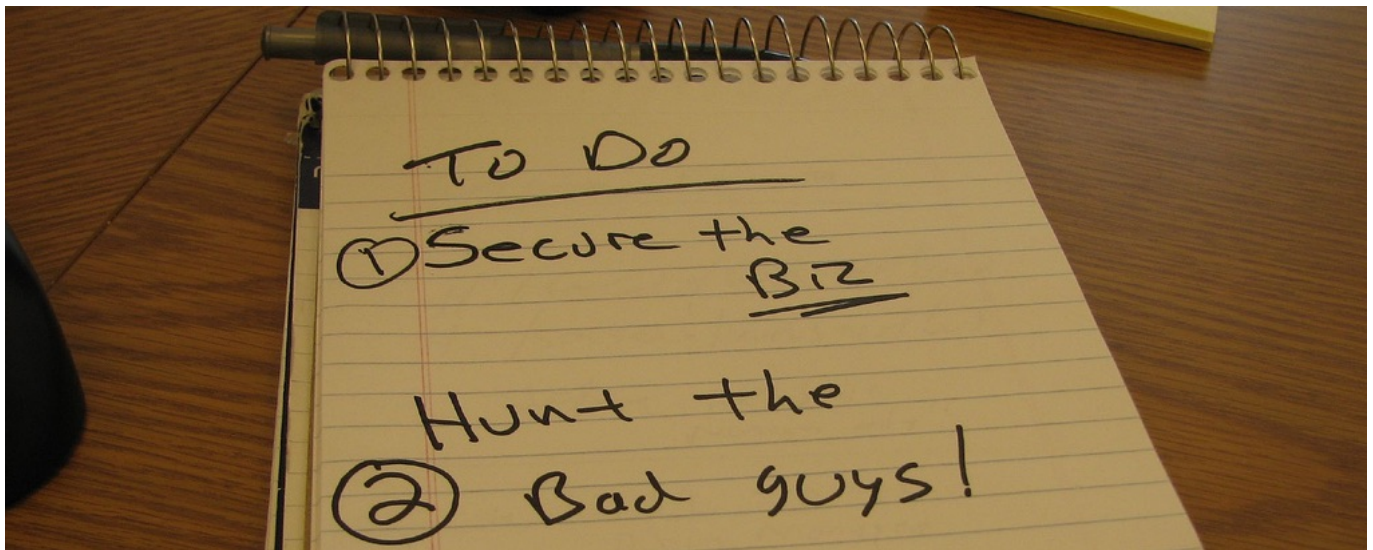


# ToDoList : Audit de performance et de qualité

---



Auteur : **David Cornacchia**

Dernière mise à jour : **15/11/2021**

# Sommaire

---

## 1. Audit technique

### 1.1 Dette technique

- Obsolescence des packages
- Version du Framework
- Version PHP

### 1.2 Analyse automatique

### 1.3 Review manuelle

- Anomalies
- Points améliorables

## 2. Audit de performances

### 2.1 Analyse des routes

### 2.2 Améliorations

- Dockerization
- Fixtures
- Tests automatisés
- Test coverage
- Corrections des anomalies
- Upgrade
- Composant de sécurité

