



ToDo & Co : Audit de la Qualité du Code et des Performances de l'Application



20 AOUT 2023

MEHDI HADDOU | DAPS - P8

SOMMAIRE

I. CONTEXTE.....	page 2
II. ÉTAPES CLÉS DE LA RÉALISATION DE LA MISSION	page 3
1. Installation de l'environnement de développement.....	page 3
2. Implémentation de Fixtures pour les tests	page 3
3. Mesures sur la qualité du code	page 3
4. Mesures de performance	page 3
5. Tests fonctionnels complets	page 3
6. Migration vers Symfony 5.4	page 5
7. Adaptation du code à Symfony 5.4	page 5
8. Correction des erreurs détectés.....	page 6
9. Correction d'anomalies	page 6
10. Implémentation de Nouvelles Fonctionnalités	page 7
11. Nouvelles mesures sur la qualité du code	page 8
12. Nouvelles mesures de performance	page 8
III. QUALITE DU CODE : STANDARDS ET BONNES PRATIQUES	page 8
IV. SUIVI DES PERFORMANCES DE L'APPLICATION	page 12
1. Formulaire de connexion.....	page 12
2. Page d'accueil	page 12
3. Page de la liste des tâches.....	page 12
4. Formulaire de création d'une nouvelle tâche	page 12
5. Formulaire d'édition d'une tâche.....	page 13
6. Page de la liste des utilisateurs	page 13
7. Formulaire de création d'un nouvel utilisateur.....	page 13
8. Formulaire d'édition d'un utilisateur	page 13
V. INTERFACE UTILISATEUR (UX) : INTERFACE ET DESIGN	page 14
1. Absence de besoins spécifiques dans le cahier des charges.....	page 14
2. Prise de décision collective	page 14
3. Optimisation continue.....	page 14
VI. CONCLUSION.....	page 15

Audit de la Qualité du Code et des Performances de l'Application

L'audit de la qualité du code et des performances d'une application revêt une importance cruciale dans le développement d'application moderne. Il s'agit d'une démarche essentielle pour garantir un fonctionnement optimal, une expérience utilisateur fluide et maintenir une base de code robuste et pérenne. Dans ce rapport, nous présenterons les différentes étapes de l'audit effectué sur l'application, mettant l'accent sur les aspects liés à la qualité du code, aux performances et à l'expérience utilisateur.

I. CONTEXTE

Dans le contexte d'une startup en pleine expansion, spécialisée dans la gestion des tâches quotidiennes, l'application **ToDo & Co** joue un rôle central. Cette application a été développée en mode accéléré, initialement dans le but de présenter un produit minimum viable (MVP) aux investisseurs potentiels. Le choix du framework PHP Symfony a été fait par le développeur précédent pour sa familiarité et sa robustesse.

La récente levée de fond permet à **ToDo & Co** de poursuivre son développement et de perfectionner l'application. En tant que développeur expérimenté, la mission est d'améliorer la qualité globale de l'application. Dans ce cadre, le rôle inclut les responsabilités suivantes :

- Implémentation de nouvelles fonctionnalités : Enrichir l'application en ajoutant de nouvelles fonctionnalités répondant aux besoins de l'entreprise et des utilisateurs.
- Correction d'anomalies : Identifier et corriger les anomalies existantes, améliorant ainsi la stabilité et la fiabilité de l'application.
- Implémentation de tests automatisés : Développement de tests automatisés pour garantir le bon fonctionnement des fonctionnalités et prévenir les régressions.

En outre, l'audit de qualité qui sera mener permettra d'évaluer en profondeur le code et les performances de l'application. Cette analyse complète donnera une vue d'ensemble des axes d'amélioration possibles et des opportunités de réduction de la dette technique.

Il est important de noter que l'audit ne requiert pas nécessairement des corrections immédiates des problèmes identifiés. Cependant, ToDo & Co appréciera tout effort visant à réduire la dette technique et à renforcer la qualité de l'application.

II. ÉTAPES CLES DE LA REALISATION DE LA MISSION

Le cheminement suivi pour mener à bien la mission confiée par le contexte s'est articulé autour de plusieurs étapes clés. Chaque étape a été soigneusement planifiée et exécutée pour garantir une amélioration significative de la qualité et des performances de l'application **ToDo & Co** :

1. Installation de l'environnement de développement :

La première étape a consisté à mettre en place un environnement de développement, conforme aux spécifications requises par l'application initiale. Cette installation a impliqué la configuration de Symfony 3.1, PHP 5.6 et MySQL 5.7 sur une version Ubuntu 22.04.

