

# **ToDo&Co**

**Documentation technique**  
**Audit de qualité et de performance**

# Sommaire

Contexte du projet

Besoin du projet

version

Qualité de code

Synthèse

Tests: couverture du code de l'application

Performance de l'application

Bilan et plan d'amélioration

## Contexte du projet

**ToDo&Co** 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 (on parle de Minimum Viable Product ou MVP).

## Objectif du projet

**ToDo&Co** a enfin réussi à lever des fonds pour le développement de l'entreprise, le projet m'a été confié afin de prendre en charge l'amélioration de l'application.

Les charges sont les suivantes :

- Identifier et de corriger les anomalies
- Implémenter de nouvelles fonctionnalités
- Implémenter des tests automatisés
- une documentation expliquant comment l'implémentation de l'authentification a été faite
- Etablir un compte rendu de cet audit de qualité
- Proposer une série d'améliorations

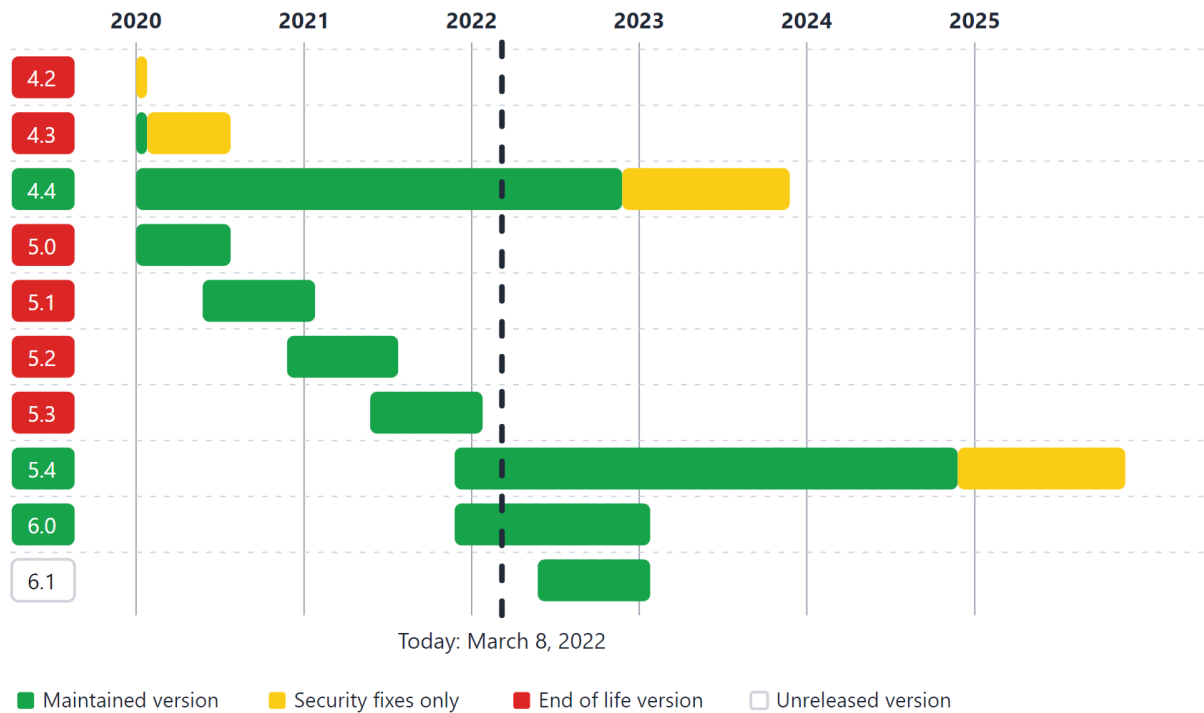
## Version

L'application a été mise à jour vers une version plus récente  
l'installation du projet initial en son état de MVP a permis de mettre en évidence un environnement technique obsolète, avec d'une part une version non maintenue

- Application avant la modification :  
Symfony 3.1.10  
php en 5.5.9  
doctrine-bundle 1.6  
doctrine-orm 2.5

D'après ce graphique la version la plus appropriée est la version 5 pour une plus longue période de maintenance

