

# Audit de code

## Qualité & performances



Version : 0.1

Date de la dernière mise à jour : 29/09/2021

# Sommaire

## Table des matières

<b>1. Audit de la 1<sup>ère</sup> version .....</b>	<b>2</b>
1.1 Qualité de code : CodeClimate et Codacy.....	2
1.1.1 CodeClimate.....	2
1.1.2 Codacy.....	2
2.2 Performances : Blackfire .....	3
<b>2. Version actuelle - Qualité de code.....</b>	<b>5</b>
1.1 Enjeux et objectifs .....	5
1.2 Analyse : Codacy.....	5
1.3 Analyse : CodeClimate.....	7
1.4 Recommandations .....	7
<b>3. Version actuelle - Performances.....</b>	<b>9</b>
2.1 Enjeux et objectifs .....	9
2.2 Analyse : Blackfire .....	9
2.3 Mesures déjà mises en place .....	11
2.4 Recommandations .....	12
<b>Conclusion .....</b>	<b>13</b>

## 1. Audit de la 1<sup>ère</sup> version

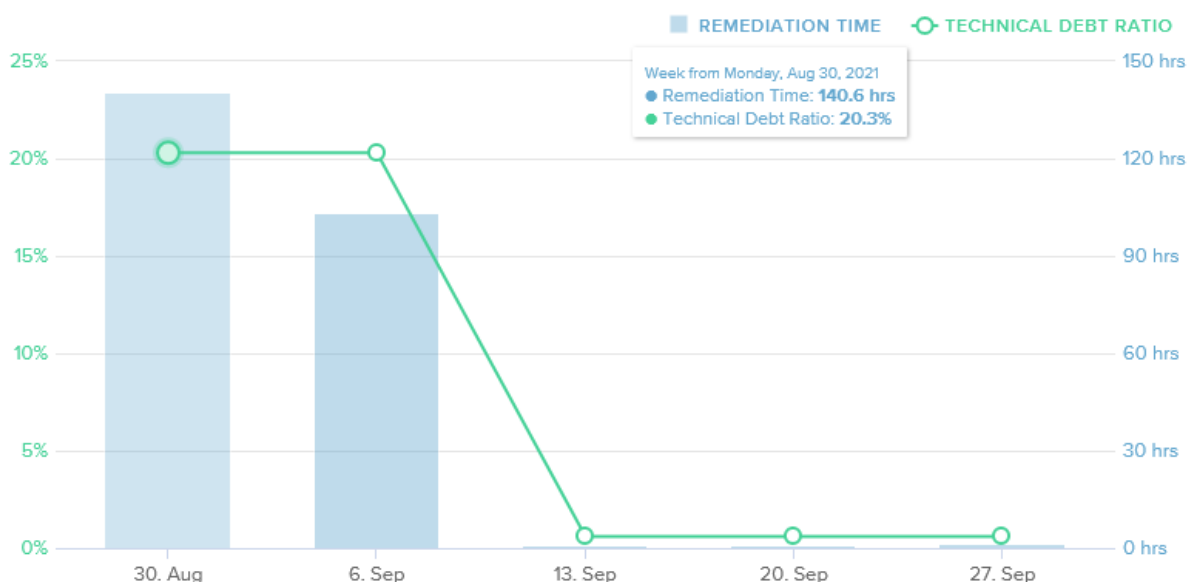
### 1.1 Qualité de code : CodeClimate et Codacy

#### 1.1.1 CodeClimate

Le graphique ci-dessous provient de CodeClimate, il s'agit de l'évolution de la dette technique depuis le 1<sup>er</sup> commit jusqu'à la version finale :

[https://codeclimate.com/github/Drx85/p8\\_todo\\_and\\_co/trends/technical\\_debt](https://codeclimate.com/github/Drx85/p8_todo_and_co/trends/technical_debt)

