

Audit

Qualité du code et performance

SOMMAIRE

- Contexte..... 1
 - Outils d'analyse* 1
 - Analyse de la qualité du code 1
 - Analyse de la performance 1
- Analyse de l'application sur sa version initiale..... 2
 - Rapport de qualité du code : Codacy*..... 2
 - Rapport de performance*..... 3
- Axes d'améliorations et mise en place de solutions 4
 - Architecture logiciel* 4
 - Environnement de travail*..... 5
 - Optimisation des performances*..... 5
 - Revue de code*..... 5
- Analyse de l'application à sa version finale 6
 - Rapport de qualité du code : Codacy* 6
 - Rapport de performance*..... 7
- Conclusion..... 8

Contexte

Todo List, développée par la startup ToDo & Co, est une application permettant à un utilisateur enregistré et connecté, de gérer ses tâches.

La stratégie de développement utilisée a été le MVP (Minimum Viable Product), permettant à la startup de présenter cette application rapidement et d'obtenir des retours d'éventuels investisseurs. Toutefois, ce choix reste une étape éphémère dans la phase de développement puisqu'à présent ToDo & Co souhaite améliorer la qualité de l'application par l'implémentation de nouvelles fonctionnalités et tests automatisés ainsi que la correction d'anomalies.

Avant d'attaquer la phase de travail sur le code de l'application, une analyse du projet initial focalisée sur sa qualité de code et de performance est donc nécessaire, afin d'en tirer d'éventuels besoins de mise à jour pour optimiser le suivi de développement de l'application et son utilisation par les futurs clients et usagers.

Vous trouverez donc dans ce document un rapport détaillé présentant :

- une analyse de l'application sur sa version initiale ;
- des axes d'améliorations possibles ;
- une analyse sur sa version actuelle.

Outils d'analyse

Voici une liste des outils qui ont servis à établir cet audit :

Outils	Description	Domaine
PHPStan	Analyse statique du code PHP afin de détecter les erreurs	Qualité du code
PHP CodeSniffer	Analyse et validation de la syntaxe du code PHP	
Codacy	Analyse statique du code	
Profiler de Symfony	Collecte des informations sur l'application	Performance

Analyse de la qualité du code

PHPStan

- a permis de détecter les différentes erreurs de code sans avoir à écrire de tests ;
- a été utilisé au niveau 8 afin d'avoir une analyse plus restrictive.

PHP CodeSniffer

- a permis de valider le respect des standards et normes PSR12.

Codacy

- a été un support d'automatisation de revue de code grâce à ses analyses faites pour chaque commit et pull request en lien avec le dépôt du projet hébergé sur la plateforme GitHub.