## Symfony Releases Calendar



## Qualité du code

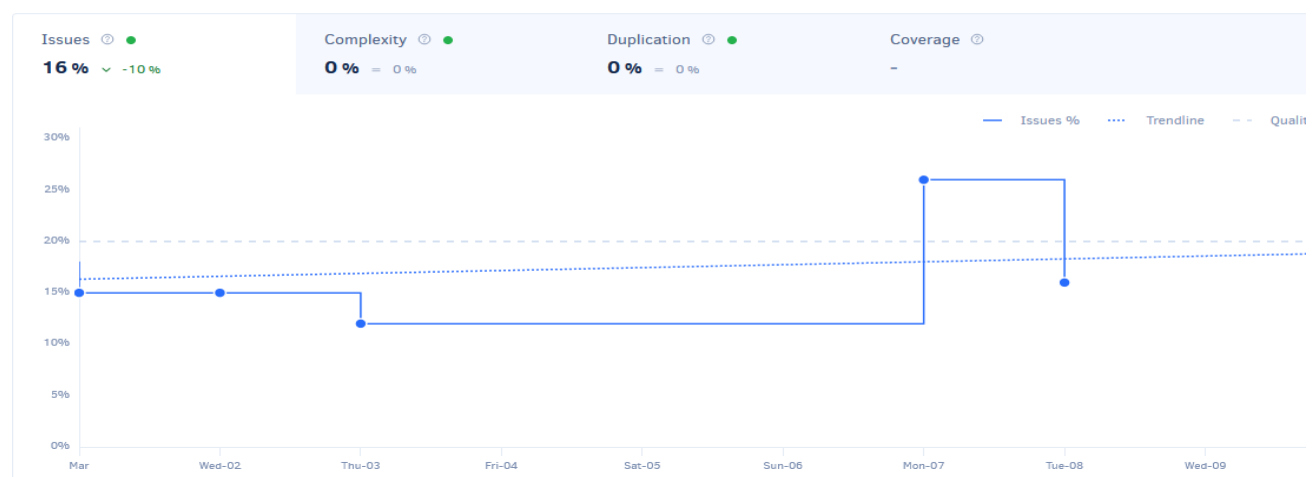
La qualité du code a été évaluée par l'outil d'automatisation Codacy qui permet de remonter différents problèmes pouvant être liés à la sécurité, la performance et le style du code.

ToDoList8 B master

[Take a tour](#) [A](#)

Quality evolution

Last 3 months Last 31 days



## Synthèse:

### Corrections d'anomalies

#### Une tâche doit être attachée à un utilisateur

Actuellement, lorsqu'une tâche est créée, elle n'est pas rattachée à un utilisateur. Il vous est demandé d'apporter les corrections nécessaires afin qu'automatiquement, à la sauvegarde de la tâche, l'utilisateur authentifié soit rattaché à la tâche nouvellement créée.

Lors de la modification de la tâche, l'auteur ne peut pas être modifié.

Pour les tâches déjà créées, il faut qu'elles soient rattachées à un utilisateur "anonyme".

#### Choisir un rôle pour un utilisateur

Lors de la création d'un utilisateur, il doit être possible de choisir un rôle pour celui-ci. Les rôles listés sont les suivants :

- rôle utilisateur (*ROLE\_USER*) ;
- rôle administrateur (*ROLE\_ADMIN*).

Lors de la modification d'un utilisateur, il est également possible de changer le rôle d'un utilisateur.

### Implémentation de nouvelles fonctionnalités

#### Autorisation

Seuls les utilisateurs ayant le rôle administrateur (*ROLE\_ADMIN*) doivent pouvoir accéder aux pages de gestion des utilisateurs.

Les tâches ne peuvent être supprimées que par les utilisateurs ayant créé les tâches en question.

Les tâches rattachées à l'utilisateur "anonyme" peuvent être supprimées uniquement par les utilisateurs ayant le rôle administrateur (*ROLE\_ADMIN*).
















#### Implémentation de tests automatisés

L'implémentation des tests automatisés (tests unitaires et fonctionnels) nécessaires pour assurer que le fonctionnement de l'application est bien en adéquation avec les demandes..

Vous prévoyez des données de tests afin de pouvoir prouver le fonctionnement dans les cas explicités dans ce document.