### Technical Debt

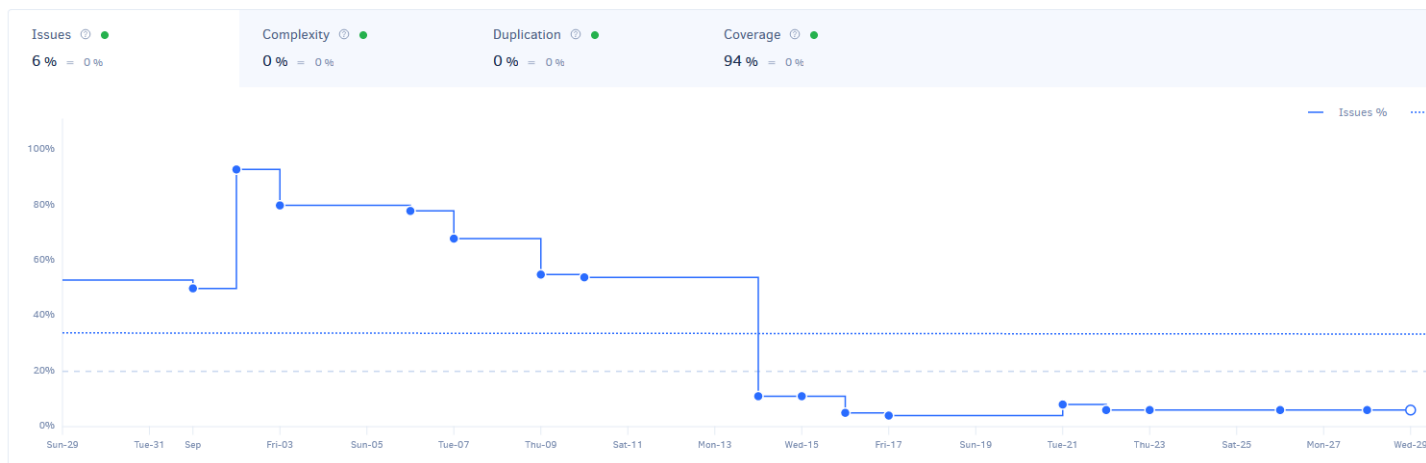


On constate que la V1 contient 20% de son code à revoir, estimé à 140 heures de travail. Il s'agit exclusivement de problème venant de Bootstrap installé en local. Pour y remédier, nous avons prévu dans la version finale de supprimer cette version locale en chargeant Bootstrap via un Content Delivery Network (CDN), et d'upgrader la version actuelle à Bootstrap 5.

Comme CodeClimate ne relève pas les mêmes erreurs que Codacy, voyons l'analyse de ce dernier.

#### 1.1.2 Codacy

Le graphique page suivante provient de Codacy, comme CodeClimate, il présente l'évolution des remarques ou problèmes techniques depuis le 1<sup>er</sup> commit jusqu'à la version finale :



Le graphique est visible de manière plus claire ici :

[https://app.codacy.com/gh/Drx85/p8\\_todo\\_and\\_co/dashboard?branch=main](https://app.codacy.com/gh/Drx85/p8_todo_and_co/dashboard?branch=main)

On constate qu'à l'installation, le projet contient 76 issues, qui affectent 50% de son nombre total de lignes. Elles concernent principalement des problèmes liés à la version 3.1 de Symfony et s'effaceront au fur et à mesure des upgrades, comme nous le lisons sur le graphique.

## 2.2 Performances : Blackfire

Le tableau ci-dessous (page suivante) regroupe l'analyse de chaque route, en local et en environnement de production.

Les données de synthèse suivantes ont été extraites de ce tableaux :

- ✕ Temps de chargement moyen : 520 ms
- ✕ Temps de chargement maximum : 607 ms (tasks/{id}/edit)
- ✕ Nombre maximum de requêtes SQL : 2
- ✕ Pic de mémoire moyen : 16,53 MB
- ✕ Pic de mémoire maximum : 19,4 MB (tasks/{id}/edit) et 19,3 MB (tasks/create)

Lien vers chaque analyse :

