

# AUDIT

---

*Expertise professionnelle du contrôle de la qualité et de la performance de l'application ToDo & Co et améliorations apportées lors de la réalisation du project*

---



## Sommaire :

### **Partie 1 : Analyse de la Performance ..... 3**

Analyse de performance avec Blackfire .....	4
Conclusion de l'analyse Blackfire .....	13
Recapitulatif de l'analyse Blackfire .....	14
Depreciations .....	15
Analyse de performance avec LightHouse .....	16
Conclusion .....	21

### **Partie 2 : Analyse de la Qualité ..... 22**

Analyse avec les outils analytiques .....	23
Analyse avec les outils relatifs au rendu html .....	28
Analyse visuelle du developpeur .....	32
Conclusion .....	35

### **Partie 3 : Ameliorations apportées ..... 36**

Comparatif de l'analyse Blackfire .....	37
Resumé comparatif de l'analyse Blackfire .....	53
Conclusion de la comparaison .....	53
Tests et test-coverage avec PHPUnit .....	54
Respect des principes SOLID .....	55
Conclusion .....	56

# Partie 1 : Analyse de la performance

---

## Introduction :

L'analyse de Performance se fera de deux points de vue. Le premier point de vue est l'analyse Back-End, le second point de vue est l'analyse Front-End. Afin de réaliser ces analyses nous avons selectionner deux outils : Blackfire pour le Back-End et Google LightHouse pour le Front-End. Ce sont deux outils très reputés dans leurs domaine. Malgré tout le choix de ces outils a été fait arbitrairement par l'auteur de l'audit.

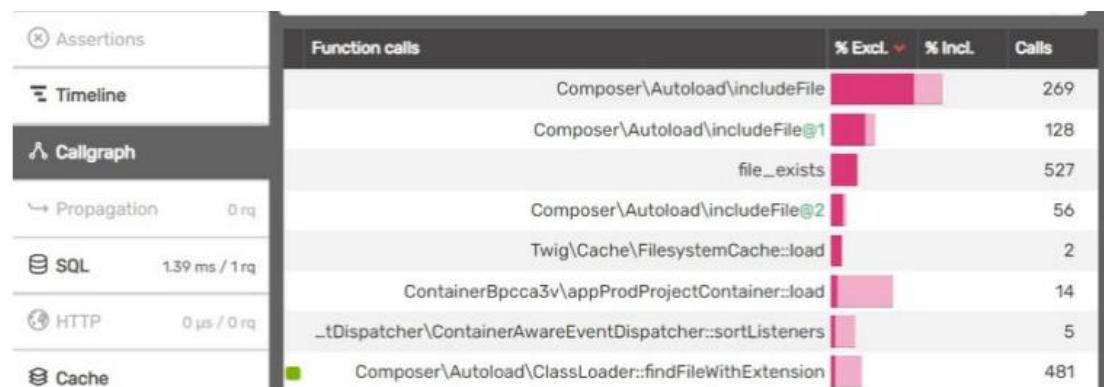
Il s'agit d'un audit détaillé, dans le cas de Blackfire, chaque requête a été profilé et figure dans cet audit, dans le cas de Google LightHouse, l'analyse a profilé le site en une seule fois mais chaque section a été retranscrite dans cet audit. Nous verrons également les versions dépréciées. Le choix a été fait de rendre aussi riche que possible cet audit, si le temps vous manque, vous pouvez vous rendre directement à la fin de chaque section.

## Analyse de performance Blackfire :

Afin de procéder à l'analyse de la performance nous allons utiliser le service SaaS Blackfire. Pour cela, nous avons migrer le projet sur Symfony 3.4 et PHP 7.2. Pour cette première partie, nous allons lancer des profiling sur toutes les requêtes possibles. Nous distinguons 2 types de requêtes ; celles qui sont accessibles sans être connecter et celles qui nécessitent d'être connecter pour y accéder. A chaque requête nous recuperons un ensemble de données :



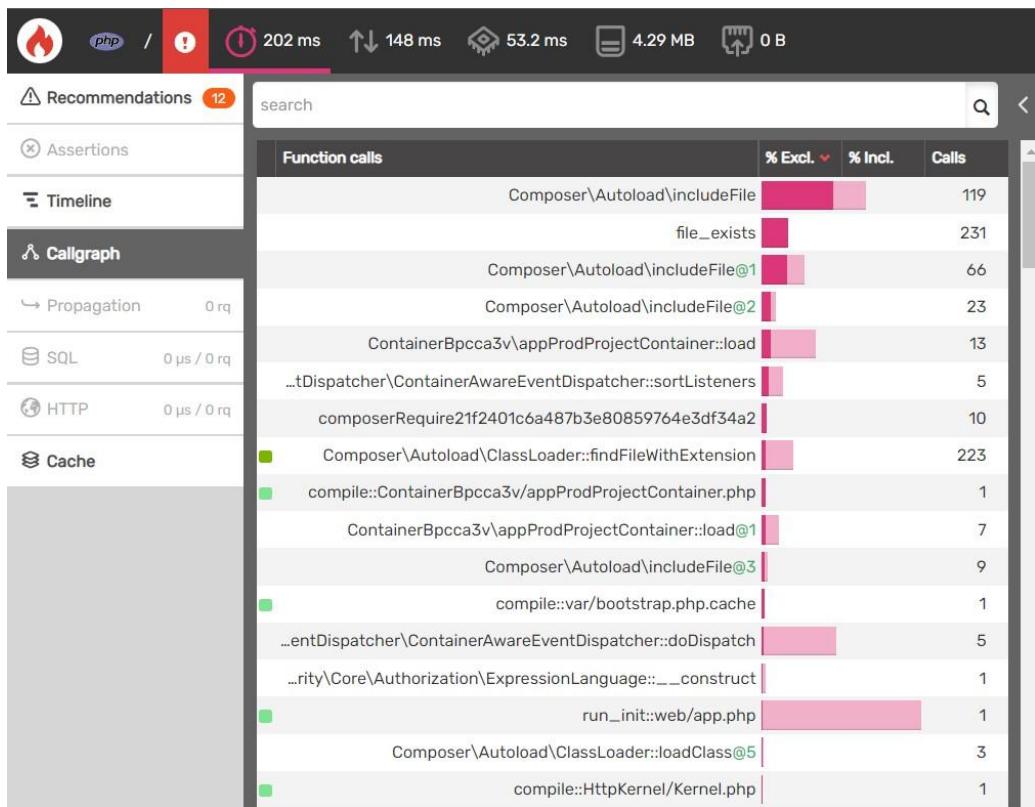
- 1 - le temps de chargement global de la page ( ici 202ms ).
- 2 - Le I/O time : le temps d'attente du système, temps durant lequel l'activité du processus n'est pas destiné à faire des calculs mais à attendre les opérations de lecture et d'écriture ( ici 148ms ).
- 3 - CPU Time : temps passé sur le processeur ( ici 53.2ms ).
- 4 - Le montant de la mémoire consommée par PHP ( ici 4.29MB ).
- 5 - Quantité de données réseau.