2. Implémentation de Fixtures pour les tests :

Pour tester le fonctionnement de l'application dans cet environnement, des fixtures ont été mises en place. Ces données de test ont permis de simuler des scénarios d'utilisation réelle et d'évaluer la réactivité de l'application.

3. Mesures sur la qualité du code :





Des outils de qualité tels que PHPCS, PHPStan et Symfony Insight ont été utilisés pour analyser en profondeur le code source. Les mesures de qualité ont permis d'identifier les zones nécessitant des améliorations en termes de normes de codage, de bonnes pratiques et de lisibilité.

4. Mesures de performance :

Les performances de l'application ont été évaluées à la fois sur un serveur local et distant. Les temps de réponse, la consommation de ressources et d'autres métriques ont été relevés pour déterminer l'efficacité opérationnelle de l'application.

5. Tests fonctionnels complets :

Des tests fonctionnels exhaustifs ont été développés pour couvrir toutes les fonctionnalités de l'application initiale. Ces tests ont été conçus pour être réutilisables lors de la migration vers Symfony 5.4.

	Code Coverage								
	Lines			Functions and Methods			Classes and Traits		
Total	<div><div></div></div>	95.19%	99 / 104	<div><div></div></div>	88.24%	30 / 34	<div><div></div></div>	75.00%	6 / 8
 Controller	<div><div></div></div>	96.77%	60 / 62	<div><div></div></div>	83.33%	10 / 12	<div><div></div></div>	75.00%	3 / 4
 Entity	<div><div></div></div>	89.66%	26 / 29	<div><div></div></div>	90.00%	18 / 20	<div><div></div></div>	50.00%	1 / 2
 Form	<div><div></div></div>	100.00%	13 / 13	<div><div></div></div>	100.00%	2 / 2	<div><div></div></div>	100.00%	2 / 2
 AppBundle.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%

Generated by php-code-coverage 4.0.8 using PHP 5.6.40-67+ubuntu22.04.1+deb.sury.org+1 with Xdebug 2.5.5 and PHPUnit 5.7.27 at Thu Aug 17 11:54:46 CEST 2023.

Nous constatons une couverture de 75% sur la totalité des fonctionnalités offertes par l'application.

	Code Coverage									
	Classes and Traits			Functions and Methods				Lines		
Total	<div><div></div></div>	0.00%	0 / 1	<div><div></div></div>	33.33%	1 / 3	<div><div></div></div> CRAP	<div><div></div></div>	77.78%	7 / 9
SecurityController	<div><div></div></div>	0.00%	0 / 1	<div><div></div></div>	33.33%	1 / 3	3.10	<div><div></div></div>	77.78%	7 / 9
loginAction				<div><div></div></div>	100.00%	1 / 1	1	<div><div></div></div>	100.00%	7 / 7
loginCheck				<div><div></div></div>	0.00%	0 / 1	2	<div><div></div></div>	0.00%	0 / 1
logoutCheck				<div><div></div></div>	0.00%	0 / 1	2	<div><div></div></div>	0.00%	0 / 1

Les fonctions loginCheck et logoutCheck ne sont jamais exécutés, ces fonctions sont tout de même nécessaires au bon fonctionnement de l'application au vu du provider pour le système d'authentification paramétré dans les fichiers de config.

	Code Coverage									
	Classes and Traits			Functions and Methods				Lines		
Total	<div></div>	0.00%	0 / 1	<div></div>	80.00%	8 / 10	CRAP	<div></div>	81.25%	13 / 16
Task	<div></div>	0.00%	0 / 1	<div></div>	80.00%	8 / 10	10.66	<div></div>	81.25%	13 / 16
__construct				<div></div>	100.00%	1 / 1	1	<div></div>	100.00%	3 / 3
getId				<div></div>	100.00%	1 / 1	1	<div></div>	100.00%	1 / 1
getCreatedAt				<div></div>	0.00%	0 / 1	2	<div></div>	0.00%	0 / 1
setCreatedAt				<div></div>	0.00%	0 / 1	2	<div></div>	0.00%	0 / 2
getTitle				<div></div>	100.00%	1 / 1	1	<div></div>	100.00%	1 / 1
setTitle				<div></div>	100.00%	1 / 1	1	<div></div>	100.00%	2 / 2
getContent				<div></div>	100.00%	1 / 1	1	<div></div>	100.00%	1 / 1
setContent				<div></div>	100.00%	1 / 1	1	<div></div>	100.00%	2 / 2
isDone				<div></div>	100.00%	1 / 1	1	<div></div>	100.00%	1 / 1
toggle				<div></div>	100.00%	1 / 1	1	<div></div>	100.00%	2 / 2

Les getter/setter de la propriété « createdAt » de la classe « Task » ne sont jamais exécutés dans l'application. En effet, la date de création d'une tâche est défini dans le constructeur de l'instance « Task » et s'applique directement à la propriété sans passer par le setter.