Route	Lien
index	<a href="https://blackfire.io/profiles/8e584014-912a-46a8-b179-91f66170f591/graph">https://blackfire.io/profiles/8e584014-912a-46a8-b179-91f66170f591/graph</a>
/login	<a href="https://blackfire.io/profiles/160751cd-1d2a-4d39-8761-d59855957a78/graph">https://blackfire.io/profiles/160751cd-1d2a-4d39-8761-d59855957a78/graph</a>
/users/create	<a href="https://blackfire.io/profiles/6cc82412-ffe1-47ea-a274-d9047a97be65/graph">https://blackfire.io/profiles/6cc82412-ffe1-47ea-a274-d9047a97be65/graph</a>
/tasks	<a href="https://blackfire.io/profiles/06a72af1-f54e-440a-a9c5-07447c7ca6ef/graph">https://blackfire.io/profiles/06a72af1-f54e-440a-a9c5-07447c7ca6ef/graph</a>
/tasks/create	<a href="https://blackfire.io/profiles/5b61f514-9daa-49e5-b3f3-64668f0f98fc/graph">https://blackfire.io/profiles/5b61f514-9daa-49e5-b3f3-64668f0f98fc/graph</a>
/tasks/{id}/edit	<a href="https://blackfire.io/profiles/c19b3652-28d4-4b72-9882-7f1e578ec0d9/graph">https://blackfire.io/profiles/c19b3652-28d4-4b72-9882-7f1e578ec0d9/graph</a>

## 200 GET https://localhost:8000/tasks/497/edit

### V1 - Edit task

Created 4 minutes ago by [ced.dep.85@hotmail.fr](#)

   0 rq 607 ms 334 ms 273 ms 19.4 MB 1.37 kB 0 µs / 0 rq 756 µs / 2 rq  Timeline

## 200 GET https://localhost:8000/tasks

### V1 - All tasks

Created 5 minutes ago by [ced.dep.85@hotmail.fr](#)

   0 rq 512 ms 281 ms 231 ms 15.7 MB 13.7 kB 0 µs / 0 rq 762 µs / 2 rq  Timeline

## 200 GET https://localhost:8000/tasks/create

### V1 - Create task

Created 5 minutes ago by [ced.dep.85@hotmail.fr](#)

   0 rq 589 ms 317 ms 272 ms 19.3 MB 628 B 0 µs / 0 rq 337 µs / 1 rq  Timeline

## 200 GET https://localhost:8000/users/create

### V1 - Create user

Created 5 minutes ago by [ced.dep.85@hotmail.fr](#)

   0 rq 578 ms 314 ms 264 ms 19 MB 628 B 0 µs / 0 rq 347 µs / 1 rq  Timeline

## 200 GET https://localhost:8000/

### V1 - Home

Created 6 minutes ago by [ced.dep.85@hotmail.fr](#)

   0 rq 494 ms 279 ms 215 ms 15.5 MB 628 B 0 µs / 0 rq 333 µs / 1 rq  Timeline

## 200 GET https://localhost:8000/login

### V1 - Login

Created 6 minutes ago by [ced.dep.85@hotmail.fr](#)

   0 rq 341 ms 203 ms 139 ms 10.3 MB 0 B 0 µs / 0 rq 0 µs / 0 rq  Timeline

## 2. Version actuelle - Qualité de code

### 1.1 Enjeux et objectifs

Il est nécessaire de respecter les codes et bonnes pratiques en vigueur lors de la réalisation de toute application pour maintenir une qualité de code la plus élevée possible. Cette qualité doit être surveillée à chaque modification du code de l'application. En effet, il s'agit d'un gage de maintenabilité, de sécurité, et de qualité perçue par tout collaborateur qui serait amené à modifier/agrémenter le code déjà existant, et même à tout collaborateur qui serait amené à relire le code.

Dans le cadre de ToDo & Co, chaque commit sera monitoré par Codacy et CodeClimate, et chaque issue remontée devra être analysée par l'équipe afin de trouver une solution adaptée.


### 1.2 Analyse : Codacy

#### Suivi des analyses :


[https://app.codacy.com/gh/Drx85/p8\\_todo\\_and\\_co/dashboard?branch=main](https://app.codacy.com/gh/Drx85/p8_todo_and_co/dashboard?branch=main)

#### Issues en cours :


##### src/Form/TaskType.php

 MEDIUM	Unused Code	Avoid unused parameters such as '\$options'.
11	<code>public function buildForm(FormBuilderInterface \$builder, array \$options)</code>	

##### src/Form/UserType.php

 MEDIUM	Unused Code	Avoid unused parameters such as '\$options'.
14	<code>public function buildForm(FormBuilderInterface \$builder, array \$options)</code>	

