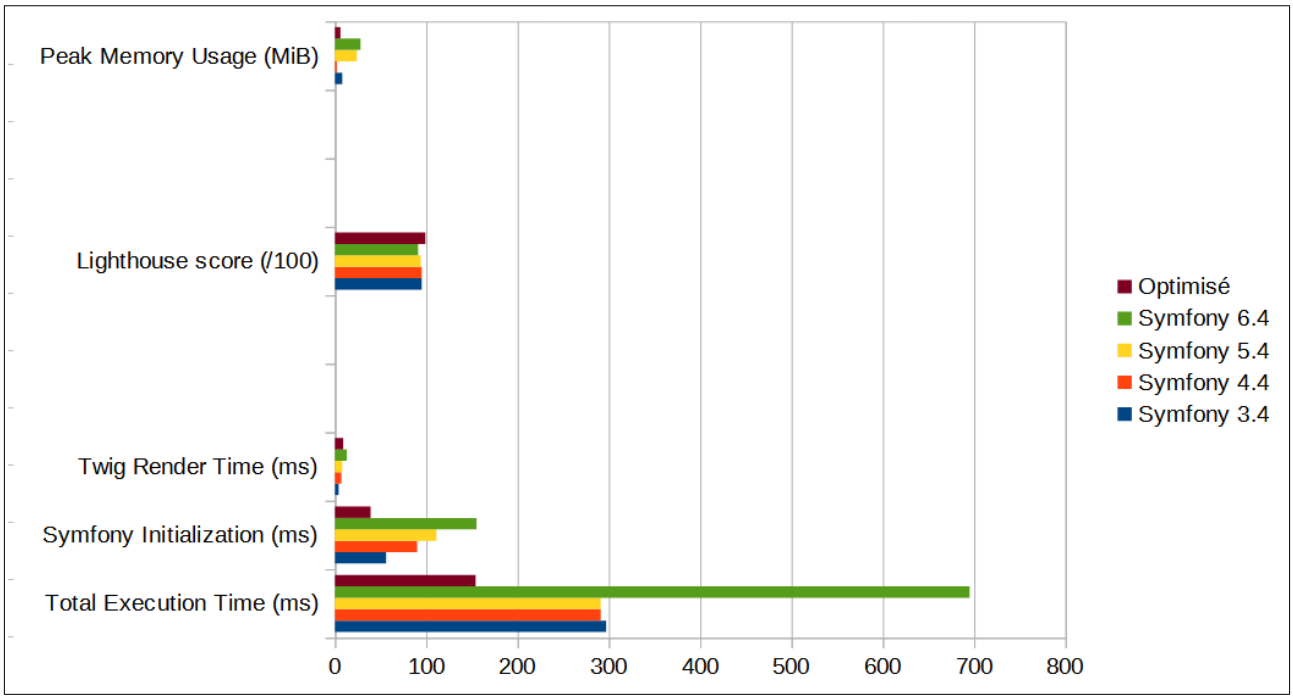
- **La mise en place de Doctrine Cache :** L'utilisation de « Doctrine Cache » réduit les temps de réponse stockant les résultat des requêtes fréquemment utilisées.

- **La mise en place de Doctrine Second Level Cache :** Le « Doctrine Second Level Cache » permet de conserver les données d'entité entre les requêtes. Ce qui diminue le besoin d'accéder à la base de données.