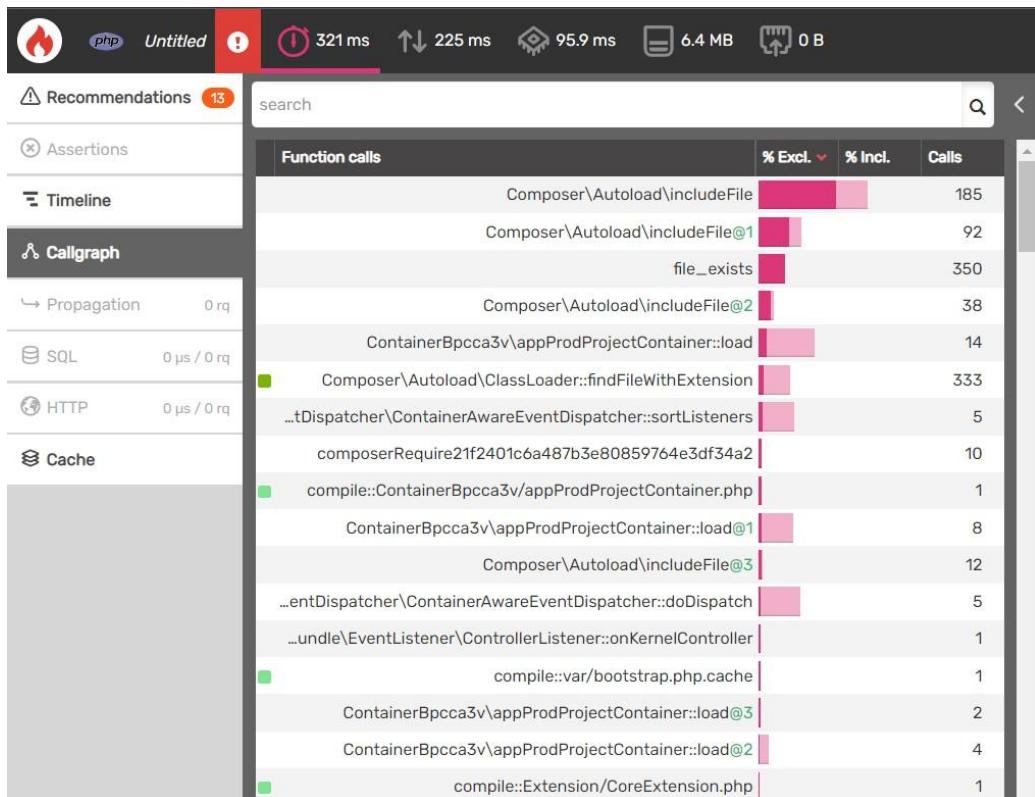
- 6 - A gauche, on récupère le nombre de requêtes SQL et le temps que cela a pris ( ici 1 requête pour 1.39ms )
- 7 - A droite nous avons les fonctions qui consomment le plus de temps en ordre décroissant : de la plus consommatrice à la moins consommatrice.

Certaines requêtes sont accessibles sans connection, d'autres demandent d'être connecter. Nous allons préciser celles qui peuvent être accessibles sans connection avec l'annotation (\*SC).

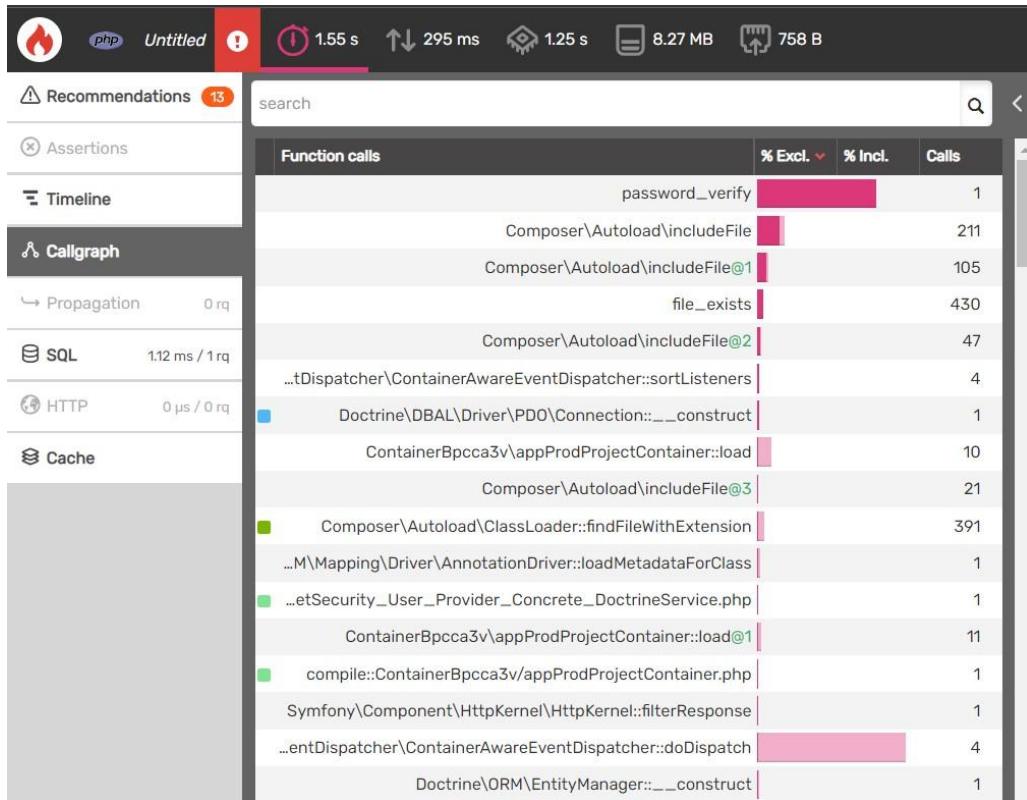
## Statut 302, Method GET, Path / :