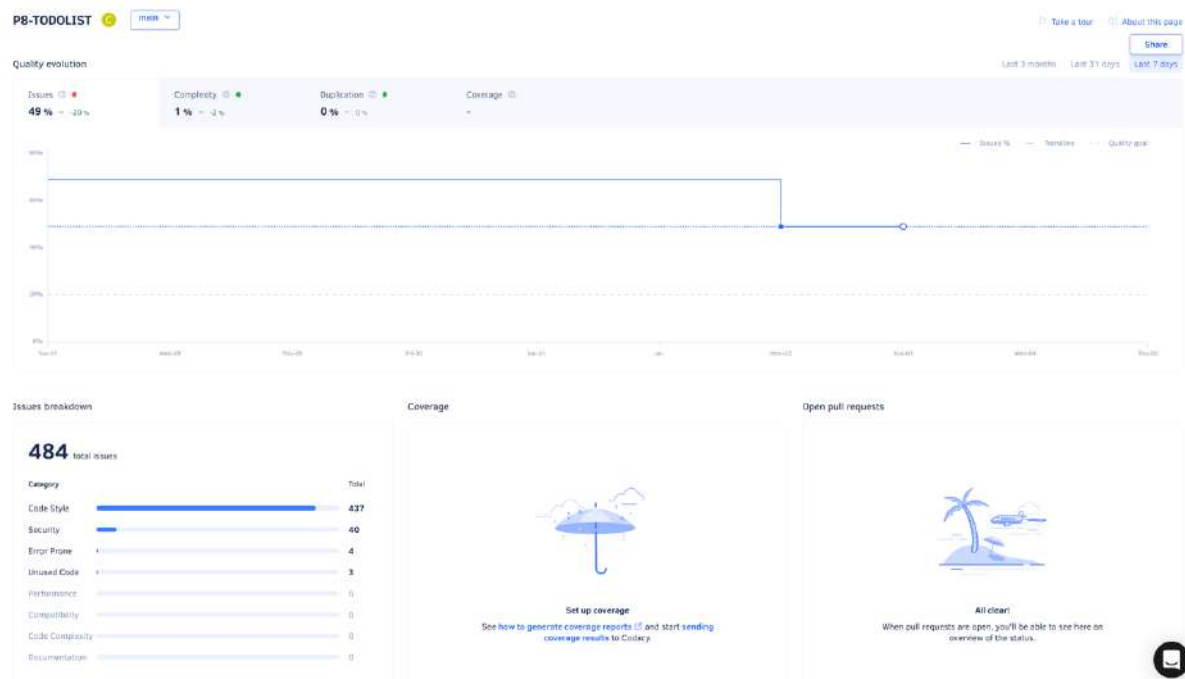
Analyse de la performance

Celle-ci s'est faite via le profiler de Symfony. Les données récoltées ont fait office d'un traitement d'analyse via tableur Excel, à retrouver dans le dossier *doc* à la racine du projet.

Analyse de l'application sur sa version initiale

Date	02 Janvier 2023
Version Symfony	3.4
PHP	7.2.34
Environnement de développement local	MAMP

Rapport de qualité du code : Codacy



Rapport Codacy du 02 Janvier 2023

Ce rapport fait état de **484 issues** et d'un pourcentage de fichiers complexes de 1%.
Codacy attribue un **badge de niveau C** à l'application.

Voici la répartition des issues par catégories :

Category	Issues total	Representative percentage
Code style	437	90 %
Security	40	8 %
Error prone	4	1 %
Unused Code	3	1 %

Rapport de performance

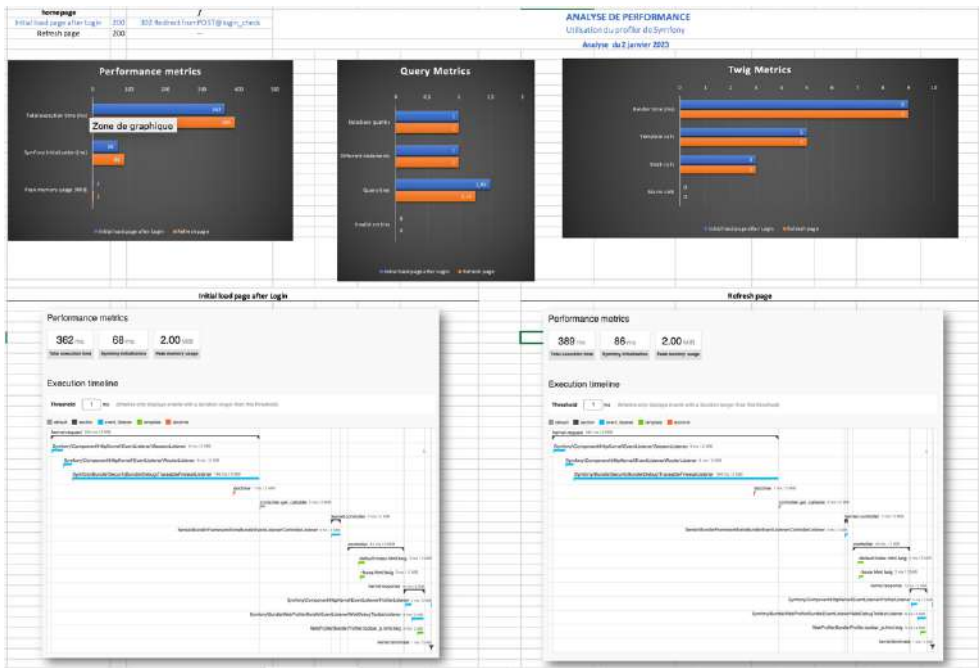
Liste des catégories de données collectées et leur description :