## 3. Gains de performances

### 3.1 Analyse des routes

## 4. Suggestions d'améliorations

# Audit technique

## Dette technique

### Obsolèscence des packages

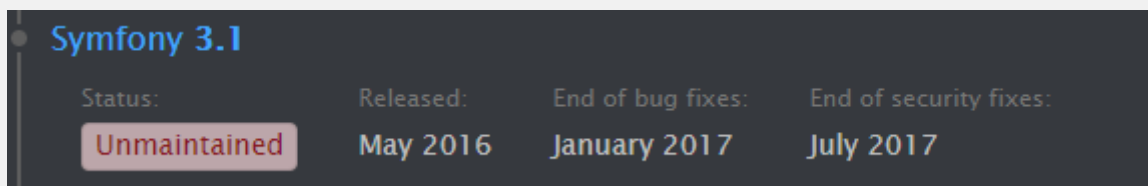
La commande `composer outdated` nous permet d'afficher la liste des paquets installés ayant des mises à jour disponibles.

```
dev@eda4ad5cad35:/var/www/html$ composer outdated
Color legend:
- patch or minor release available - update recommended
- major release available - update possible
doctrine/annotations      v1.2.7 1.13.2 Docblock Annotations Parser
doctrine/cache             v1.6.0 2.1.1  Caching library offering an object-oriented API for many cache backends
doctrine/collections       v1.3.0 1.6.8  Collections Abstraction library
doctrine/common            v2.6.1 3.2.0  Common Library for Doctrine projects
doctrine/dbal              v2.5.5 2.13.4 Database Abstraction Layer
doctrine/doctrine-bundle   1.6.4 2.4.3  Symfony DoctrineBundle
doctrine/doctrine-cache-bundle 1.3.0 1.4.0  Symfony Bundle for Doctrine Cache
Package doctrine/doctrine-cache-bundle is abandoned, you should avoid using it. No replacement was suggested.
doctrine/doctrine-fixtures-bundle v2.4.1 3.4.1  Symfony DoctrineFixturesBundle
doctrine/inflector         v1.1.0 1.4.4  Common String Manipulations with regard to casing and singular/plural rules.
doctrine/instantiator       1.0.5 1.4.0  A small, lightweight utility to instantiate objects in PHP without invoking their constructors
doctrine/lexer              v1.0.1 1.0.2  Base library for a lexer that can be used in Top-Down, Recursive Descent Parsers.
doctrine/orm                v2.5.5 2.10.2 Object-Relational-Mapper for PHP
fzaninotto/faker            v1.9.2 1.9.2  Faker is a PHP library that generates fake data for you.
Package fzaninotto/faker is abandoned, you should avoid using it. No replacement was suggested.
guzzlehttp/promises        1.4.1 1.5.1  Guzzle promises library
guzzlehttp/psr7            1.8.2 1.8.3  PSR-7 message implementation that also provides common utility methods
incenteev/composer-parameter-handler v2.1.2 2.1.4  Composer script handling your ignored parameter file
monolog/monolog            1.21.0 1.26.1 Sends your logs to files, sockets, inboxes, databases and various web services
nelmio/alice               v2.3.6 3.6.0  Expressive fixtures generator
paragonie/random_compat    v2.0.3 v9.99.100 PHP 5.x polyfill for random_bytes() and random_int() from PHP 7
phpunit/php-code-coverage  4.0.8 6.1.4  Library that provides collection, processing, and rendering functionality for PHP code coverage information.
phpunit/php-file-iterator  1.4.5 2.0.4  FilterIterator implementation that filters files based on a list of suffixes.
phpunit/php-timer          1.0.9 2.1.3  Utility class for timing
phpunit/php-token-stream   2.0.2 3.1.3  Wrapper around PHP's tokenizer extension.
Package phpunit/php-token-stream is abandoned, you should avoid using it. No replacement was suggested.
phpunit/phpunit            5.7.27 7.5.20 The PHP Unit Testing framework.
phpunit/phpunit-mock-objects 3.4.4 6.1.2  Mock Object library for PHPUnit
Package phpunit/phpunit-mock-objects is abandoned, you should avoid using it. No replacement was suggested.
psr/log                    1.0.2 1.1.4  Common interface for logging libraries
sebastian/comparator       1.2.4 3.0.3  Provides the functionality to compare PHP values for equality
sebastian/diff             1.4.3 3.0.3  Diff implementation
sebastian/environment      2.0.0 4.2.4  Provides functionality to handle HHVM/PHP environments
sebastian/exporter         2.0.0 3.1.3  Provides the functionality to export PHP variables for visualization
sebastian/global-state     1.1.1 2.0.0  Snapshotting of global state
sebastian/object-enumerator 2.0.1 3.0.4  Traverses array structures and object graphs to enumerate all referenced objects
sebastian/recursion-context 2.0.0 3.0.1  Provides functionality to recursively process PHP variables
sebastian/resource-operations 1.0.0 2.0.2  Provides a list of PHP built-in functions that operate on resources
sensio/distribution-bundle v5.0.13 v5.0.25 Base bundle for Symfony Distributions
Package sensio/distribution-bundle is abandoned, you should avoid using it. No replacement was suggested.
sensio/framework-extra-bundle v3.0.16 v5.5.7 This bundle provides a way to configure your controllers with annotations
sensio/generator-bundle    v3.0.11 v3.1.7 This bundle generates code for you
Package sensio/generator-bundle is abandoned, you should avoid using it. Use symfony/maker-bundle instead.
sensiolabs/security-checker v4.0.0 v6.0.3 A security checker for your composer.lock
Package sensiolabs/security-checker is abandoned, you should avoid using it. Use https://github.com/fabpot/local-php-security-checker instead.
swiftmailer/swiftmailer    v5.4.3 v6.3.0 Swiftmailer, free feature-rich PHP mailer
symfony/monolog-bundle     2.11.1 v3.7.0  Symfony MonologBundle
symfony/phpunit-bridge     v3.1.6 v5.3.10 Symfony PHPUnit Bridge
symfony/polyfill-apcu      v1.2.0 v1.23.0 Symfony polyfill backporting apcu.* functions to lower PHP versions
symfony/polyfill-intl-icu v1.2.0 v1.23.0 Symfony polyfill for intl's ICU-related data and classes
symfony/polyfill-mbstring v1.2.0 v1.23.1 Symfony polyfill for the Mbstring extension
symfony/polyfill-php56     v1.2.0 v1.20.0 Symfony polyfill backporting some PHP 5.6+ features to lower PHP versions
symfony/polyfill-php70     v1.2.0 v1.20.0 Symfony polyfill backporting some PHP 7.0+ features to lower PHP versions
symfony/polyfill-util      v1.2.0 v1.23.0 Symfony utilities for portability of PHP codes
symfony/swiftmailer-bundle v2.3.11 v3.5.2  Symfony SwiftmailerBundle
symfony/symfony            v3.1.6 v4.4.33 The Symfony PHP framework
twig/twig                  v1.27.0 v2.13.1 Twig, the flexible, fast, and secure template language for PHP
```