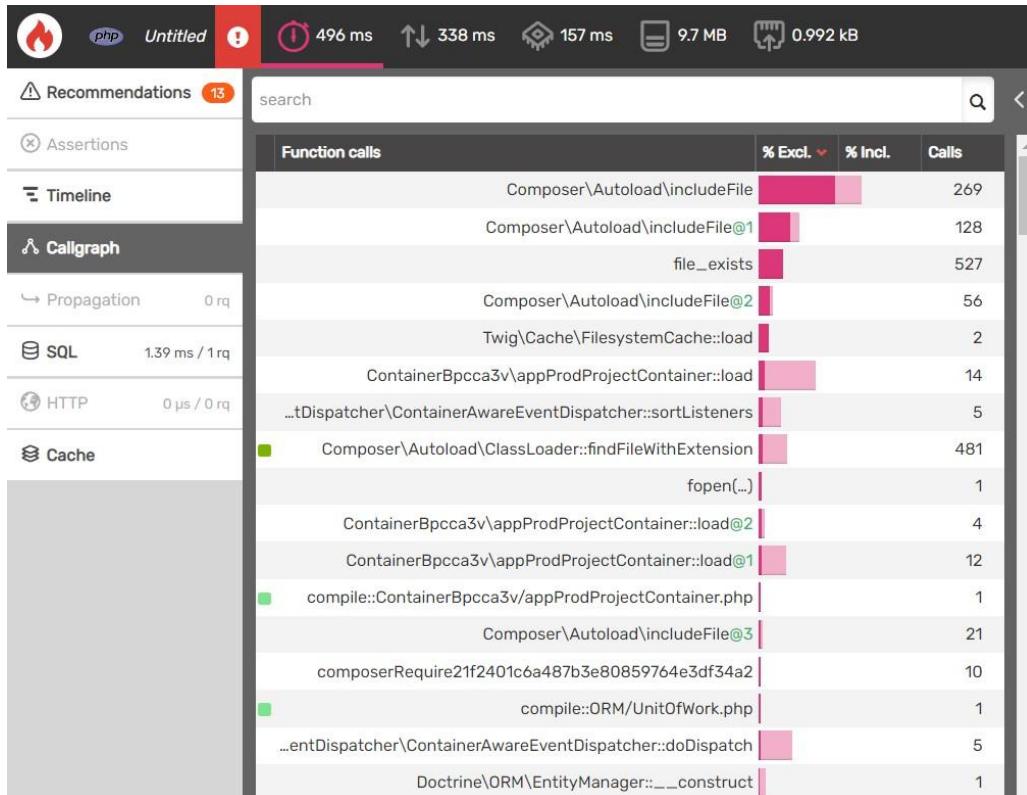
## Statut 200, Method GET, Path /login (\*SC) :



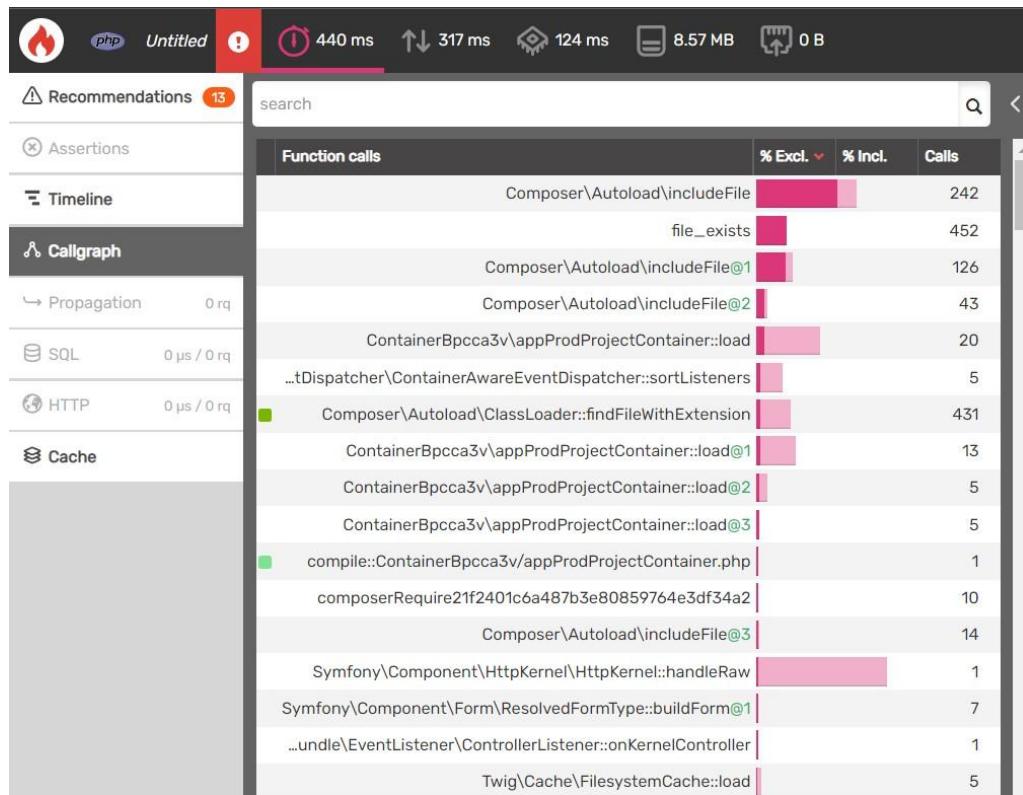
Statut 302, Method POST, Path /login\_check (\*SC) :



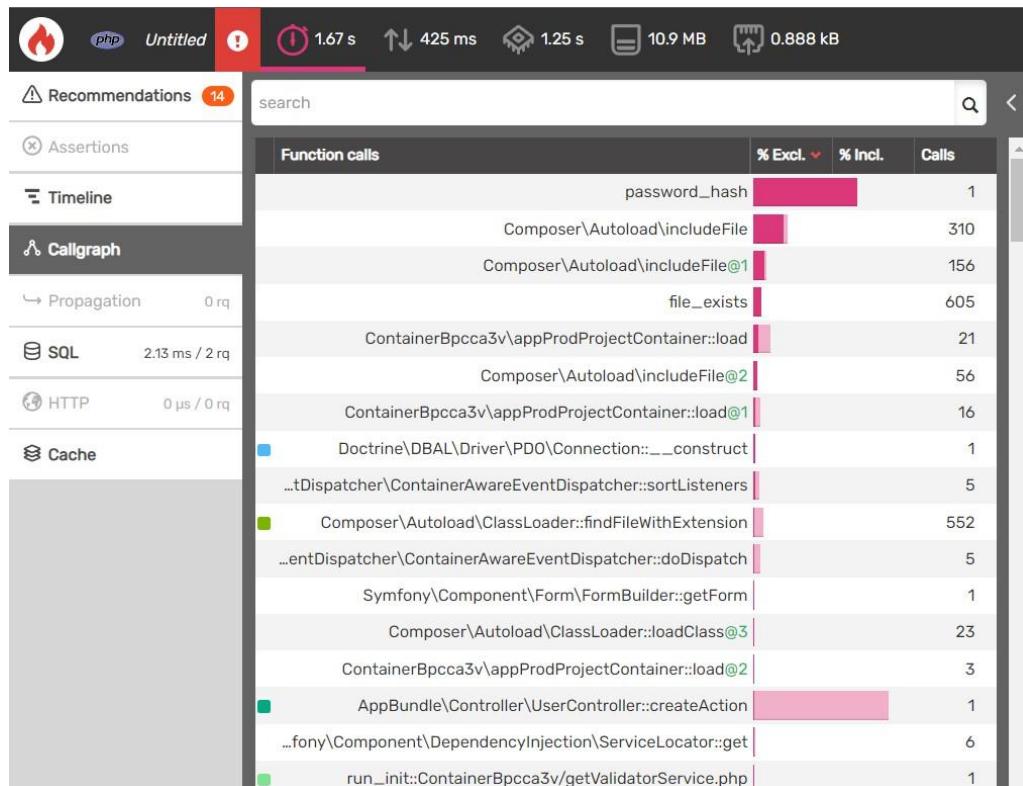
Statut 200, Method GET, Path /users (\*SC) :