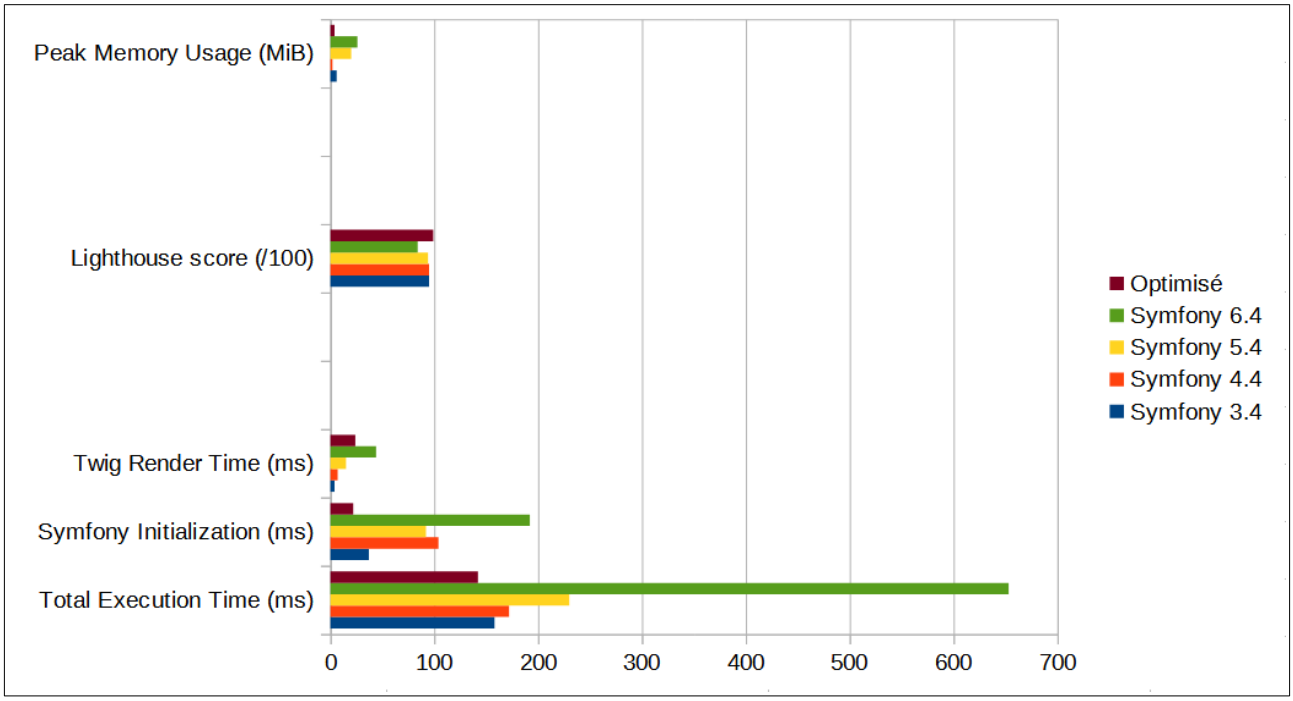
- **Activer l'extension OPCache dans php.ini :** L'activation d'OPCache compilent et stockent les scripts PHP en bytecode précompilé. Le serveur à donc pas besoin de recompiler le script.

- **La mise en place** du **Cache HTTP :** Le « Cache HTTP » réduit la charge sur le serveur et améliore les temps de chargement des pages en sauvegardant les versions statiques.