## Version du Framework

La version Symfony utilisée est la **3.1**

Comme indiqué dans la documentation cette version n'est plus maintenue depuis 2017.

A dark-themed banner for Symfony 3.1. It features the text 'Symfony 3.1' in a large, light blue font. Below it, there are four columns of information: 'Status:' with a red 'Unmaintained' badge, 'Released:' with 'May 2016', 'End of bug fixes:' with 'January 2017', and 'End of security fixes:' with 'July 2017'.

source : [calendrier de release Symfony](#)

---

## Version php

La version haute utilisable est la 7.1.33

C'est sur cette version qu'a été fait cet audit.


Branch	Date		Last Release	Notes
7.2	30 Nov 2020	11 months ago	<a href="#">7.2.34</a>	<a href="#">A guide is available for migrating from PHP 7.2 to 7.3.</a>
7.1	1 Dec 2019	1 year, 11 months ago	<a href="#">7.1.33</a>	<a href="#">A guide is available for migrating from PHP 7.1 to 7.2.</a>
7.0	10 Jan 2019	2 years, 9 months ago	<a href="#">7.0.33</a>	<a href="#">A guide is available for migrating from PHP 7.0 to 7.1.</a>
5.6	31 Dec 2018	2 years, 10 months ago	<a href="#">5.6.40</a>	<a href="#">A guide is available for migrating from PHP 5.6 to 7.0.</a>

source : [doc officielle php](#)

## Analyse automatique


L'analyse de qualité du code faite grace à l'outil [CodeClimate](#) n'a révélé que quelques anomalies non critiques, qui pourront être corrigées facilement.