Un rapport de couverture de code au terme du projet. Il faut que le taux de couverture soit supérieur à 70 %

## Tests couverture du code de l'application

C:\wamp64\www\projet8\src / (Dashboard)									
	Code Coverage								
	Lines			Functions and Methods			Classes and Traits		
Total		94.69%	107 / 113		97.50%	39 / 40		90.00%	9 / 10
■ Controller		90.77%	59 / 65		90.00%	9 / 10		75.00%	3 / 4
■ DataFixtures		n/a	0 / 0		n/a	0 / 0		n/a	0 / 0
■ Entity		100.00%	39 / 39		100.00%	26 / 26		100.00%	2 / 2
■ Form		100.00%	7 / 7		100.00%	2 / 2		100.00%	2 / 2
■ Repository		100.00%	2 / 2		100.00%	2 / 2		100.00%	2 / 2
■ 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%

Une couverture 94% a été implémentée

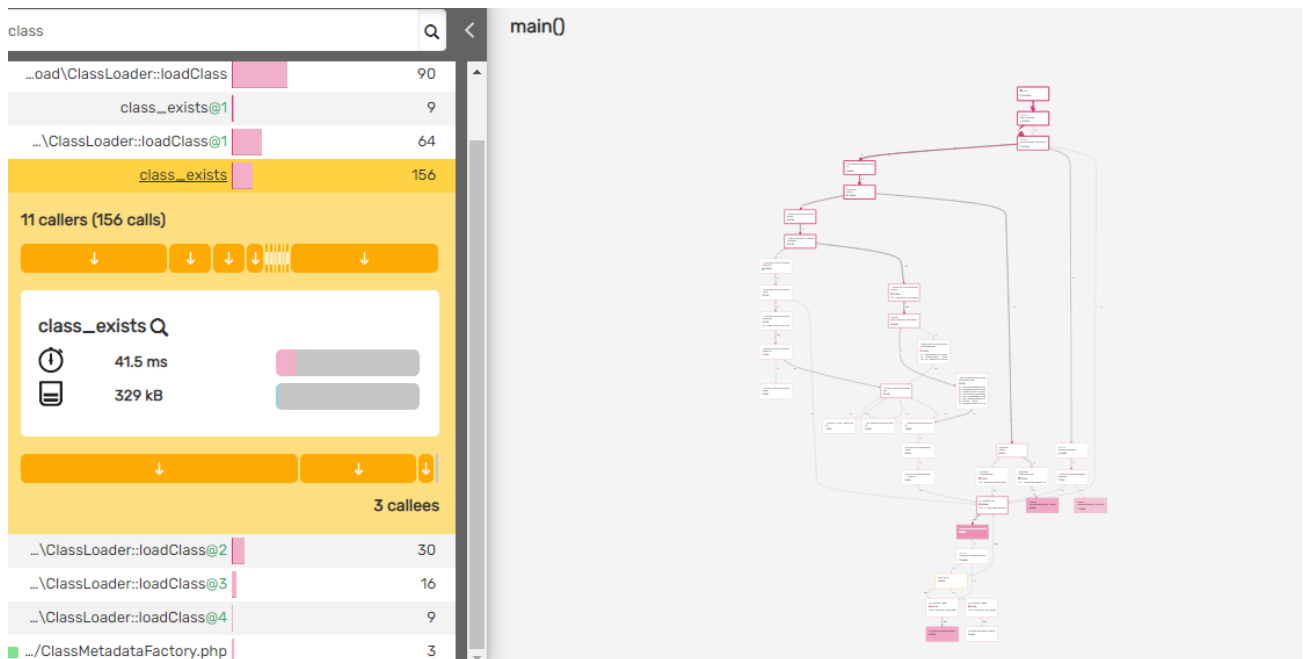
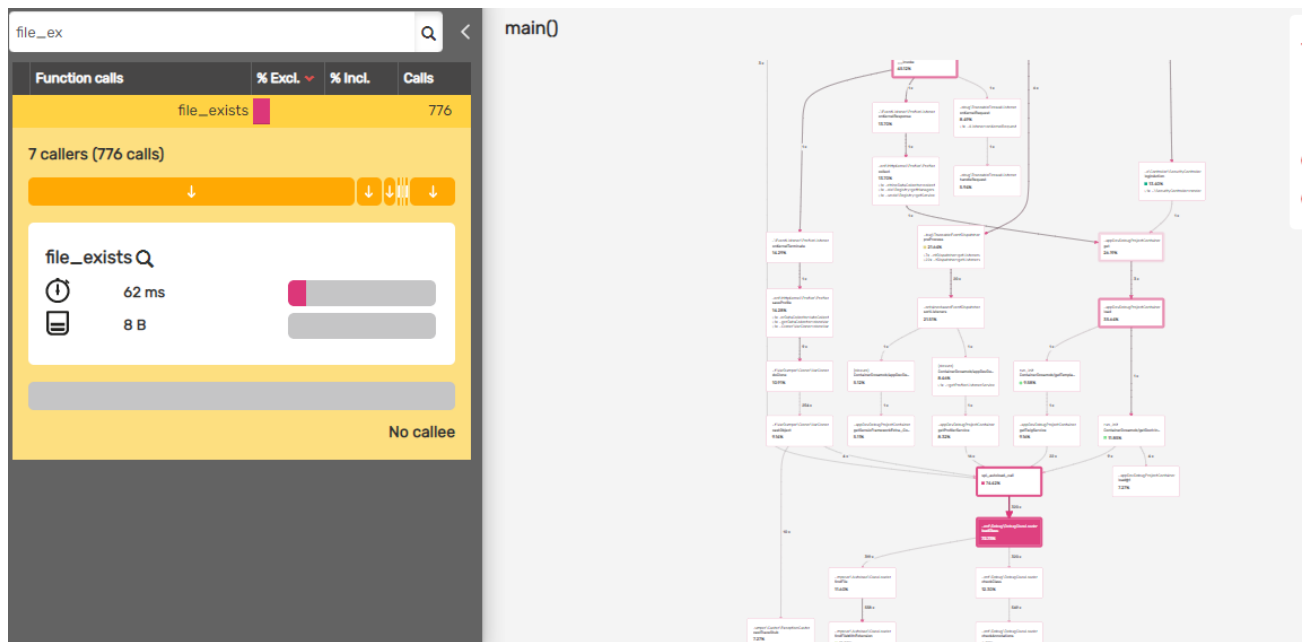
## Performance de l'application