3.2 Page d’accueil

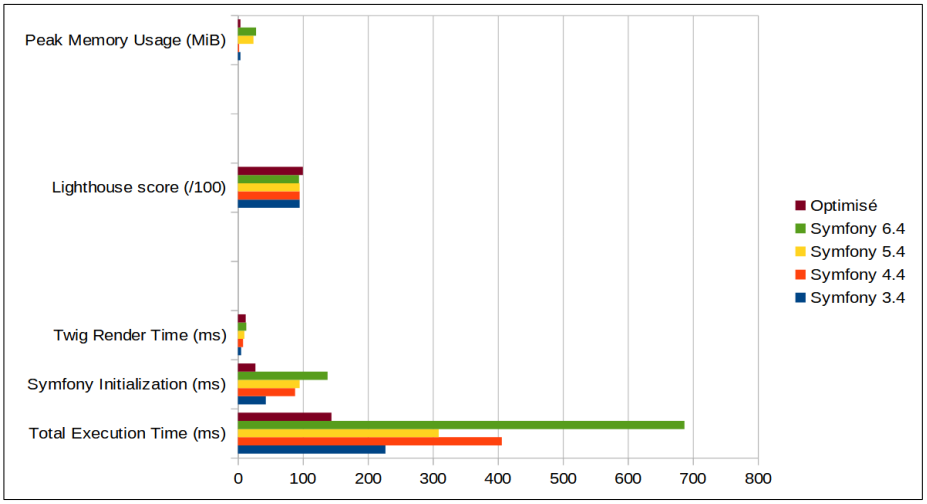


3.3 Page de connexion

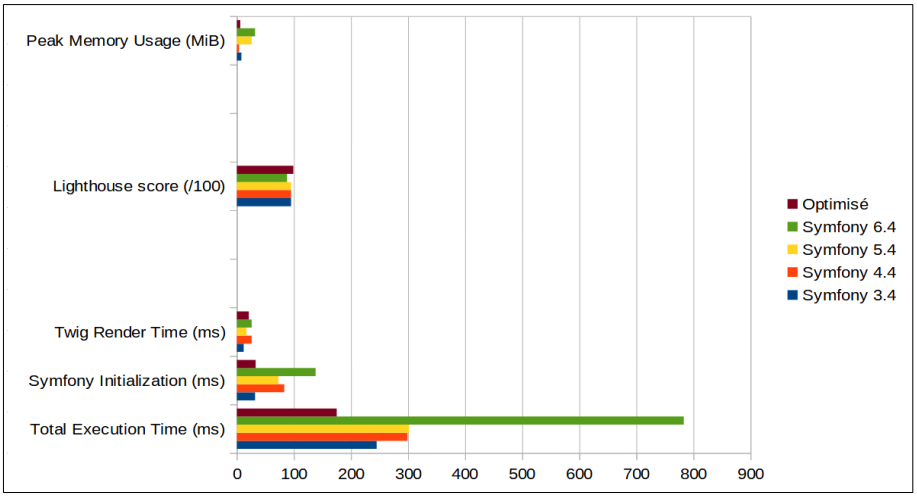


3.4 Les tâches

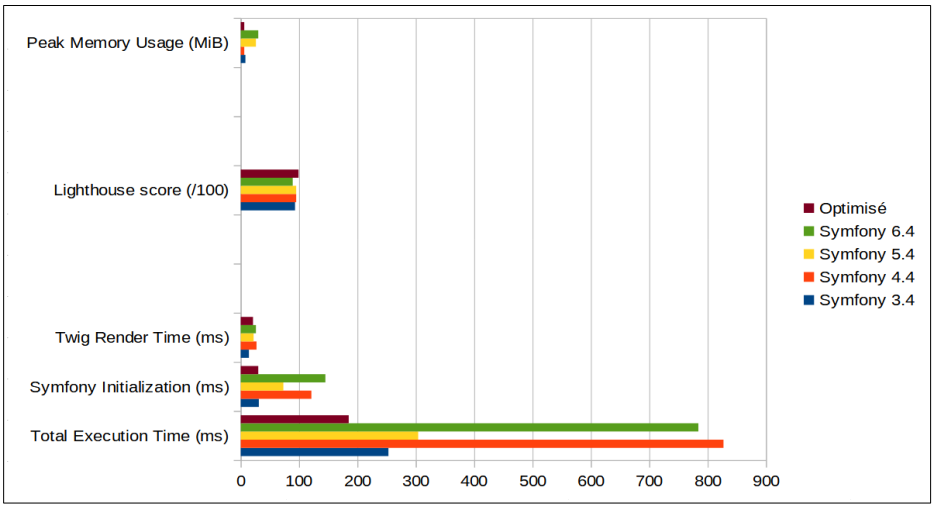
3.4.1 Listes des tâches



3.4.2 Ajouter une tâche

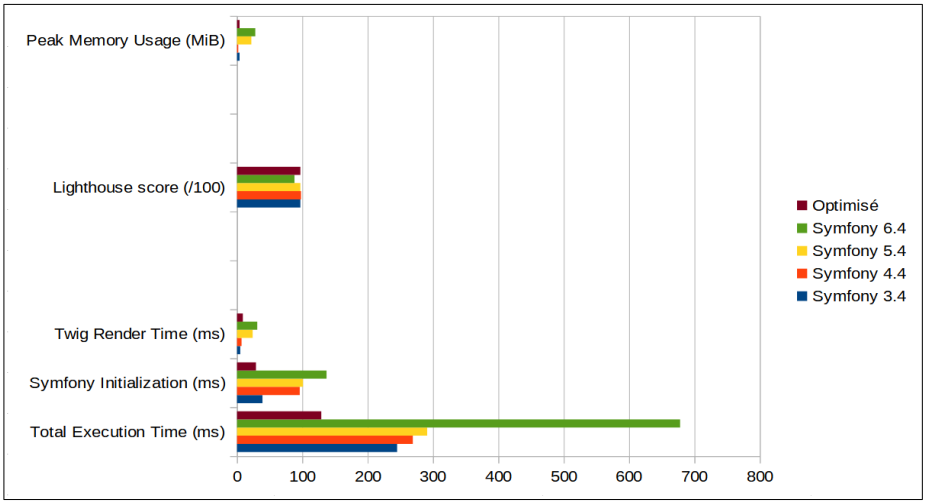


3.4.3 Modifier une tâche

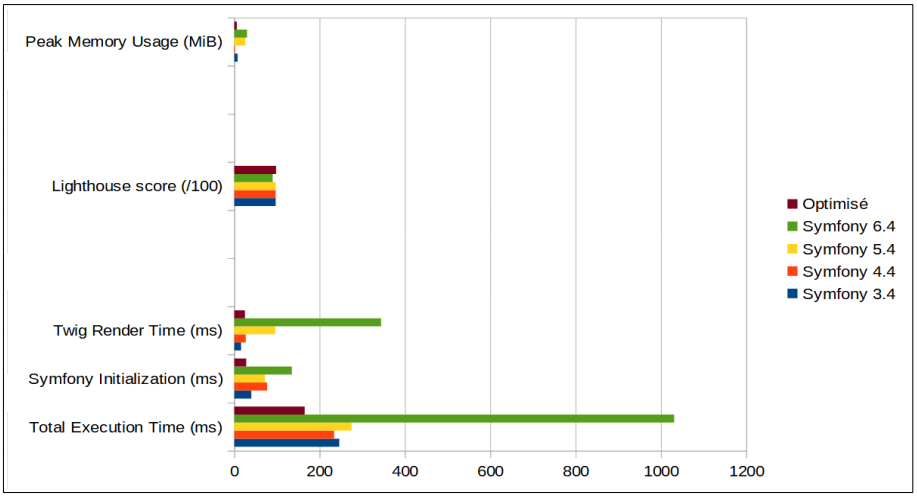


3.5 Les utilisateurs

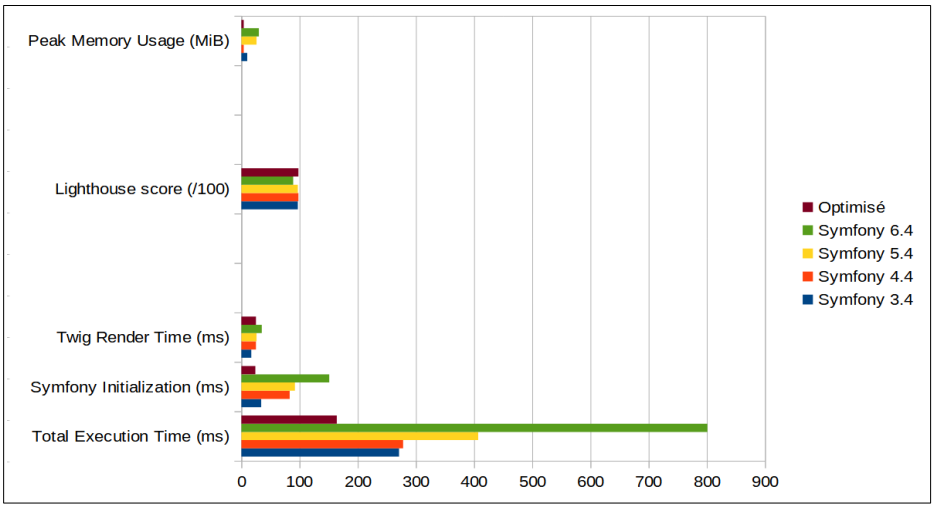
3.5.1 Listes des utilisateurs



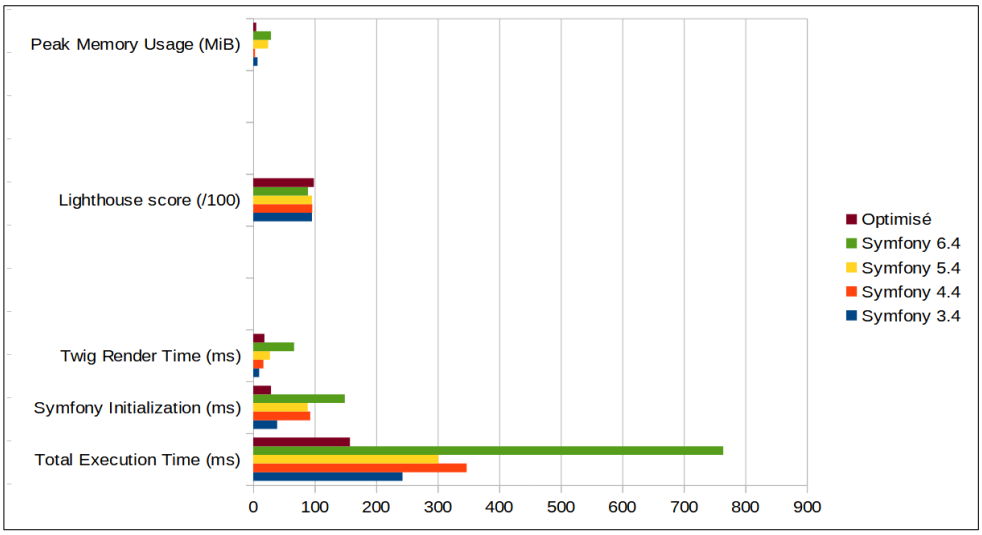
3.5.2 Ajouter un utilisateur



3.5.3 Modifier un utilisateur



3.6 Performance globale de l'application