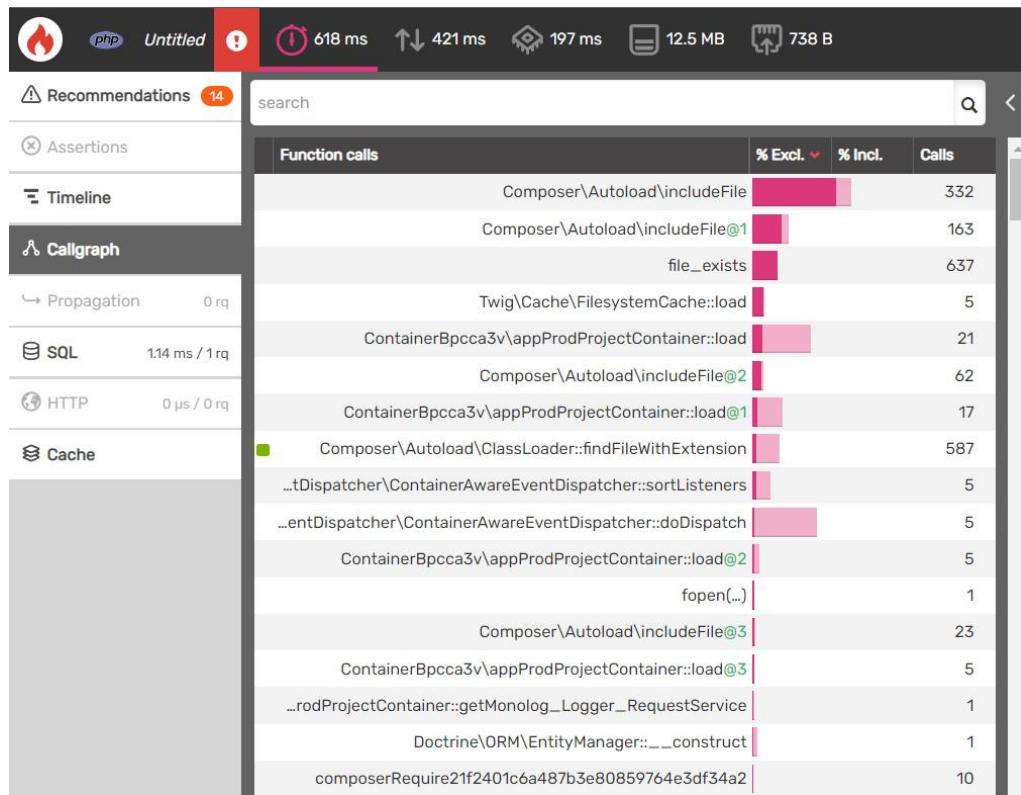
Statut 200, Method GET, Path /users/create (\*SC) :



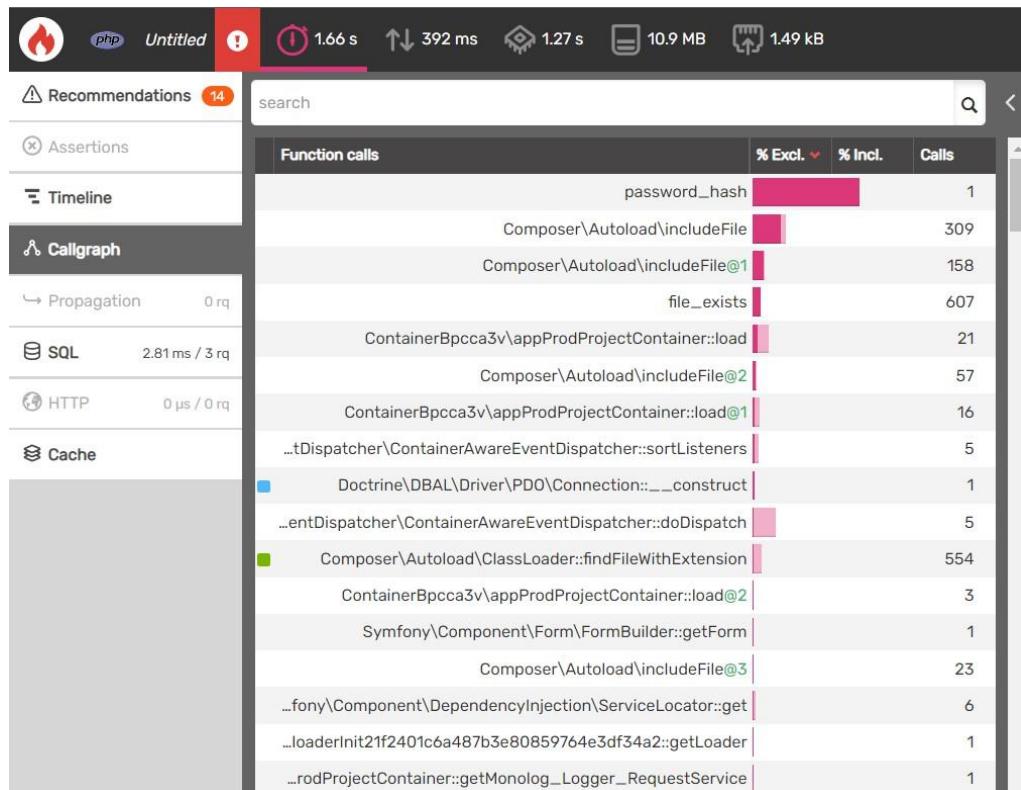
Statut 302, Method POST, Path /users/create (\*SC) :



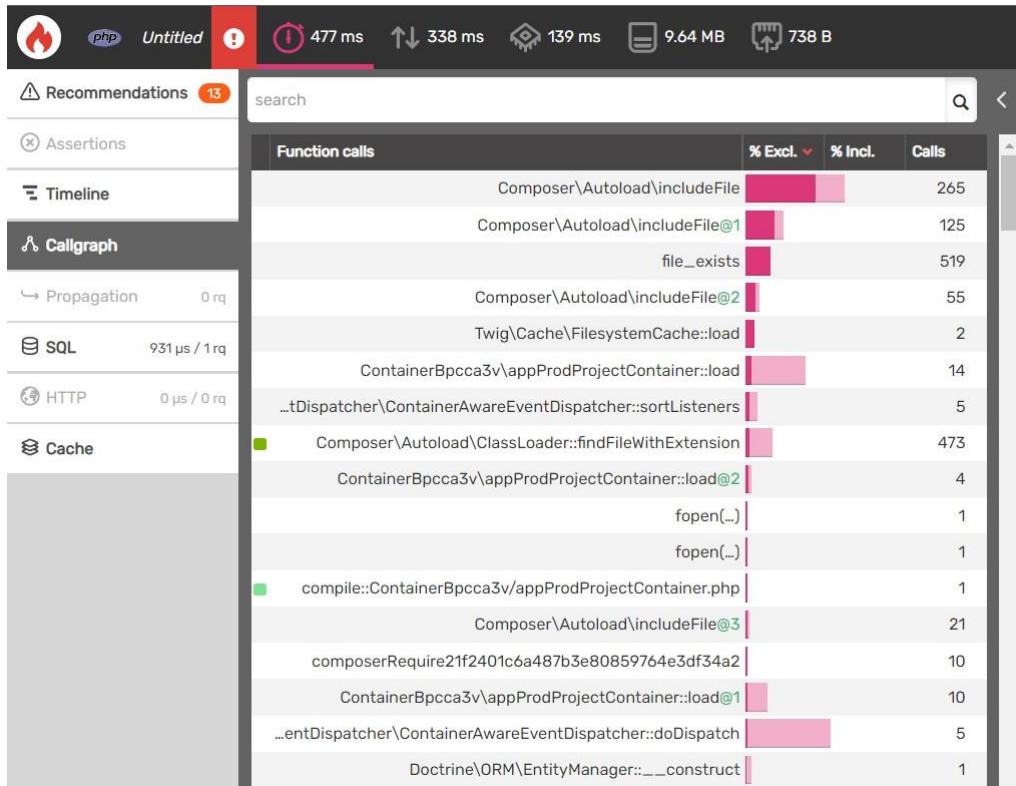
Statut 200, Method GET, Path /users/{id}/edit (\*SC) :