### src/Controller/SecurityController.php

 MEDIUM | Unused Code | Avoid unused parameters such as '\$request'.


```
15 public function loginAction(Request $request, AuthenticationUtils $authenticationUtils)
```

### src/Form/TaskType.php

 MEDIUM | Unused Code | Avoid unused parameters such as '\$options'.

```
11 public function buildForm(FormBuilderInterface $builder, array $options)
```

### src/Form/UserType.php

 MEDIUM | Unused Code | Avoid unused parameters such as '\$options'.

```
15 public function buildForm(FormBuilderInterface $builder, array $options)
```

## Review manuelle

### Anomalies

- Le bouton "*Consulter la liste des tâches à faire*" renvoie vers la liste de **toutes** les taches.
- Le bouton "*Consulter la liste des taches terminées*" ne renvoie nulle part.
- Setter manquant pour l'attribut `isDone` de `Task::class`
- Fichiers manquant (bootstrap/jquery)  
Une erreur dans la console signalait des fichiers manquants
  - `web/js/jquery.js`
  - `web/css/bootstrap.min.css.map`

### Points améliorables

- Ajouter des contraintes de validation sur les entités et les formulaires ( [voir doc Validation](#) )
- Personnalisation des pages d'erreurs (*500, 404, etc...*)
- Utiliser l'injection de dépendances plutôt que les containers
- Spécification des verbes HTTP pour les routes

# Audit de performances

## Analyse des routes

L'analyse des performances de l'application a été effectuée avec l'outil [Blackfire](#).

Parmis les données récoltées voici un récap du temps d'exécution (*en millisecondes*) ainsi que la mémoire utilisée (*en mégabytes*) pour chaque route de l'application.

Route	Temps d'exécution	Memoire utilisée (MB)
<b>Home</b>		
Page d'accueil	86.8 ms	13.5 MB
<b>Login</b>		
Login ( <i>formulaire</i> )	66.2 ms	11.2 MB
Login ( <i>process</i> )	66.2 ms	13.3 MB
Déconnexion	79.2 ms	13.3 MB
<b>Users</b>		
Création d'un utilisateur ( <i>formulaire</i> )	95 ms	14.1 MB
Création d'un utilisateur ( <i>process</i> )	455 ms	16.7 MB
Edition d'un utilisateur ( <i>formulaire</i> )	118 ms	17 MB
Edition d'un utilisateur ( <i>process</i> )	448 ms	16.9 MB
<b>Tasks</b>		
Liste des tâches ( <i>toutes</i> )	86.1 ms	13.6 MB
Liste des tâches ( <i>à faire</i> )	87.3 ms	13.6 MB
Liste des tâches ( <i>terminées</i> )	86.1 ms	13.6 MB
Création d'une tâche ( <i>formulaire</i> )	107 ms	16.4 MB
Création d'une tâche ( <i>process</i> )	112 ms	16.3 MB
Edition d'une tâche ( <i>formulaire</i> )	109 ms	16.5 MB
Edition d'une tâche ( <i>process</i> )	120 ms	16.3 MB
Changement de statut d'une tâche ( <i>toggle</i> )	90.9 ms	13.8 MB
Suppression d'une tâche	87.7 ms	13.8 MB

# Améliorations

---

## Dockerization

L'intégration de Docker permet de facilement et rapidement configurer un environnement complet qui pourra être installé sur n'importe quelle machine. Cela permet entre autres, de fournir à tous les développeurs travaillant sur le même projet, un environnement identique (version et extensions php, version mysql, services annexes,etc...).

[Lien vers la pull request : Dockerize](#)

---

## Implémentation de tests automatisés

Les tests unitaires et fonctionnels permettent de s'assurer du fonctionnement des différents éléments de l'application. Les tests pouvant être lancés à tout moments ils permettent par exemple, pendant la phase de développement d'une nouvelle fonctionnalité, de s'assurer que l'implémentation ne provoque pas d'effets de bords sur le reste de l'application.