6. Migration vers Symfony 5.4 :

L'application a été migré vers Symfony 5.4 (Version LTS), une étape cruciale pour bénéficier des dernières améliorations et de la stabilité à long terme offertes par cette version. Mise en place d'un environnement PHP 8.1 et MySQL 8 pour accueillir et faire tourner l'application Symfony 5.4.

7. Adaptation du code à Symfony 5.4 :

Le code source et les tests ont été adaptés pour être compatibles avec Symfony 5.4. Les ajustements ont été apportés jusqu'à ce que tous les tests passent avec succès.

Couverture des tests après migration, avant correction des anomalies et des ajouts des nouvelles fonctionnalités :

/home/styx/Documents/projects/PHP/P8/Project/src / (Dashboard)

		Code Coverage					
		Lines		Functions and Methods		Classes and Traits	
Total	<div><div></div></div>	95.86%	139 / 145	<div><div></div></div>	90.70%	39 / 43	<div><div></div></div> 72.73%8 / 11
■ Controller	<div><div></div></div>	98.70%	76 / 77	<div><div></div></div>	90.91%	10 / 11	<div><div></div></div> 75.00%3 / 4
■ Entity	<div><div></div></div>	84.38%	27 / 32	<div><div></div></div>	86.36%	19 / 22	<div><div></div></div> 0.00%0 / 2
■ Form	<div><div></div></div>	100.00%	23 / 23	<div><div></div></div>	100.00%	4 / 4	<div><div></div></div> 100.00%2 / 2
■ Repository	<div><div></div></div>	100.00%	2 / 2	<div><div></div></div>	100.00%	2 / 2	<div><div></div></div> 100.00%2 / 2
■ Security	<div><div></div></div>	100.00%	11 / 11	<div><div></div></div>	100.00%	4 / 4	<div><div></div></div> 100.00%1 / 1
Kernel.php		n/a	0 / 0		n/a	0 / 0	n/a0 / 0

Legend

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

Generated by php-code-coverage 9.2.27 using PHP 8.2.8 and PHPUnit 9.5.28 at Thu Aug 17 9:46:34 UTC 2023.




Avec la version 5.4 de Symfony, nous constatons une couverture de 72,73% sur la totalité des fonctionnalités offertes par l'application.

/home/styx/Documents/projects/PHP/P8/Project/src / Controller / SecurityController.php

		Code Coverage					
		Lines		Functions and Methods		Classes and Traits	
Total	<div><div></div></div>	85.71%	6 / 7	<div><div></div></div>	50.00%	1 / 2	<div><div></div></div> 0.00%0 / 1
SecurityController	<div><div></div></div>	85.71%	6 / 7	<div><div></div></div>	50.00%	1 / 2	<div><div></div></div> 0.00%0 / 1
login	<div><div></div></div>	100.00%	6 / 6	<div><div></div></div>	100.00%	1 / 1	<div><div></div></div> 1
logout	<div><div></div></div>	0.00%	0 / 1	<div><div></div></div>	0.00%	0 / 1	<div><div></div></div> 2






























Le SecurityController ne prend plus en compte la route « login_check ». La fonction logout, à l'image de la version 3.1 ne sera jamais exécuté non plus mais reste nécessaire au bon fonctionnement de l'application.

/home/styx/Documents/projects/PHP/P8/Project/src / Entity / Task.php

	Code Coverage									
	Lines				Functions and Methods				Classes and Traits	
Total		78.57%	11 / 14		80.00%	8 / 10	CRAP		0.00%	0 / 1
.Task		78.57%	11 / 14		80.00%	8 / 10	10.98		0.00%	0 / 1
__construct		100.00%	1 / 1		100.00%	1 / 1	1			
getId		100.00%	1 / 1		100.00%	1 / 1	1			
getTitle		100.00%	1 / 1		100.00%	1 / 1	1			
setTitle		100.00%	2 / 2		100.00%	1 / 1	1			
getContent		100.00%	1 / 1		100.00%	1 / 1	1			
setContent		100.00%	2 / 2		100.00%	1 / 1	1			
getCreatedAt		0.00%	0 / 1		0.00%	0 / 1	2			
setCreatedAt		0.00%	0 / 2		0.00%	0 / 1	2			
isDone		100.00%	1 / 1		100.00%	1 / 1	1			
toggle		100.00%	2 / 2		100.00%	1 / 1	1			

La classe « Task » contient le même résultat que la version 3.1 étant donné que les getter/setter ne sont jamais utilisés dans les fonctionnalités de l'application.