Performance metrics	Temps d'exécution de la page, temps d'initialisation de Symfony, quantité de mémoire maximale allouée au script
Twig Metrics	Données liées à la génération des vues
Query Metrics	Requêtes vers la base de données, temps d'exécution...

Aperçu du fichier Excel recensant les données collectées lors de l'analyse faite avec le Profiler de Symfony.

ANALYSE DE PERFORMANCE															
Utilisation du profiler de Symfony															
Analyse du 2 janvier 2023															
Step 1 -> bin/console cache:clear															
Step 2 -> symfony server:start															
Step 3 -> analyse															
Symfony: 3.4															
Page: 2.2.14															

Recensement des données d'analyse de performance récoltées via le Profiler de Symfony



Analyse graphique des données d'analyse de performance pour la page d'accueil de l'application

Axes d'améliorations et mise en place de solutions

Architecture logiciel

La première étape d'amélioration de l'existant s'est portée sur la migration du projet vers une version actuelle du framework sur laquelle repose son architecture.

La version initiale, Symfony 3.1, n'est plus maintenue depuis Janvier 2017 sur le plan de suivi des bugs et depuis Juillet 2017 pour la partie sécurité. Il était donc pertinent de la faire évoluer vers sa version actuelle, à savoir la 6.2.

Ceci nécessitera encore deux mises à jour mineures vers la version 6.4 qui sera - si l'on s'en réfère au passif d'évolution de ce framework - une version LTS (Long Term Support). Ceci laissera alors trois ans de répit à la startup sur cette partie puisque c'est la durée de vie de ces versions spéciales.

PHP a été également mis à jour depuis la version 5.6.40 à la 8.2.1 dans un souci de sécurité, sortie en décembre 2022 elle durera jusqu'en décembre 2025. Les versions inférieures à la 8 ne sont plus conseillées car elles peuvent être exposées à des vulnérabilités de sécurité non corrigées.

La montée de version du Framework s'est déroulée en plusieurs étapes, afin de résoudre au fur et à mesure les nombreuses dépréciations de dépendances et mises à jour des bibliothèques, pour ne pas influencer la logique et le code de l'application.

Versions	PHP
3.1	5.6.40
3.4 LTS	
4.0	
4.4 LTS	
5.0	
5.4 LTS	7.2.34
6.0	8.2.1
6.2	

Récapitulatif des différentes migrations

Environnement de travail

Afin de faciliter le travail de futurs développeurs associés à ce projet, un environnement Docker a été créé et leur permettra alors de travailler avec les outils déjà installés et paramétrés au projet. Plus de détails peuvent être trouvés dans le document [contributing.md](#) se trouvant à la racine du projet.

Optimisation des performances

La mise en place de l'extension OpCache peut optimiser l'exécution du code afin d'améliorer les performances, en stockant en mémoire les scripts PHP compilés en opcode. Ces fichiers seront alors chargés en priorité lors d'une nouvelle requête les nécessitant. Le temps de chargement des pages, devrait en être impacté lors de la prochaine analyse.