[Lien vers la pull request : Test Legacy](#)

---

## Ajout de Fixtures pour les tests et le développement

La création de fixtures ( *ou jeu de données* ) offre la possibilité de simuler le comportement de l'application en phase d'exploitation. Il facilite le développement et les tests des features qui seront chargés d'interagir avec la base de données.

[Lien vers la pull request : Fixtures](#)

---

## Ajout agent test coverage

L'analyse de la couverture de tests permet de s'assurer du taux de couverture des tests automatisés. Une bonne couverture garantie la détection des éventuels dysfonctionnements de l'application.

[Lien vers le rapport de test coverage](#)

*nb: certaines parties du code ont été volontairement exclues du coverage. (ex: routes non utilisées \_login\_check & logout)*

## Corrections des anomalies

- Boutons de navigations :
  - fix: des liens défectueux
  - fix: des routes manquantes
- Ajout de la méthode Task::isDone()
- Ajout des fichiers manquants :
  - `web/js/jquery.js`
  - `web/css/bootstrap.min.css.map`



# Upgrade

## Symfony

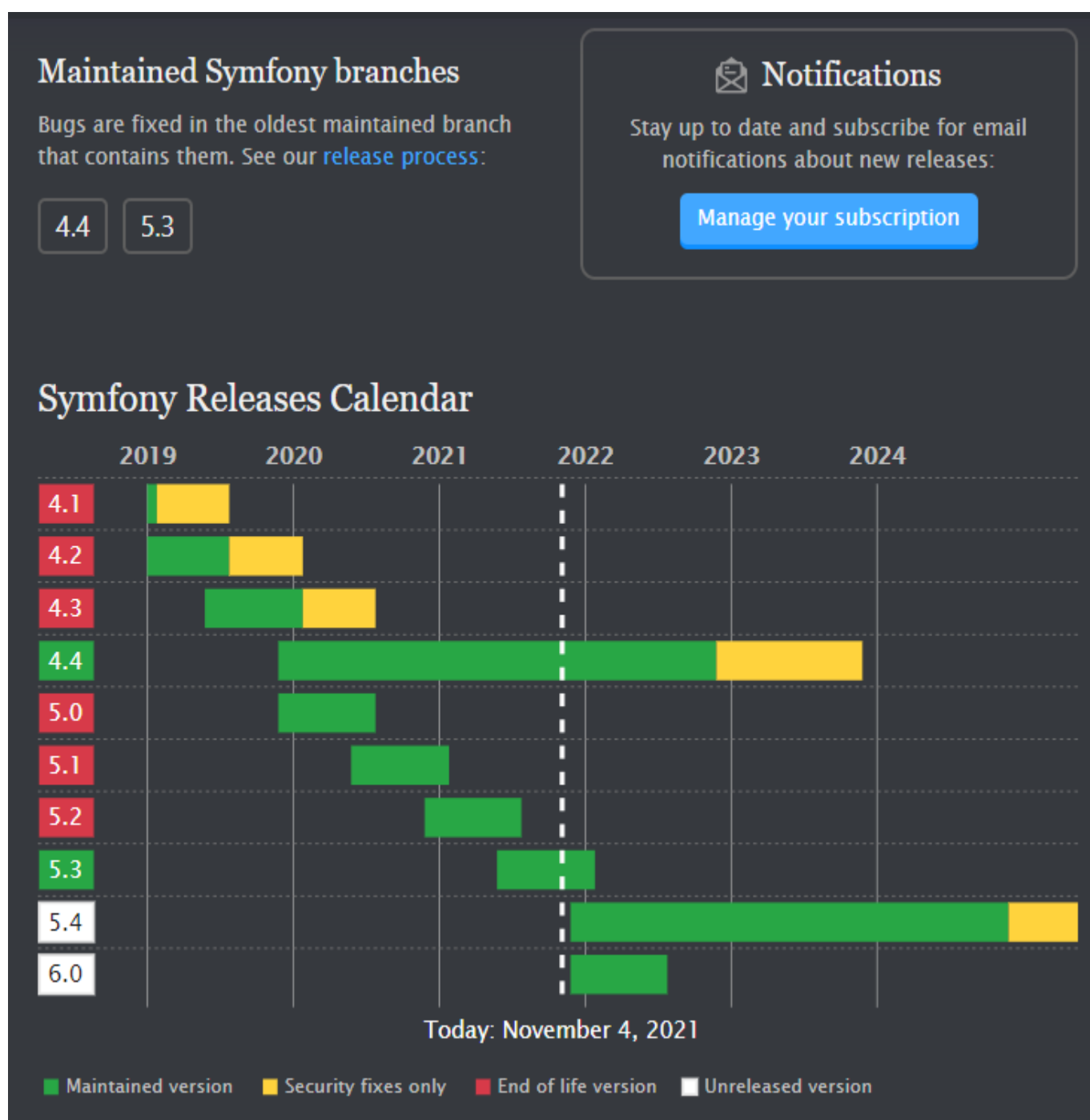
Migration vers la dernière version LTS de symfony.