/home/styx/Documents/projects/PHP/P8/Project/src / Entity / User.php

	Code Coverage									
	Lines				Functions and Methods				Classes and Traits	
Total		88.89%	16 / 18		91.67%	11 / 12	CRAP		0.00%	0 / 1
.User		88.89%	16 / 18		91.67%	11 / 12	12.20		0.00%	0 / 1
getId		100.00%	1 / 1		100.00%	1 / 1	1			
getUsername		100.00%	1 / 1		100.00%	1 / 1	1			
setUsername		100.00%	2 / 2		100.00%	1 / 1	1			
getUserIdentifier		100.00%	1 / 1		100.00%	1 / 1	1			
setRoles		100.00%	3 / 3		100.00%	1 / 1	1			
setRoles		0.00%	0 / 2		0.00%	0 / 1	2			
getPassword		100.00%	1 / 1		100.00%	1 / 1	1			
setPassword		100.00%	2 / 2		100.00%	1 / 1	1			
getSalt		100.00%	1 / 1		100.00%	1 / 1	1			
eraseCredentials		100.00%	1 / 1		100.00%	1 / 1	1			
getEmail		100.00%	1 / 1		100.00%	1 / 1	1			
setEmail		100.00%	2 / 2		100.00%	1 / 1	1			

Dans la classe « User », nous remarquons une nouvelle propriété « rôle » qui n'était pas présente dans la version 3.1. En effet, lorsque nous avons adapté l'authentification avec la version 5.4, celle-ci l'a ajouté par défaut.

8. Correction des erreurs détectés :

Les outils PHPCS, PHPStan et Symfony Insight ont mis en évidence diverses erreurs et problèmes dans le code. Chacun de ces problèmes a été corrigé méthodiquement pour garantir la robustesse et la fiabilité du code.

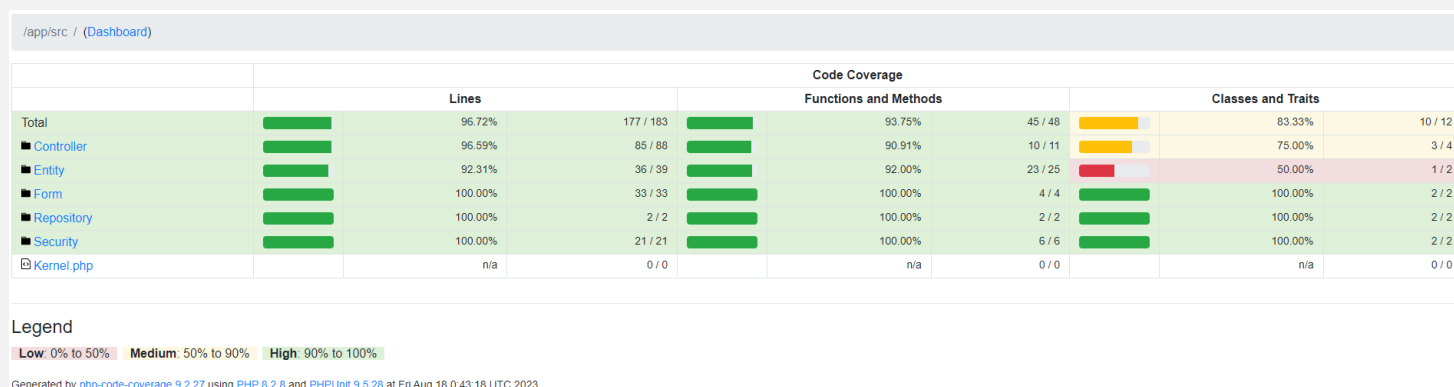
9. Correction d'anomalies :

Les anomalies relevées dans le cahier des charges ont été résolues en suivant une approche de test-driven development (TDD). Les tests correspondants ont été rédigés avant d'apporter des modifications au code, garantissant ainsi l'intégrité des fonctionnalités.