Revue de code

Un pre-commit hooks a été mis en place pour le suivi de la qualité du code. Ceci a permis d'avoir un suivi automatisé à chaque commit effectué pendant la phase de développement, débutant par le lancement d'une analyse de PHP Unit pour les tests, suivi par celles de PHPStan et PHP CodeSniffer.


```
➔ P8 git:(feature/#10) ✕ git commit -m "Authentication Implementation Documentation & update code #10"
Lancer une analyse de PHP Unit ? [y/n]
Lancer une analyse de PHP Stan ? [y/n]
Lancer une analyse de Code Sniffer ? [y/n]
Do you want to commit files ? [y/n]
```

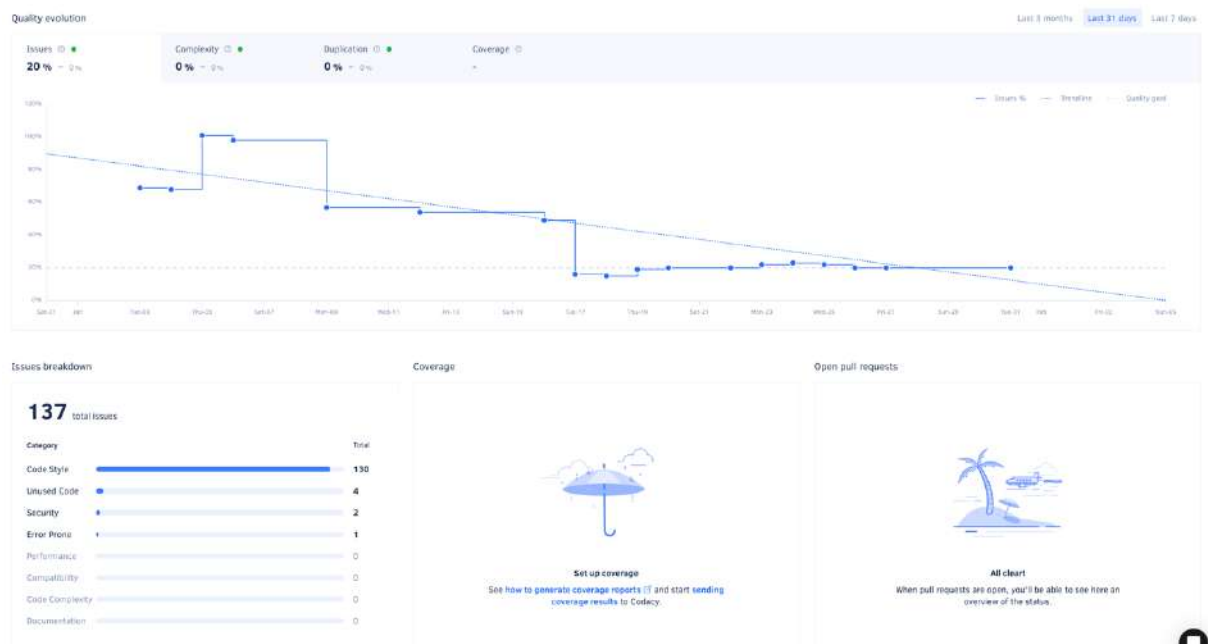
Automatisation des tests et analyses de code avant chaque commit

Analyse de l'application à sa version finale

Date	30 Janvier 2023
Version Symfony	6.2
PHP	8.1
Environnement de développement local	Docker

Rapport de qualité du code : Codacy

Codacy attribue désormais un badge de **niveau B** à l'application : **P8-TODOLIST** 
Le rapport fait état de **137 issues** dont un niveau de complexité des fichiers à 0%.



Rapport Codacy du 30 Janvier 2023

Voici un tableau présentant le total d'issues par catégories et leur taux de variation par rapport à la version initiale du projet :

Category	Issues - Version initiale	Issues - Version finale	Variation
Code style	437	130	-70 %
Security	40	4	-90 %
Error prone	4	2	-50 %
Unused Code	3	1	-66 %
TOTAL	484	137	-71 %

On peut voir que le résultat de cette mise en place de suivi de qualité de code a porté ses fruits puisque nous nous retrouvons actuellement avec une baisse significative de **71%** des issues total générées par Codacy.