Blackfire.io est un service disponible sur navigateur et en ligne de commande pour gérer la performance de vos applications PHP. C'est avant tout un **profileur bien plus performant** que celui fourni dans le Framework Symfony, mais c'est également bien plus que cela :

- son empreinte est minimale et ne requiert **aucune modification** de votre code ;
- il permet de trouver **les causes** de vos problèmes de performance ;
- la plateforme stocke et permet d'**analyser et de comparer vos profils** de mesure sur la durée et de façon collaborative ;
- il est parfaitement intégré avec les solutions de test et d'intégration continue, pour **éliminer les régressions** de performance.

search <span>Q</span>			
Function calls	% Excl. ▾	% Incl.	Calls
...poser\Autoload\includeFile	<div><div></div></div>		89
...ser\Autoload\includeFile@1	<div><div></div></div>		64
...lProdContainer.php/97-242	<div><div></div></div>		1
...poser/autoload_static.php	<div><div></div></div>		1
...r_Private_TwigService::do	<div><div></div></div>		1
...__KernelProdContainer::load	<div><div></div></div>		22
...er\Autoload\includeFile@2	<div><div></div></div>		30
...ernelProdContainer::load@1	<div><div></div></div>		12
...ltEntityManagerService::do	<div><div></div></div>		2
...er\Autoload\includeFile@3	<div><div></div></div>		16
...pile::ORM/UnitOfWork.php	<div><div></div></div>		1
...50f05c37ea2fbddb4faeb0	<div><div></div></div>		14
...entDispatcher.php/265-271	<div><div></div></div>		28
...5l\getTranslatorService::do	<div><div></div></div>		1
...ntityManager::__construct	<div><div></div></div>		1
...__KernelProdContainer.php	<div><div></div></div>		1
...orms/AbstractPlatform.php	<div><div></div></div>		1

L'analyse de Blackfire donne : le temps de chargement du site, la mémoire utiliser pour son chargement, les fonctions utilisées et son nombre d'appels. Chaque fonction est détaillée avec son temps d'exécution et la mémoire utilisée.



La diminution du nombre d'appels des fonctions file\_exists de l'autoloader de Composer



## **Bilan et plan d'amélioration**

### **Fonctionnalité**

Ajout de la Pagination, filtres, recherches pour les tâches/

Ajouter la possibilité de supprimer un utilisateur, dans la limite d'avoir toujours au moins un compte administrateur actif

Pouvoir supprimer un utilisateur

Améliorer le design de l'application

Ajout d'un commentaire

### **OPCache :**

*OPcache améliore les performances de PHP en stockant le bytecode des scripts précompilés en mémoire partagée, faisant ainsi qu'il n'est plus nécessaire à PHP de charger et d'analyser les scripts à chaque demande*

### **Hébergeur :**

Hébergement web sera performant et rapide, plus votre site sera apprécié par les visiteurs. En plus, l'aspect vitesse joue un rôle toujours plus grand concernant le référencement de votre site sur Google. Il ne faut donc surtout pas négliger cet aspect (mais aussi tous les autres qui composent une offre – car un hébergement web reste un ensemble de fonctionnalités, de services...).