##### src/Kernel.php

 MINOR	Code Style	The method configureContainer uses an else expression. Else clauses are basically not necessary
22	<code>} else {</code>	

#### src/Security/LoginFormAuthenticator.php

	MEDIUM	Unused Code	Avoid unused parameters such as '\$token'.
67	public function onAuthenticationSuccess(Request \$request, TokenInterface \$token, string \$providerKey): ?Response		
	MEDIUM	Unused Code	Avoid unused parameters such as '\$request'.
76	public function onAuthenticationFailure(Request \$request, AuthenticationException \$exception): ?Response		

#### tests/Controller/DefaultControllerTest.php

	MEDIUM	Security	The use of function dirname() is discouraged
24	\$this->databaseTool->loadAliceFixture([dirname(__DIR__) . '/Fixtures/user.yaml']);		

#### tests/Controller/SecurityControllerTest.php

	MEDIUM	Security	The use of function dirname() is discouraged
54	\$this->databaseTool->loadAliceFixture([dirname(__DIR__) . '/Fixtures/user.yaml']);		

#### tests/Controller/TaskControllerTest.php

	MEDIUM	Security	The use of function dirname() is discouraged
73	\$this->databaseTool->loadAliceFixture([dirname(__DIR__) . '/Fixtures/user.yaml']);		
	MEDIUM	Security	The use of function dirname() is discouraged
166	\$this->databaseTool->loadAliceFixture([dirname(__DIR__) . '/Fixtures/task_user.yaml']);		

#### tests/Controller/UserControllerTest.php

	MEDIUM	Security	The use of function dirname() is discouraged
105	\$this->databaseTool->loadAliceFixture([dirname(__DIR__) . '/Fixtures/user.yaml']);		

#### tests/Entity/UserTest.php

	MEDIUM	Code Style	The class UserTest has 12 non-getter- and setter-methods. Consider refactoring UserTest to keep number of methods under 10.
11	class UserTest extends KernelTestCase		
	MEDIUM	Security	The use of function dirname() is discouraged
88	\$this->databaseTool->loadAliceFixture([dirname(__DIR__) . '/Fixtures/user.yaml']);		

On constate des issues à 3 niveaux :

- × Formulaires (code inutilisé, moyenne)
- × Fichiers générés par Symfony (sécurité, moyenne et bonnes pratiques, mineure)
- × Tests (sécurité et bonnes pratiques, moyenne)

Types de problème :

- × Paramètre de fonction inutilisé

- × Utilisation possiblement inutile d'un else
- × Fonction `dirname()` déconseillée
- × Nombre de méthodes dans une classe supérieur à 10

### 1.3 Analyse : CodeClimate

#### Suivi des analyses :

[https://codeclimate.com/github/Drx85/p8\\_todo\\_and\\_co](https://codeclimate.com/github/Drx85/p8_todo_and_co)

#### Issue en cours :

## Showing 1 of 1 total issue

Method `buildForm` has 29 lines of code (exceeds 25 allowed). Consider refactoring. OPEN

```

23     public function buildForm(FormBuilderInterface $builder, array $options)
24     {
25         $this->askPassword = $options['askPassword'];
26         $builder->add('username', TextType::class, ['label' => "Nom d'utilisateur"]);
27         if ($this->askPassword === true) {

```

Found in `src/Form/UserType.php` - About 1 hr to fix

### 1.4 Recommandations

Le projet ne comporte pas énormément d'issues, toutefois, il y a 6 failles de sécurité dans les tests, liées à la fonction `dirname()` qui est déconseillée. Il s'agit par conséquent d'un problème à résoudre en priorité.

- × L'expression `(dirname(__DIR__) . 'file.ext')` peut être remplacée par `__DIR__ . '/../file.ext'`.

Toujours dans les tests, la classe qui se charge de tester l'entité User contient de nombreuses méthodes, ce qui ne rentre pas dans le cadre des bonnes pratiques.

- × Un effort de refactorisation peut être fait au niveau de la classe `App\Tests\Entity\UserTest` pour alléger le nombre de méthodes : les fonctions `setUp()` et `assertHasErrors()` communes aux classes de test des entités pourraient par exemple être contenue dans une classe parente.