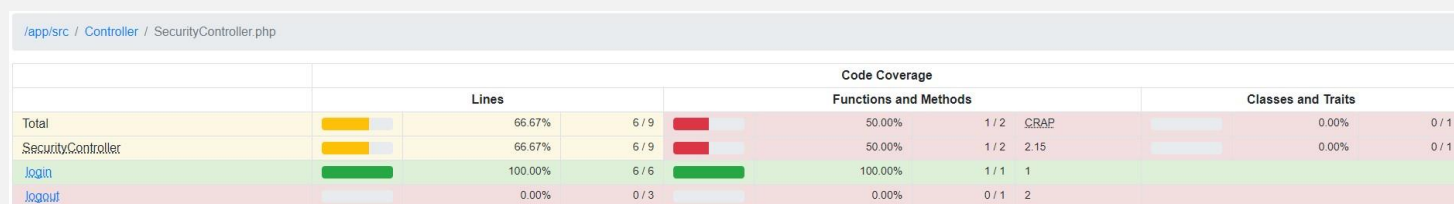
10. Implémentation de Nouvelles Fonctionnalités :

De nouvelles fonctionnalités ont été développées conformément aux exigences spécifiées en suivant une approche de TDD.

Résultats de la couverture des tests après correction des anomalies et ajouts des nouvelles fonctionnalités :


















Nous constatons une couverture de 83.33% sur la totalité des fonctionnalités offertes par l'application.



Même constat que les anciens tests.



Nous remarquons qu'il y a une nouvelle propriété pour l'auteur de la tâche, celle-ci avait été demandé dans les correction des anomalies. Ce résultat confirme la bonne utilisation de la propriété dans la nouvelle version de l'application. En revanche, les getter/setter pour la date de création ne sont toujours pas corrigés volontairement. En effet, lorsque la partie « front-end » de l'application sera en cours d'amélioration, il est fort probable que celle-ci sera affichée.

				Code Coverage					
		Lines			Functions and Methods			Classes and Traits	
Total		100.00%	22 / 22		100.00%	13 / 13	CRAP		100.00% 1 / 1
User		100.00%	22 / 22		100.00%	13 / 13	14		100.00% 1 / 1
getId		100.00%	1 / 1		100.00%	1 / 1	1		
getUsername		100.00%	1 / 1		100.00%	1 / 1	1		
setUsername		100.00%	2 / 2		100.00%	1 / 1	1		
getUserIdentifier		100.00%	1 / 1		100.00%	1 / 1	1		
getRoles		100.00%	3 / 3		100.00%	1 / 1	1		
setRoles		100.00%	2 / 2		100.00%	1 / 1	1		
getPassword		100.00%	1 / 1		100.00%	1 / 1	1		
setPassword		100.00%	2 / 2		100.00%	1 / 1	1		
getSalt		100.00%	1 / 1		100.00%	1 / 1	1		
eraseCredentials		100.00%	1 / 1		100.00%	1 / 1	1		
getEmail		100.00%	1 / 1		100.00%	1 / 1	1		
setEmail		100.00%	2 / 2		100.00%	1 / 1	1		
isAdmin		100.00%	4 / 4		100.00%	1 / 1	2		

Nous pouvons constater que le setter pour le rôle d'un utilisateur est bien couvert contrairement à la couverture précédente. En effet, en ajoutant la fonctionnalité sur la possibilité de définir un rôle à un utilisateur lors de sa création et sa modification, celle-ci appelle le setter afin de définir le rôle de l'utilisateur.

11. Nouvelles mesures sur la qualité du code :

Les outils de qualité ont été réutilisés pour évaluer la qualité du code après les modifications et les ajouts. Ces mesures ont permis de confirmer les améliorations apportées.

12. Nouvelles mesures de performance :

Les performances de l'application ont été réévaluées après les changements, en tenant compte des nouvelles fonctionnalités et des modifications apportées. Les mesures ont été prises à la fois sur un serveur local et distant pour obtenir une vue complète.

III. QUALITE DU CODE : STANDARDS ET BONNES PRATIQUES

La qualité du code est un pilier fondamental dans le développement d'applications logicielles. Elle englobe l'ensemble des pratiques, conventions et standards mis en place pour créer un code clair, maintenable et évolutif. Dans le cadre de cet audit, nous avons utilisé plusieurs outils tels que PHPCS, PHPStan et SymfonyInsight pour évaluer la qualité du code de l'application. Ces outils ont permis de détecter des erreurs, des violations de normes de codage, ainsi que des opportunités d'amélioration.