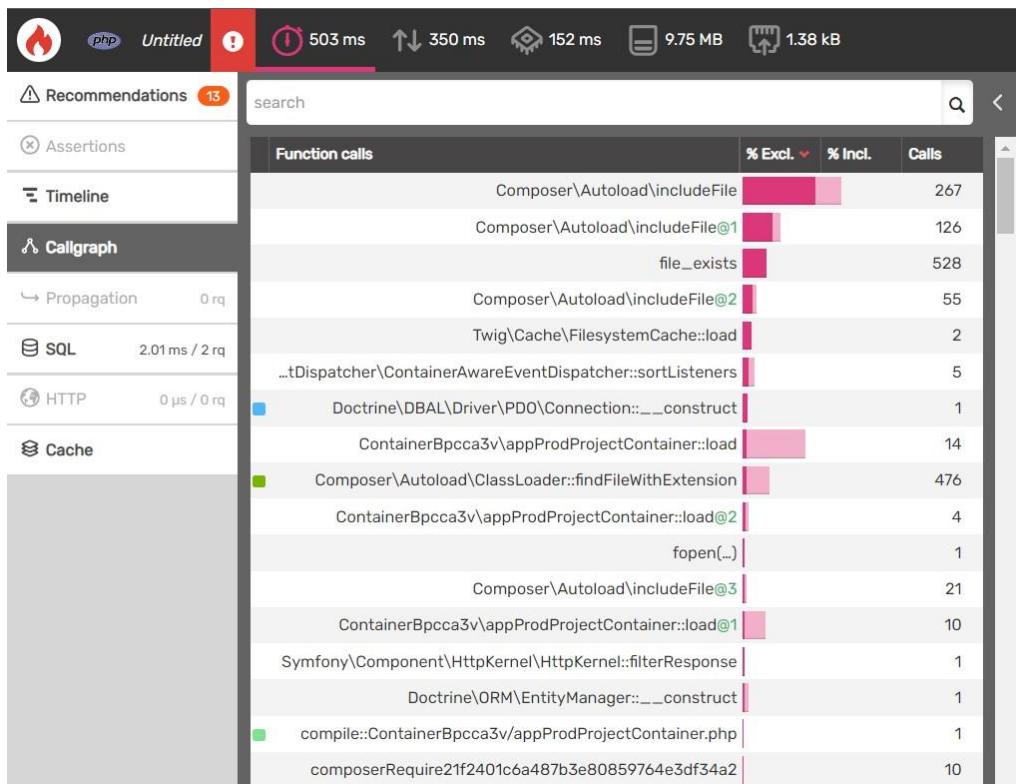
Statut 302, Method POST, Path /users/{id}/edit (\*SC) :



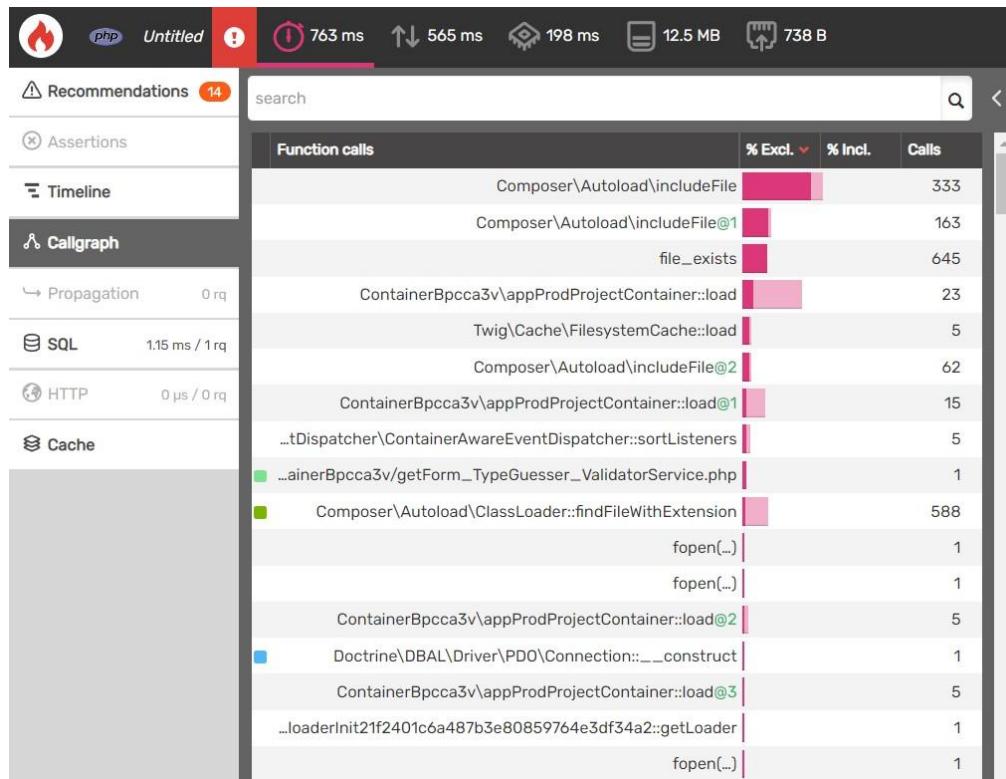
## Statut 200, Method GET, Path / :