Dans les fonctions de création des formulaires, Codacy relève que le paramètre `$options` est inutilisé, sauf qu'il s'agit d'une fonction implémentée et qu'elle doit donc être appelée avec tous ses paramètres. On retrouve un cas similaire dans la classe `App\Security\LoginFormAuthenticator`

- × Passer les issues « *Paramètre de fonction inutilisé* » en faux-positif lorsque qu'il s'agit de fonctions implémentées.

CodeClimate relève la fonction `buildForm` comme trop longue dans `App\Form\UserType`.

- × Il pourrait être judicieux de refactoriser `isGranted()`, en la passant par le `OptionsResolver` par exemple, et d'externaliser le contenu de `addModelTransformer()` dans une fonction privée.

### 3. Version actuelle - Performances

#### 2.1 Enjeux et objectifs

Les performances de l'application, notamment déterminées par les choix de l'équipe de développement, se répercutent directement sur l'expérience utilisateur. Les standards en termes de temps de chargement sont de plus en plus élevés, et il est nécessaire de garder à l'esprit que les connexions à une applications peuvent se faire depuis des endroits où le réseau n'est pas bon ou médiocre. De plus, si le site voit un jour un fort trafic, de nombreuses requêtes par utilisateur pourraient le ralentir de manière significative.

Notre projet doit donc être le plus léger et optimisé possible. Il est suivi par le puissant outil Blackfire, qui a l'avantage d'offrir des analyses plus précises et poussées que l'outil Profiler de base de Symfony.

#### 2.2 Analyse : Blackfire

Le tableau ci-dessous (page suivante) regroupe l'analyse de chaque route, en local et en environnement de production. Cette disposition permet d'auditer de manière la plus précise possible : elle n'est pas dépendante d'un éventuel hébergeur qui viendrait ajouter des couches inutiles car nous n'avons de toute façon pas la main mise dessus, mais sans souffrir du chargement des composants de développement comme le débogueur ou le profiler.

L'analyse a été effectuée après avoir mis les données en cache.

Les données de synthèse suivantes ont été extraites de ce tableau :

- × Temps de chargement moyen : 195 ms (520 ms V1)
- × Temps de chargement maximum : 229 ms (users/{id}/edit) (607 ms V1)
- × Nombre maximum de requêtes SQL : 2 (2 V1)
- × Pic de mémoire moyen : 17,73 MB (16,53 MB V1)
- × Pic de mémoire max : 20,30 MB (tasks/{id}/edit) et 20,20 MB (tasks/create) (19,4 V1)

La diff. de mémoire s'explique par la version plus récente de Symfony qui est plus complexe.

**Lien vers chaque analyse :**

Route	Lien
index	<a href="https://blackfire.io/profiles/12eef1de-4f6d-4000-8b3c-76340cbf38bd/graph">https://blackfire.io/profiles/12eef1de-4f6d-4000-8b3c-76340cbf38bd/graph</a>
/login	<a href="https://blackfire.io/profiles/32fcf364-5b73-44fa-9b9d-8292acae8d82/graph">https://blackfire.io/profiles/32fcf364-5b73-44fa-9b9d-8292acae8d82/graph</a>
/users	<a href="https://blackfire.io/profiles/03160fe0-78d9-4f72-871a-48138bb2da5c/graph">https://blackfire.io/profiles/03160fe0-78d9-4f72-871a-48138bb2da5c/graph</a>
/users/create	<a href="https://blackfire.io/profiles/fb7959c5-cc56-47f0-9a9c-bf34cb2c9a32/graph">https://blackfire.io/profiles/fb7959c5-cc56-47f0-9a9c-bf34cb2c9a32/graph</a>
/tasks	<a href="https://blackfire.io/profiles/d6d87a27-781c-4bfc-8780-4ec703080906/graph">https://blackfire.io/profiles/d6d87a27-781c-4bfc-8780-4ec703080906/graph</a>
/tasks/create	<a href="https://blackfire.io/profiles/1bbc7f7a-49ae-404a-91a0-019153e6a965/graph">https://blackfire.io/profiles/1bbc7f7a-49ae-404a-91a0-019153e6a965/graph</a>
/users/{id}/edit	<a href="https://blackfire.io/profiles/23c8eb04-cd88-4ea3-a9fc-ece52b7da3d8/graph">https://blackfire.io/profiles/23c8eb04-cd88-4ea3-a9fc-ece52b7da3d8/graph</a>
/tasks/{id}/edit	<a href="https://blackfire.io/profiles/9a4caba7-3a05-4c27-b4ac-7fdb92ec04b5/graph">https://blackfire.io/profiles/9a4caba7-3a05-4c27-b4ac-7fdb92ec04b5/graph</a>
/tasks-finished	<a href="https://blackfire.io/profiles/0129d535-3026-44f9-b662-a2daad2fed57/graph">https://blackfire.io/profiles/0129d535-3026-44f9-b662-a2daad2fed57/graph</a>
/tasks-not-finished	<a href="https://blackfire.io/profiles/3183f28a-1800-4787-bd8f-036757e0a040/graph">https://blackfire.io/profiles/3183f28a-1800-4787-bd8f-036757e0a040/graph</a>