Résultats avant migration :

- PHPCS (PSR12) :

```
Registering sniffs in the PSR1, PSR2, PSR12 standard... DONE (63 sniffs registered)
Creating file list... DONE (9 files in queue)
Changing into directory src/AppBundle/Controller
Processing SecurityController.php [PHP => 226 tokens in 42 lines]... DONE in 3ms (1 errors, 0 warnings)
Processing UserController.php [PHP => 536 tokens in 69 lines]... DONE in 6ms (1 errors, 1 warnings)
Processing TaskController.php [PHP => 660 tokens in 94 lines]... DONE in 8ms (1 errors, 1 warnings)
Processing DefaultController.php [PHP => 88 tokens in 17 lines]... DONE in 1ms (1 errors, 0 warnings)
Changing into directory src/AppBundle/Form
Processing TaskType.php [PHP => 145 tokens in 25 lines]... DONE in 2ms (1 errors, 0 warnings)
Processing UserType.php [PHP => 256 tokens in 28 lines]... DONE in 2ms (1 errors, 0 warnings)
Changing into directory src/AppBundle
Processing AppBundle.php [PHP => 34 tokens in 9 lines]... DONE in 0ms (1 errors, 0 warnings)
Changing into directory src/AppBundle/Entity
Processing Task.php [PHP => 444 tokens in 93 lines]... DONE in 5ms (1 errors, 0 warnings)
Processing User.php [PHP => 449 tokens in 90 lines]... DONE in 5ms (1 errors, 0 warnings)
```

```
-----
FOUND 1 ERROR AND 1 WARNING AFFECTING 2 LINES
-----
```

```
 1 | ERROR    | [x] End of line character is invalid; expected "\n" but found "\r\n"
18 | WARNING   | [ ] Line exceeds 120 characters; contains 131 characters
-----
```

- PHPSTAN (LEVEL 9) :

```
[ERROR] Found 58 errors
```

Le résultat de toutes les erreurs retournées par PHPStan sont disponibles sur ce [lien](#).

- SYMFONY INSIGHT :



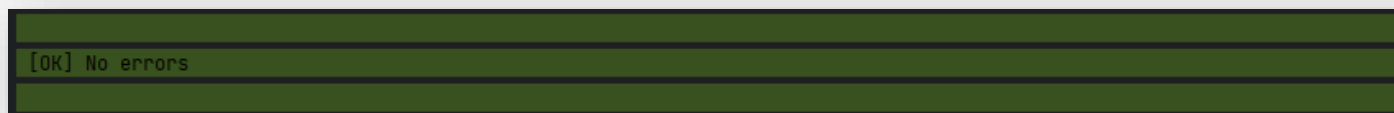
Parmi les bonnes pratiques adoptées, nous avons suivi les Standards de Codage PHP (PSR). Les PSR offrent des lignes directrices claires pour la structure du code, le nommage des classes, des méthodes et des variables, la gestion des espaces et des tabulations, ainsi que d'autres conventions essentielles pour un code cohérent et lisible.

Résultats après migration et correction des erreurs :

- PHPCS (PSR12) :

```
Registering sniffs in the standard... DONE (60 sniffs registered)
Creating file list... DONE (21 files in queue)
Loading cache... DONE (21 files in cache)
Changing into directory src/Controller
Processing SecurityController.php [loaded from cache]... DONE in 0ms (0 errors, 0 warnings)
Processing UserController.php [loaded from cache]... DONE in 0ms (0 errors, 0 warnings)
Processing TaskController.php [loaded from cache]... DONE in 0ms (0 errors, 0 warnings)
Processing DefaultController.php [loaded from cache]... DONE in 0ms (0 errors, 0 warnings)
Changing into directory src/Form
Processing TaskType.php [loaded from cache]... DONE in 0ms (0 errors, 0 warnings)
Processing UserType.php [loaded from cache]... DONE in 0ms (0 errors, 0 warnings)
Changing into directory src/Repository
Processing UserRepository.php [loaded from cache]... DONE in 0ms (0 errors, 0 warnings)
Processing TaskRepository.php [loaded from cache]... DONE in 0ms (0 errors, 0 warnings)
Changing into directory src/DataFixtures
Processing TaskFixtures.php [loaded from cache]... DONE in 0ms (0 errors, 0 warnings)
Processing UserFixtures.php [loaded from cache]... DONE in 0ms (0 errors, 0 warnings)
Changing into directory src/Entity
Processing Task.php [loaded from cache]... DONE in 0ms (0 errors, 0 warnings)
Processing User.php [loaded from cache]... DONE in 0ms (0 errors, 0 warnings)
Changing into directory src/Security/Voter
Processing TaskVoter.php [loaded from cache]... DONE in 0ms (0 errors, 0 warnings)
Changing into directory src/Security
Processing SecurityAuthenticator.php [loaded from cache]... DONE in 0ms (0 errors, 0 warnings)
Changing into directory src
Processing Kernel.php [loaded from cache]... DONE in 0ms (0 errors, 0 warnings)
Changing into directory tests/Controller
Processing UserControllerTest.php [loaded from cache]... DONE in 0ms (0 errors, 0 warnings)
Processing DefaultControllerTest.php [loaded from cache]... DONE in 0ms (0 errors, 0 warnings)
Processing SecurityControllerTest.php [loaded from cache]... DONE in 0ms (0 errors, 0 warnings)
Processing TaskControllerTest.php [loaded from cache]... DONE in 0ms (0 errors, 0 warnings)
Changing into directory tests
Processing bootstrap.php [loaded from cache]... DONE in 0ms (0 errors, 0 warnings)
Changing into directory tests/UtilsTests
Processing UtilsTests.php [loaded from cache]... DONE in 0ms (0 errors, 0 warnings)
```