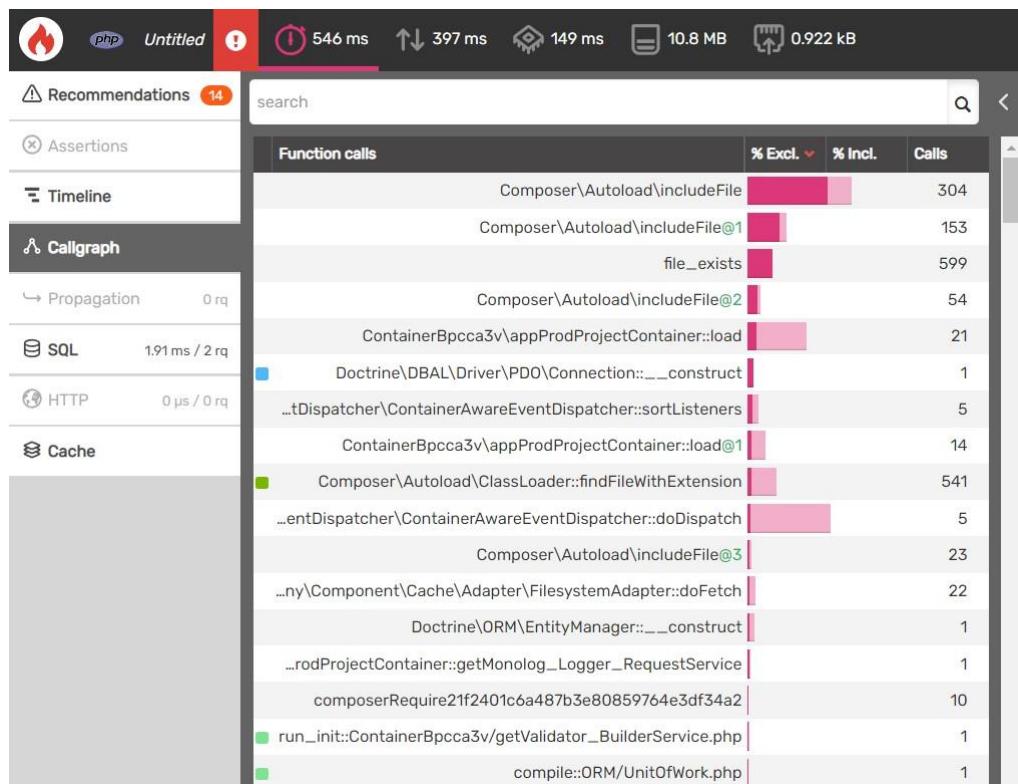
## Statut 200, Method GET, Path /tasks :



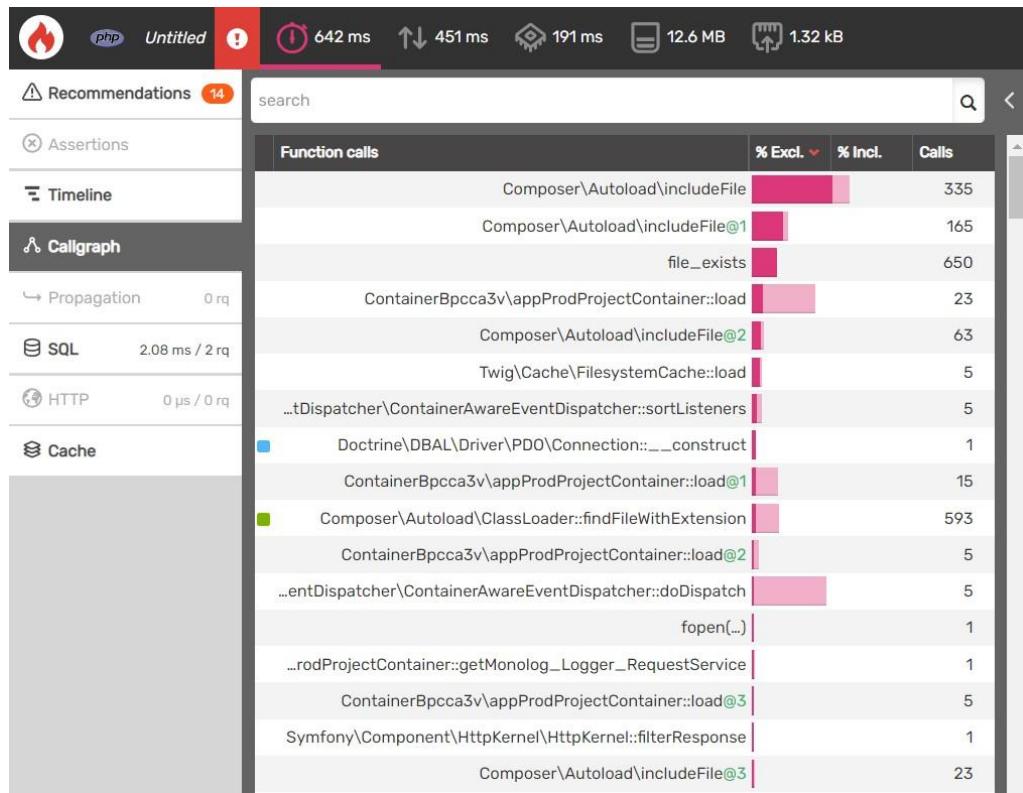
Statut 200, Method GET, Path /tasks/create :



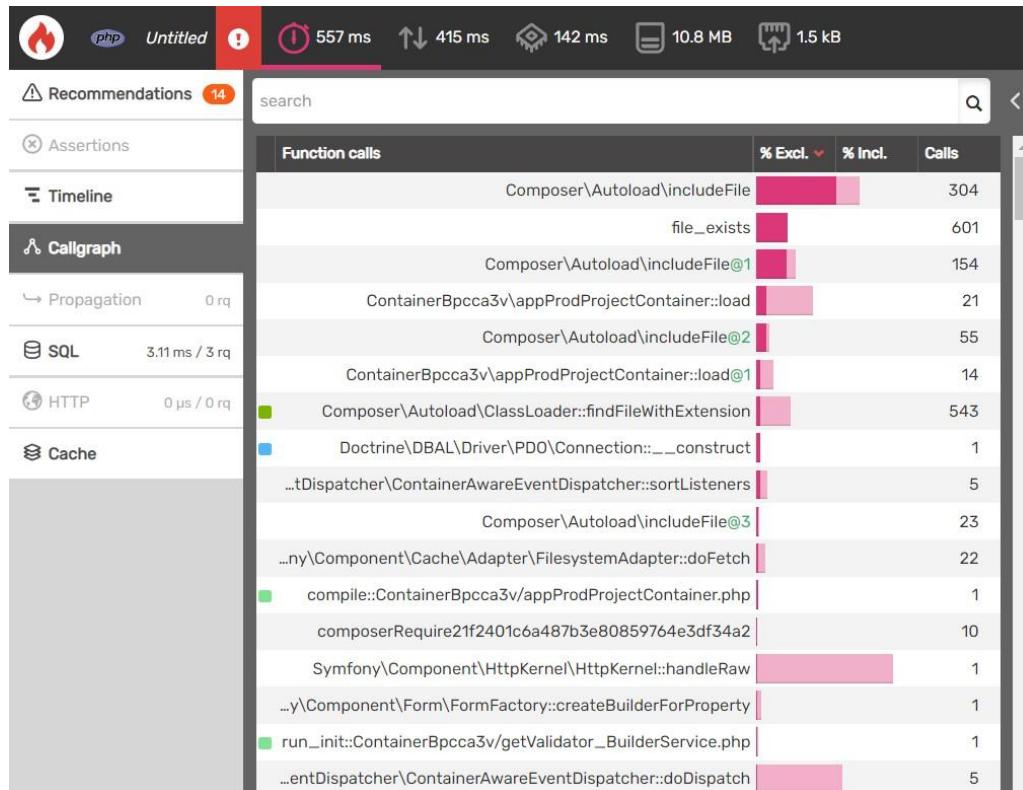
Statut 302, Method POST, Path /tasks/create :