## 200 GET https://localhost:8000/users/364/edit

### Edit member

Created 30 minutes ago by [ced.dep.85@hotmail.fr](#)

  | 0 rq 229 ms 163 ms 66.6 ms 20.8 MB 1.26 kB 0 µs / 0 rq 635 µs / 2 rq Timeline

## 200 GET https://localhost:8000/tasks/510/edit

### Edit task

Created 31 minutes ago by [ced.dep.85@hotmail.fr](#)

  | 0 rq 215 ms 160 ms 54.8 ms 20.3 MB 1.63 kB 0 µs / 0 rq 680 µs / 2 rq Timeline

## 200 GET https://localhost:8000/tasks

### All tasks

Created 32 minutes ago by [ced.dep.85@hotmail.fr](#)

  | 0 rq 204 ms 149 ms 54.8 ms 16.5 MB 7.51 kB 0 µs / 0 rq 3.74 ms / 2 rq Timeline

## 200 GET https://localhost:8000/tasks-finished

### Tasks finished

Created 32 minutes ago by [ced.dep.85@hotmail.fr](#)

  | 0 rq 190 ms 137 ms 52.4 ms 18 MB 5.15 kB 0 µs / 0 rq 647 µs / 2 rq Timeline

## 200 GET https://localhost:8000/tasks-not-finished

### Tasks to do

Created 33 minutes ago by [ced.dep.85@hotmail.fr](#)

  | 0 rq 191 ms 137 ms 53.8 ms 18 MB 3.79 kB 0 µs / 0 rq 750 µs / 2 rq Timeline

## 200 GET https://localhost:8000/users

### Members list

Created 33 minutes ago by [ced.dep.85@hotmail.fr](#)

  | 0 rq 180 ms 137 ms 42.9 ms 16.2 MB 2.35 kB 0 µs / 0 rq 684 µs / 2 rq Timeline

## 200 GET https://localhost:8000/tasks/create

### Create task

Created 34 minutes ago by [ced.dep.85@hotmail.fr](#)

  | 0 rq 219 ms 164 ms 55.2 ms 20.2 MB 735 B 0 µs / 0 rq 374 µs / 1 rq Timeline

## 200 GET https://localhost:8000/

### Home

Created 36 minutes ago by [ced.dep.85@hotmail.fr](#)

  | 0 rq 181 ms 132 ms 49 ms 15.9 MB 735 B 0 µs / 0 rq 358 µs / 1 rq Timeline

## 200 GET https://localhost:8000/users/create

### Create user

Created 37 minutes ago by [ced.dep.85@hotmail.fr](#)

  | 0 rq 189 ms 131 ms 57.8 ms 17.9 MB 0 B 0 µs / 0 rq 0 µs / 0 rq Timeline

## 200 GET https://localhost:8000/login

### Login

Created 38 minutes ago by [ced.dep.85@hotmail.fr](#)

  | 0 rq 148 ms 103 ms 44.2 ms 13.5 MB 0 B 0 µs / 0 rq 0 µs / 0 rq Timeline

## 2.3 Mesures déjà mises en place

### 1. Cache applicatif :

Dans la version actuelle, les pages qui listent les tâches contiennent un cache Twig qui permet d'éviter d'effectuer des requêtes inutiles car identiques à chaque chargement. Un champ "modifié le" a été ajouté à l'entité Task et ajouté dans l'id du cache pour invalider ce dernier en cas de modification d'une tâche. L'id de l'utilisateur si connecté est également présent dedans, pour invalider le cache si l'utilisateur se connecte / déconnecte / change de compte, car le contenu mis en cache contient une condition dépendant d'un accès de permission en fonction de l'utilisateur.