**Le graphique ci-dessus est basé sur une moyenne de toutes les valeurs par version.*

Pour conclure cet audit de performance, on va comparé les performances globales de la version sous Symfony 3.4 et la version optimisée de l'application :

- **Total Execution Time (en ms) :** Le temps d'exécution total montre une réduction drastique par rapport à la version de Symfony 3.4.
  On est passé d'un temps total d'exécution de 242ms en moyenne à 157ms.
  C'est une baisse de 85ms soit d'environ 35.1 %.

- **Symfony Initialization Time (en ms) :** L'initialisation de Symfony montre aussi une amélioration considérable.
  On est passé de 39ms en moyenne à 29ms.
  C'est une baisse de 10ms, soit d'environ 25.6 %.

- **Twig Render Time (en ms) :** Le temps de rendu Twig montre une régression.
  On est passé de 9.7ms en moyenne à 18.2ms.
  C'est une augmentation de 8.5ms soit d'environ 87.2 %.

- **Peak Memory Usage (en MiB) :** La consommation de mémoire maximale est légèrement inférieure pour la version optimisée par rapport à Symfony 3.4, ce qui implique une meilleure optimisation de la mémoire.
  On est passé de 7.00MiB en moyenne à 5.00MiB.
  C'est une baisse de 2.00MiB soit d'environ 28.6 %.

- **Lighthouse Score (sur 100) :** Le score Lighthouse à légèrement augmenté, il est passé de 95.5/100 à 98.6/100.
  C'est une augmentation de 3.1 %.