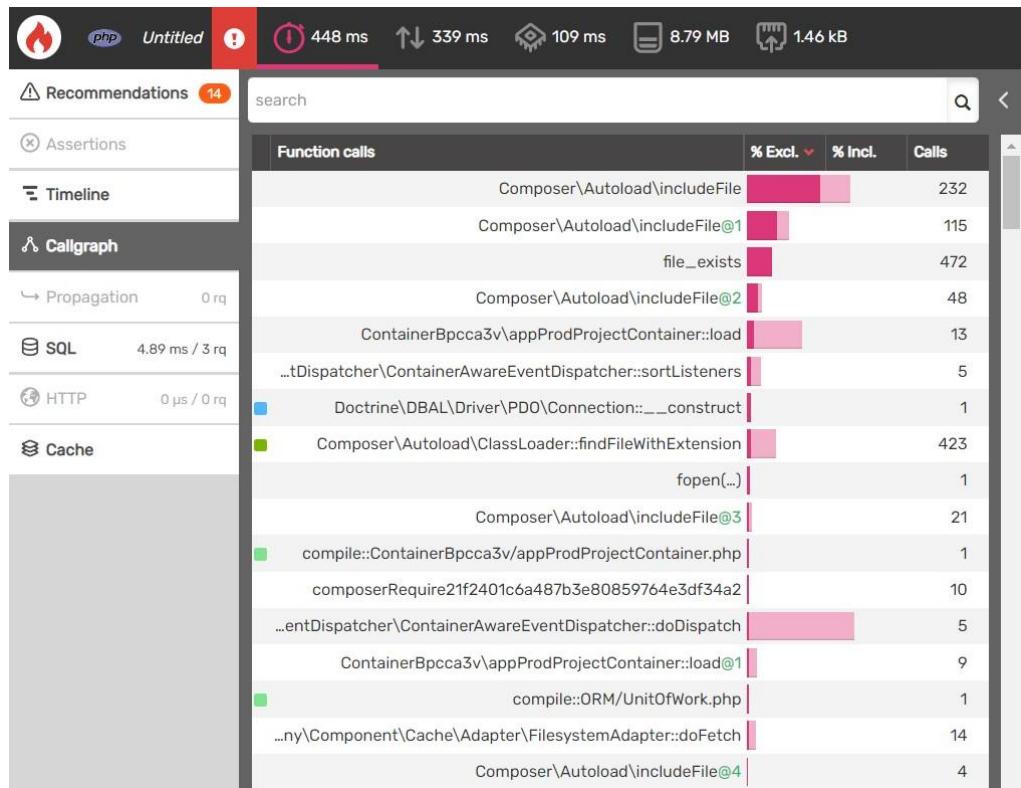
Statut 200, Method GET, Path /tasks/{id}/edit :



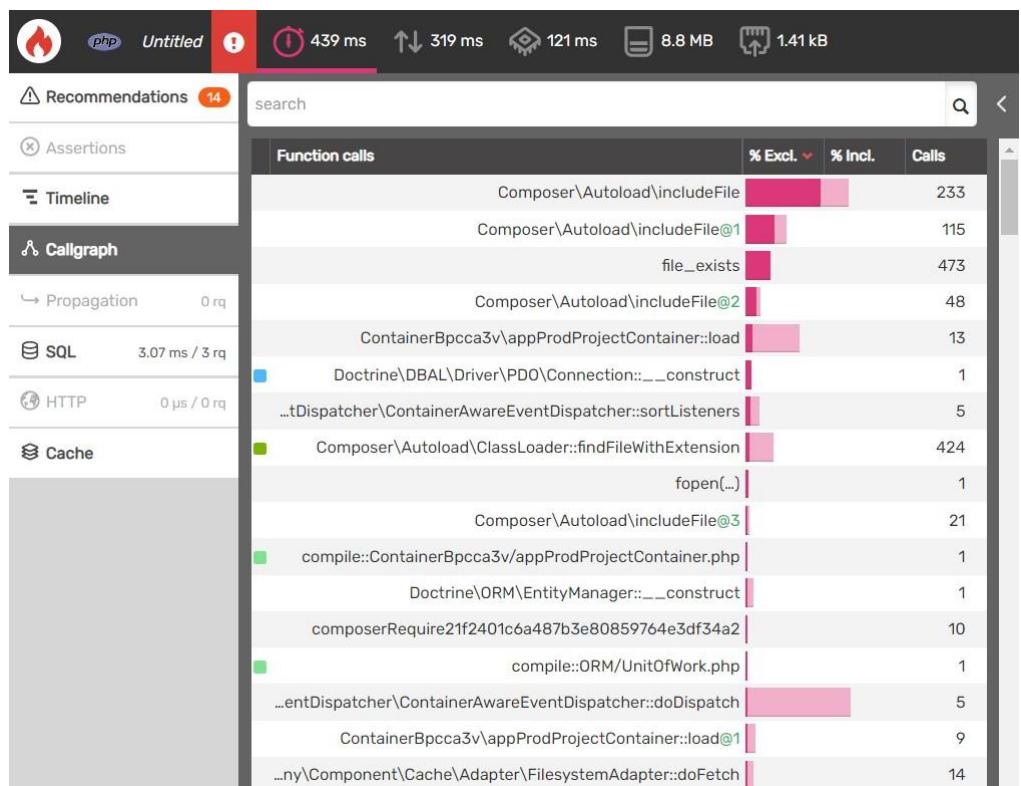
Statut 302, Method POST, Path /tasks/{id}/edit :