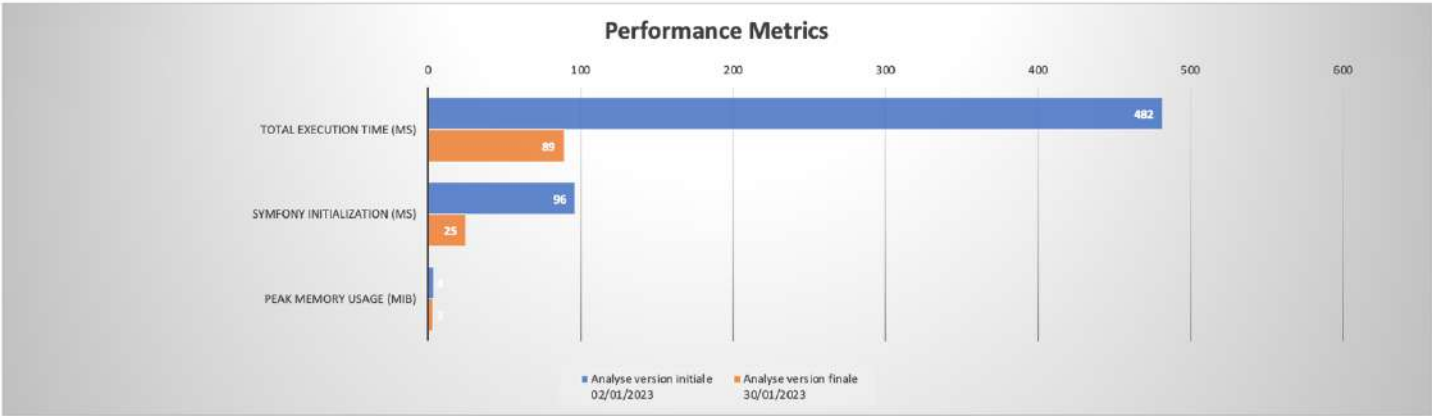
Il sera possible d'améliorer ce rendu dans le futur par une mise en place encore plus restrictive sur les analyses de code lors de pre-commit. Egalement la mise en place d'une configuration d'exclusions de fichiers spécifiques peut être envisagée afin de ne pas prendre en compte le code généré par symfony et des différentes librairies utilisées.

Rapport de performance

Suite à une analyse de la performance effectuée à nouveau via le profiler de Symfony voici un détail des métriques. Ces chiffres sont le résultat de moyennes calculées à partir des relevés de chaque catégorie.

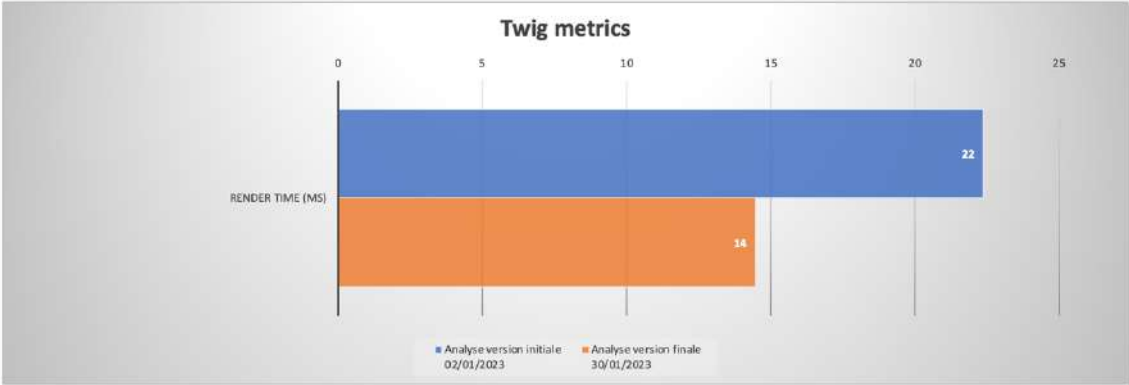
Analyse version initiale
02/01/2023
Analyse version finale
30/01/2023

Performance metrics		
Total execution time (ms)	Symfony initialization (ms)	Peak memory usage (MiB)
482	96	4
89	25	3



On peut constater que l'extension OPcache installée lors de la migration du projet, a permis d'améliorer considérablement les temps d'exécution au niveau du chargement des pages. C'est un gain de temps moyen de **81%** qui a été réalisé.