- PHPSTAN (LEVEL 9) :



- SYMFONY INSIGHT :



En adoptant de telles pratiques, l'application devient plus maintenable, moins sujette aux erreurs et plus facilement extensible, tout en favorisant la collaboration entre les développeurs et en facilitant la compréhension du code.

IV. SUIVI DES PERFORMANCES DE L'APPLICATION

Un aspect critique pour tout projet est la gestion de ses performances. Les performances d'une application ont un impact direct sur l'expérience de l'utilisateur, la réactivité et l'efficacité globale. Dans cette optique, nous avons mené une évaluation approfondie des performances de l'application **ToDo & Co**, avec pour objectif d'identifier les zones potentielles d'amélioration et de garantir une expérience utilisateur optimale.

Pour mesurer avec précision les performances de l'application, nous avons utilisé le Profiler Symfony, un outil puissant intégré au framework. Le profiler Symfony a été choisi pour sa capacité à collecter les données détaillées sur l'exécution de l'application, allant des temps de chargement des pages aux requêtes SQL exécutées.

Afin d'obtenir des résultats représentatifs de l'utilisation réelle, nous avons pris des mesures de performance à la fois sur un serveur local et sur un serveur distant (VPS). Cette approche nous a permis de prendre en compte les variations potentielles de performance en fonction de l'environnement d'hébergement.

Une étape cruciale de l'évaluation a été la comparaison des mesures de performance entre la version 3.1 et la version 5.4 de l'application. Cette comparaison nous a donné un aperçu précieux de l'impact de la mise à jour sur les performances globales de l'application.

Résultats des mesures de performance (Serveur distant) :

Le cache a été libéré pour chaque prise de mesure. Les mesures pour la version 5.4 ont été prises après correction des anomalies et ajouts des nouvelles fonctionnalités.

Version 3.1

Version 5.4

1. Formulaire de connexion :

Performance metrics		
548 ms	363 ms	7.75 MB
Total execution time	Symfony initialization	Peak memory usage

Performance metrics		
158 ms	31 ms	4.00 MiB
Total execution time	Symfony initialization	Peak memory usage

2. Page d'accueil :

Performance metrics		
489 ms	319 ms	7.75 MB
Total execution time	Symfony initialization	Peak memory usage

Performance metrics		
127 ms	27 ms	4.00 MiB
Total execution time	Symfony initialization	Peak memory usage

3. Page de la liste des tâches :

Performance metrics		
1705 ms	1292 ms	7.75 MB
Total execution time	Symfony initialization	Peak memory usage

Performance metrics		
178 ms	32 ms	4.00 MiB
Total execution time	Symfony initialization	Peak memory usage

Query Metrics			
2	2	6.35 ms	0
Database Queries	Different statements	Query time	Invalid entities

Query Metrics			
2	2	3.18 ms	0
Database Queries	Different statements	Query time	Invalid entities

4. Formulaire de création d'une nouvelle tâche :

Performance metrics		
1158 ms	409 ms	12.25 MB
Total execution time	Symfony initialization	Peak memory usage

Performance metrics		
249 ms	29 ms	6.00 MiB
Total execution time	Symfony initialization	Peak memory usage

Version 3.1

5. Formulaire d'édition d'une tâche :

Performance metrics

1629 _{ms}	471 _{ms}	12.25 _{MB}
Total execution time	Symfony initialization	Peak memory usage

Query Metrics

2	2	3.83 _{ms}	0
Database Queries	Different statements	Query time	Invalid entities

6. Page de la liste des utilisateurs :

Performance metrics

575 _{ms}	359 _{ms}	7.75 _{MB}
Total execution time	Symfony initialization	Peak memory usage

Query Metrics

2	2	3.56 _{ms}	0
Database Queries	Different statements	Query time	Invalid entities

7. Formulaire de création d'un nouvel utilisateur :

Performance metrics

1062 _{ms}	314 _{ms}	12.50 _{MB}
Total execution time	Symfony initialization	Peak memory usage

8. Formulaire d'édition d'un utilisateur :

Performance metrics

1079 _{ms}	326 _{ms}	12.25 _{MB}
Total execution time	Symfony initialization	Peak memory usage