La page qui contient la liste des membres n'a pas été mise en cache car elle ne requête pas une autre table, contrairement aux pages de tâches. Une seule requête est donc effectuée, qu'il est inutile de mettre en cache car la requête persisterait de toute façon à chaque chargement pour vérifier si le contenu a été modifié.

### 2. Problème n+1 :

Le système de cache a ses limites, car il n'est pas utilisé lors du 1<sup>er</sup> chargement d'une page ou lorsqu'il est invalidé (modification d'un élément contenu à l'intérieur). C'est pour cela qu'il est indispensable de ne pas se reposer complètement dessus, et effectuer des tests de performances sans l'activer. Le lien suivant montre l'analyse Blackfire de la page de toutes les tâches sans cache et sans résolution du problème : <https://blackfire.io/profiles/e9876583-19a1-492a-a33a-c41667fae0e2/graph>

On peut voir que 11 requêtes SQL sont effectuées, ce qui traduit un problème n+1 : chaque tâche qui est liée à un utilisateur n'ayant pas encore été chargé induit une requête SQL pour récupérer le nom de l'utilisateur. Il est dangereux de ne pas traiter ce problème car le nombre de requête SQL augmentera au fur et à mesure que le nombre de nouveaux utilisateurs liés à une tâche croît.

Le problème n+1 a déjà été traité dans le code. Il effectue une jointure avec les utilisateurs lors de la récupération des tâches, ce qui permet de tout regrouper en une seule requête. Comme on cherche à afficher les tâches et qu'une tâche ne peut avoir un seul utilisateur (relation ManyToOne), cette solution est adaptée car elle ne dupliquera pas les récupérations comme dans le cas d'une relation OneToMany.

On constate que, toujours sans cache, le nombre de requêtes retombe à 2 : <https://blackfire.io/profiles/983ea1ec-35fc-4b70-851c-ac74d006d33c/graph>

Dans ce cas, le cache applicatif devient moins pertinent. Il est toutefois utile de le conserver car il permettra de résoudre d'éventuelles failles de performances en cas d'évolution de l'application (par exemple si l'on rajoute des images pour les tâches).

## 2.4 Recommandations

Il est toujours possible d'aller plus loin dans l'optimisation de performances. Voici les recommandations à effectuer pour cette application, classées par ordre décroissant d'importance et d'impact :

En production, il est judicieux de ne pas s'arrêter qu'à un seul type de cache. Nous avons déjà notre cache applicatif avec Twig. Afin d'optimiser encore plus les requêtes SQL, nous pouvons utiliser également un cache côté Doctrine.

- × Nous pouvons par exemple mettre en place APCu de manière simple et rapide en suivant ce tutoriel SymfonyCasts : <https://symfonycasts.com/screencast/symfony-fundamentals/cache-config>

Actuellement, Symfony analyse les variables d'environnement déclarées dans le fichier .env à chaque requête HTTP. Cela utilise des ressources (quelques millisecondes) et pourrait être évité en production.

- × Pour pallier à ce problème, il est conseillé de définir de réelles variables d'environnement au sein du serveur. Si APP\_ENV est défini, Symfony ne se préoccupera plus du fichier .env

## Conclusion

La qualité de code et la performance de l'application sont deux axes à prendre en compte pendant le développement d'un service web. Nous avons analysé ce premier axe via deux outils différents, qui nous révèlent peu d'issues, et nous avons vu quels correctifs apporter pour y pallier. Concernant les performances, nous avons constaté grâce à Blackfire que nous obtenons des résultats corrects, qu'il est toujours possible d'améliorer grâce aux correctifs proposés, et qu'il sera nécessaire de suivre au cours de la croissance de l'application ainsi que de son nombre d'utilisateurs.