Twig Metrics
Render time (ms)
22
14

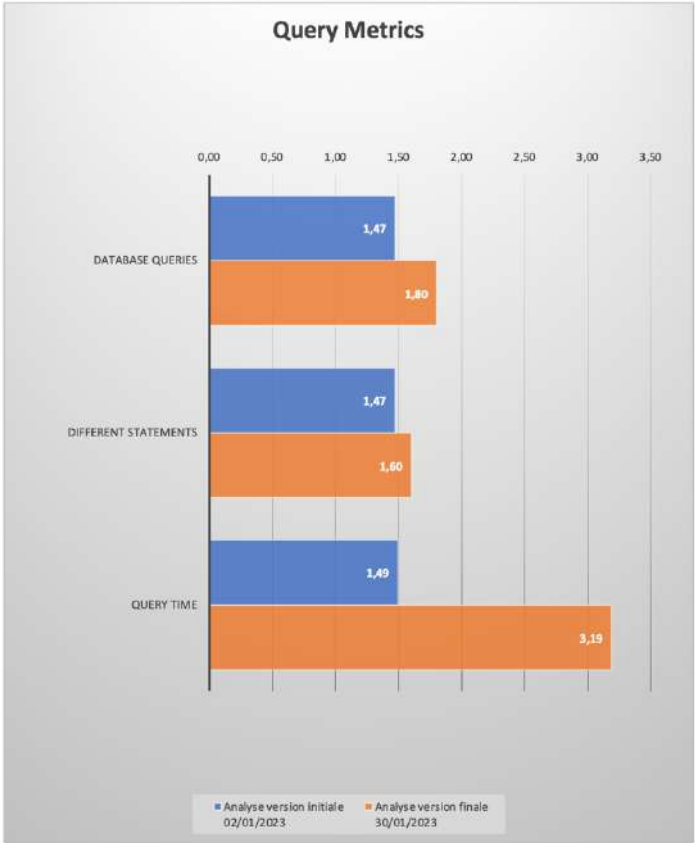


Le temps de rendu des templates s'est également amélioré, passant de 22ms à 14ms en moyenne.

Query Metrics		
Database queries	Different statements	Query time
1,47	1,47	1,49
1,80	1,60	3,19

On peut remarquer que les temps de requêtes sont plus long. Ceci s'explique par le fait que les requêtes, présentes lors de la première analyse, étaient moins complexes que celles qui ont été mises en place lors du travail effectué sur la correction d'anomalies et de nouvelles fonctionnalités.

A présent, les utilisateurs sont liés à leurs propres tâches et un administrateur a la possibilité de récupérer toutes les tâches de l'application en connaissant les détails de leur créateur. Ces relations ont donc complexifié les requêtes en apportant des jointures SQL entre les tables Task et User ce qui engendre alors un temps de traitement plus long.



Conclusion

- La migration du Framework Symfony vers sa dernière version permet de garantir la compatibilité des versions mineures et le support de sa prochaine version majeure pour les années futures.
- La mise en place d'un suivi de qualité de code via le pre-commit et les outils d'analyse permet à présent d'avoir une vision globale et une capacité de refactorisation au fur et à mesure de l'avancement et de l'évolution du projet.
- Pour l'analyse des performances, en l'état actuel des choses, le profiler nous a permis de constater le gain de temps d'exécution des pages réalisé grâce à la migration du framework et la mise en place de l'extension OpCache. Si les besoins s'en font sentir et que l'analyse a besoin d'être plus poussée, la startup a la possibilité de souscrire à un contrat payant auprès de Blackfire ou d'intégrer New Relic dans une réflexion d'intégration continue. Ces deux applications sont spécialisées dans l'analyse et le rapport de performances.