Le passage à une version supérieure du framework serait l'une des principales solutions pour améliorer l'application. Toutefois, le choix de la version sur laquelle migrer est primordiale.

Le [calendrier de release Symfony](#) indique qu'en moyenne une version LTS (*Long-term support*) sort tous les 2 ans.

Par définition, ces versions sont celles bénéficiant de la période de maintenance la plus longue (*~3ans pour les bugs et ~3,5ans pour la sécurité*) .

A l'heure ou cet audit est effecuté la version LTS la plus récente est la version **4.4**.



## PHP

L'upgrade de version Symfony permet notamment d'utiliser des versions supérieurs de PHP.

Il est donc recommandé d'utiliser la version **7.4**.

Comme indiqué dans [la documentation officielle PHP](#), le langage a fait l'objet de nombreuses évolutions, ainsi que l'ajout de différentes **fonctionnalités** (*propriétés typées, fonctions flèches, déballage dans les tableaux, etc...*).

C'est donc cette version qui été utilisé pour le profiling de la branch **develop** du projet qui contient les améliorations de l'application.

## Dépendances

L'utilisation de la version 4.4 de PHP permet également d'utiliser de nombreux packages via composeur ainsi que des versions supérieures de la majorités des dépendances.

Voici une liste non-éxhaustives des packages concernés :

- sensio/framework-extra-bundle : 3.0 => 5.1
- phpunit/phpunit : 5.0 => 9.5
- nelmio/alice : 2.1 => 3.0
- symfony/profiler : 1.0 => 4.4
- symfony/security : *composant de sécurité*
- symfony/validator : *validation de création/édition d'entité*
- symfony/dotenv : *gestion de variables d'environnement*
- symfony/form : *formulaires*
- php-coveralls : *rapport de couverture de tests*

## Structure des fichiers

Afin de correspondre au fonctionnement de la version 4 de Symfony et en particulier **Symfony Flex**, la structure des fichiers doit être modifiée comme suis :

```
dossier-principale/
├── assets/
├── bin/
│   └── console
├── config/
│   ├── bundles.php
│   ├── packages/
│   ├── routes.yaml
│   └── services.yaml
├── public/
│   └── index.php
├── src/
│   ├── ...
│   └── Kernel.php
├── templates/
├── tests/
├── translations/
├── var/
└── vendor/
```

## Composant de sécurité

L'installation du package de sécurité permet une gestion simplifiée des accès aux différentes parties de l'application.