Version 5.4

Performance metrics

238 _{ms}	26 _{ms}	6.00 _{MiB}
Total execution time	Symfony initialization	Peak memory usage

Query Metrics

2	2	4.50 _{ms}	0
Database Queries	Different statements	Query time	Invalid entities

Performance metrics

145 _{ms}	31 _{ms}	4.00 _{MiB}
Total execution time	Symfony initialization	Peak memory usage

Query Metrics

2	2	3.35 _{ms}	0
Database Queries	Different statements	Query time	Invalid entities

Performance metrics

298 _{ms}	45 _{ms}	6.00 _{MiB}
Total execution time	Symfony initialization	Peak memory usage

Performance metrics

179 _{ms}	31 _{ms}	6.00 _{MiB}
Total execution time	Symfony initialization	Peak memory usage

Cette analyse comparative met en évidence les gains significatifs de performance obtenus avec la migration vers la version 5.4 et de la correction des erreurs sur la qualité du code. Les temps d'exécution réduits, les améliorations de l'initialisation Symfony et la réduction de la consommation de mémoire démontrent l'efficacité de la mise à jour dans l'optimisation globale de l'application.

Les résultats du détail de la séquence temporelle (Execution Timeline) pour chacune des requêtes sont disponibles sur le repository du projet ([Version 3.1](#), [Version 5.4](#)).

V. INTERFACE UTILISATEUR (UX) : INTERFACE ET DESIGN

L'Expérience Utilisateur (UX) occupe une place centrale dans le succès et la satisfaction des utilisateurs d'une application. Elle englobe tous les aspects de l'interaction entre utilisateur et l'application, y compris la conception de l'interface et le design visuel.

Dans le cadre de cet audit, nous avons pris en compte l'importance de l'UX en évaluant l'interface utilisateur existante. Cependant, il est important de noter que nous n'avons pas apporté de modifications significatives au design ou à la structure de l'interface pour les raisons suivantes :

1. Absence de besoins spécifiques dans le cahier des charges :

Le cahier des charges initial ne comportait pas de spécifications détaillées concernant les besoins en termes de design et de structure de l'interface. Afin d'éviter d'allonger la dette technique, nous avons choisi de ne pas entreprendre de modifications majeures sans des directives claires à ce sujet.

2. Prise de décision collective :

Etant donné que le design et la structure de l'interface peuvent avoir un impact significatif sur l'expérience des utilisateurs et sur la complexité du développement, il est préférable de prendre des décisions stratégiques de manière collective.

Nous recommandons d'organiser une réunion dédiée pour discuter des attentes spécifiques en matière de design et de structure, ainsi que pour déterminer le framework front-end mieux adapté aux objectifs de l'application.

3. Optimisation continue :

Plutôt que d'apporter des modifications qui pourraient potentiellement nécessiter des ajustements futurs, nous avons choisi d'attendre une direction claire concernant les évolutions du design et de la structure.

Cela permettra d'orienter les modifications de manière cohérente et de s'assurer qu'elles correspondent aux attentes et aux objectifs de l'application.

VI. CONCLUSION

Cet audit a été entrepris pour améliorer l'application ToDo & Co, construite sur Symfony 3.1, dans le contexte d'une startup en pleine expansion. L'objectif était d'assurer une meilleure qualité, une expérience utilisateur améliorée et des performances optimisées.

Notre démarche s'est articulée autour d'un processus bien défini : la mise en place d'un environnement de développement, la réalisation de tests, la mesure de la qualité du code, l'évaluation des performances, et enfin, la migration vers Symfony 5.4.

L'installation de l'environnement de développement a permis de démarrer l'application et de créer des jeux de données de test grâce aux fixtures. Les outils tels que PHPCS, PHPStan et Symfony Insight ont été utilisés pour évaluer la qualité du code, détecter et corriger les erreurs. Le suivi des performances de l'application s'est fait à travers le profiler Symfony, qui a fourni des métriques détaillées sur les performances, tant sur un serveur local que sur un serveur distant (VPS). Cette analyse a montré des améliorations significatives après la migration vers Symfony 5.4, avec des temps d'exécution et des consommations mémoire réduits.

L'expérience utilisateur a également été prise en compte, bien que le design n'ait pas été modifié dans cette itération. Cependant, l'importance du design et l'utilisation de frameworks front-end ont été soulignées comme des axes d'amélioration potentiels pour l'avenir.

Cet audit a démontré que la qualité du code, les performances et l'UX sont des aspects clés du développement d'une application. Des pistes d'amélioration ont été identifiées pour continuer à optimiser l'application ToDo & Co dans son évolution future.