*plus d'informations sur le composant de sécurité [ici](#)*

## Bonnes pratiques

- Création de class **Repositories** récupéré par injections de dépendances.
- Utiliser la classe **EntityManager** au lieu de `ObjectManager`.
- Utilisation de **Listener** pour l'encodage du mot de passe utilisateur.
- Suppression du suffix "Action" dans les noms de methods des controller.
- Utilisation de **Voter** pour la gestion des accès aux différentes actions effectuelles sur les entités.
- Utilisation des déclarations de type (type-hint).

# Gains de performances

En comparant les performances de la branche principale avec la branche **develop** qui contient toutes les améliorations présentées ci-dessus, on constate une nette amélioration des performances.

Gain moyen	%
Temps d'exécution	65,6%
Utilisation mémoire	75,5%

Route	Temps d'exécution	Mémoire utilisée (MB)
<b>Home</b>		
Page d'accueil	28 ms <b>(-68%)</b>	2.99 MB <b>(-78%)</b>
<b>Login</b>		
Login ( <i>formulaire</i> )	18 ms <b>(-76%)</b>	2.64 MB <b>(-73%)</b>
Login ( <i>process</i> )	125 ms <b>(-78%)</b>	2.99 MB <b>(-70%)</b>
Déconnexion	26.5 ms <b>(-78%)</b>	2.88 MB <b>(-66%)</b>
<b>Users</b>		
Création d'un utilisateur ( <i>formulaire</i> )	43.3 ms <b>(-70%)</b>	4.27 MB <b>(-54%)</b>
Création d'un utilisateur ( <i>process</i> )	157 ms <b>(-73%)</b>	4.43 MB <b>(-65%)</b>
Edition d'un utilisateur ( <i>formulaire</i> )	44.4 ms <b>(-74%)</b>	4.38 MB <b>(-62%)</b>
Edition d'un utilisateur ( <i>process</i> )	53.3 ms <b>(-74%)</b>	4.46 MB <b>(-88%)</b>
<b>Tasks</b>		
Liste des tâches ( <i>toutes</i> )	30.7 ms <b>(-77%)</b>	3.11 MB <b>(-67%)</b>
Liste des tâches ( <i>à faire</i> )	30.6 ms <b>(-77%)</b>	3.13 MB <b>(-65%)</b>
Liste des tâches ( <i>terminées</i> )	31.3 ms <b>(-77%)</b>	3.13 MB <b>(-64%)</b>
Création d'une tâche ( <i>formulaire</i> )	38.6 ms <b>(-75%)</b>	4.03 MB <b>(-64%)</b>
Création d'une tâche ( <i>process</i> )	44 ms <b>(-75%)</b>	4.14 MB <b>(-61%)</b>
Edition d'une tâche ( <i>formulaire</i> )	40 ms <b>(-75%)</b>	4.04 MB <b>(-63%)</b>
Edition d'une tâche ( <i>process</i> )	43.5 ms <b>(-75%)</b>	4.13 MB <b>(-64%)</b>
Changement de statut d'une tâche ( <i>toggle</i> )	35.6 ms <b>(-76%)</b>	3.29 MB <b>(-61%)</b>
Suppression d'une tâche	35.3 ms <b>(-76%)</b>	3.38 MB <b>(-60%)</b>

# Suggestions d'améliorations

---

## Cache

La mise en place de mise en cache sur certaines pages permettrait de réduire d'avantage le chargement.

## Pagination

Dans l'hypothèse où le nombre de tâches et d'utilisateurs atteindraient un nombre important, un système de pagination permettrait d'améliorer le confort de navigation et également de limiter le nombre d'éléments chargés sur certaines pages.

## Navigation à facettes

La navigation à facettes permettrait de filtrer les tâches affichées, de manière intuitive (ex: période de création, auteur, etc...).

## Migration version php 8

La version 8 de PHP est relativement récente à l'heure actuelle mais il serait très probablement bénéfique, à terme, de migrer vers cette version afin de bénéficier de nouvelles fonctionnalités et évolutions liées au langage (ex : Attributs, Types d'union, etc... ).