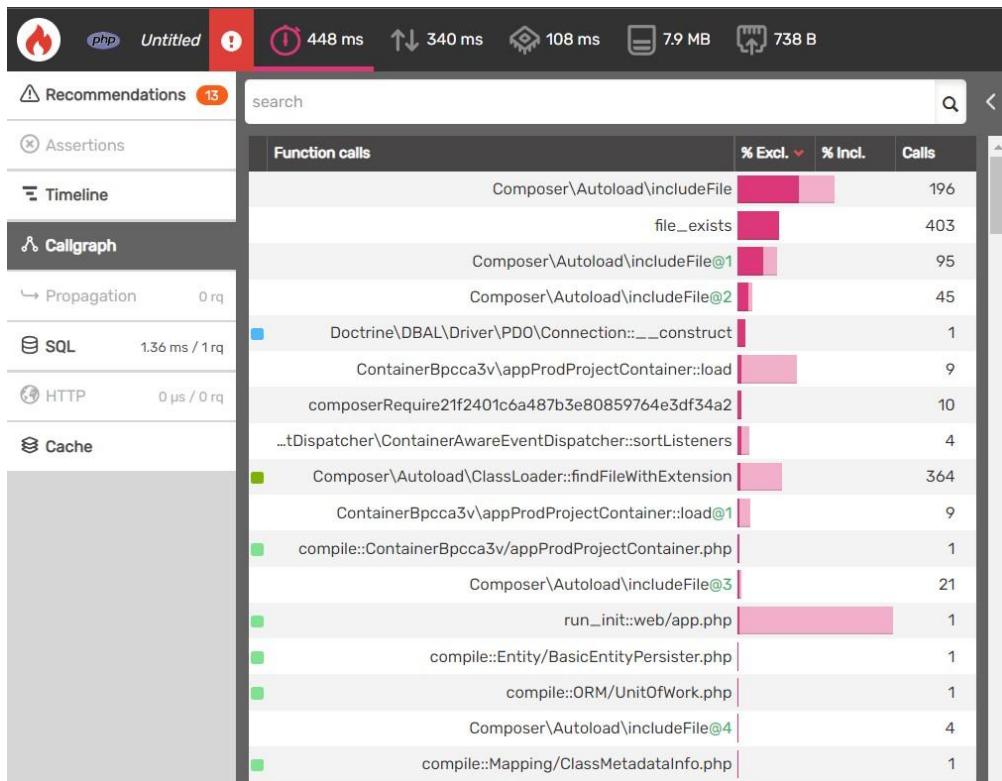
## Statut 302, Method POST, Path /tasks/{id}/toggle :



## Statut 302, Method POST, Path /tasks/{id}/delete :



Statut 302, Method GET, Path /logout :



### Conclusion de l'analyse de performance de Blackfire :

On voit que les requêtes les plus longues sont /login\_check avec 1.55 sec et /users/create et users/{if}/edit avec 1.66 sec, cela est tout-à-fait normal car elles font intervenir les fonctions consommatrices de temps et d'énergie que sont password\_verify() et password\_hasher(). Sur toutes les autres requêtes on voit la prédominance des fonctions propres à l'autoloader de Composer : Composer/Autoloader/includeFile() et file\_exists(), cela doit être optimisé par la mise en cache de l'autoloader de Composer. De même, on peut remarquer l'appel aux fonctions du Container de Symfony, une amélioration possible et de rassembler tous les services du Container dans un seul fichier. Cependant l'amélioration la plus notable à mettre en oeuvre est l'utilisation d'OPCache afin de mettre en cache le rendu de chaque page et gagner ainsi en temps et performance. On remarquera au passage de ces analyses, une faille de sécurité majeure qui est le libre accès au page d'édition et de consultation des utilisateurs ce qui devra également être corriger.

## Recapitulatif de l'analyse des requêtes HTTP :

Les différentes requêtes possibles	Temps	Mémoire
Statut 302, Method GET, Path / :	202 ms	4,29 MB
Statut 200, Method GET, Path /login :	321 ms	6,4 MB
Statut 302, Method POST, Path /login_check :	1,55 sec	8,27 MB
Statut 200, Method GET, Path /users :	496 ms	9,7 MB
Statut 200, Method GET, Path /users/create :	440 ms	8,57 MB
Statut 302, Method POST, Path /users/create :	1,67 sec	10,9 MB
Statut 200, Method GET, Path /users/{id}/edit :	618 ms	12,5 MB
Statut 302, Method POST, Path /users/{id}/edit :	1,66 sec	10,9 MB
Statut 200, Method GET, Path / :	477 ms	9,64 MB
Statut 200, Method GET, Path /tasks :	503 ms	9,75 MB
Statut 200, Method GET, Path /tasks/create :	763 ms	12,5 MB
Statut 302, Method POST, Path /tasks/create :	546 ms	10,8 MB
Statut 200, Method GET, Path /tasks/{id}/edit :	642 ms	12,6 MB
Statut 302, Method POST, Path /tasks/{id}/edit :	557 ms	10,8 MB
Statut 302, Method POST, Path /tasks/{id}/toggle :	448 ms	8,79 MB
Statut 302, Method POST, Path /tasks/{id}/delete :	439 ms	8,8 MB
Statut 302, Method GET, Path /logout :	448 ms	7,9 MB

## Depréciations :

La version de Symfony 3.4 (3.1 original) est obsolète.  
 PHP 7.2 ( 5.4 original) est également une ancienne version non à jour.

```
Ikan.Hiu@Suranadi MINGW64 ~/Desktop/projet8-TodoList (master)
$ composer update
Loading composer repositories with package information
Info from https://repo.packagist.org: #StandWithUkraine
Updating dependencies
Lock file operations: 0 installs, 3 updates, 0 removals
- Upgrading symfony/http-client-contracts (v1.1.12 => v1.1.13)
- Upgrading symfony/mime (v4.4.42 => v4.4.43)
- Upgrading symfony/service-contracts (v1.1.12 => v1.1.13)
Writing lock file
Installing dependencies from lock file (including require-dev)
Package operations: 0 installs, 3 updates, 0 removals
- Upgrading symfony/mime (v4.4.42 => v4.4.43): Extracting archive
- Upgrading symfony/service-contracts (v1.1.12 => v1.1.13): Extracting archive
- Upgrading symfony/http-client-contracts (v1.1.12 => v1.1.13): Extracting archive
Package doctrine/doctrine-cache-bundle is abandoned, you should avoid using it. No replacement was suggested.
Package doctrine/reflection is abandoned, you should avoid using it. Use roave/better-reflection instead.
Package sensio/distribution-bundle is abandoned, you should avoid using it. No replacement was suggested.
Package sensiolabs/security-checker is abandoned, you should avoid using it. Use https://github.com/fabpot/local-php-security-checker instead.
Package swiftmailer/swiftmailer is abandoned, you should avoid using it. Use symfony/mail instead.
Package symfony/swiftmailer-bundle is abandoned, you should avoid using it. Use symfony/mail instead.
Generating autoload files
28 packages you are using are looking for funding.
Use the 'composer fund' command to find out more!
phpstan/extensio-installer: Extensions installed
> Incenteev\ParameterHandler\ScriptHandler::buildParameters
Updating the "app/config/parameters.yml" file
> Sensio\Bundle\DistributionBundle\Composer\ScriptHandler::buildBootstrap
> Sensio\Bundle\DistributionBundle\Composer\ScriptHandler::clearCache
// Clearing the cache for the dev environment with debug true
```

Package doctrine/doctrine-cache-bundle is abandoned, you should avoid using it. No replacement was suggested.  
 Package doctrine/reflection is abandoned, you should avoid using it. Use roave/better-reflection instead.  
 Package sensio/distribution-bundle is abandoned, you should avoid using it. No replacement was suggested.  
 Package sensiolabs/security-checker is abandoned, you should avoid using it. Use https://github.com/fabpot/local-php-security-checker instead.  
 Package swiftmailer/swiftmailer is abandoned, you should avoid using it. Use symfony/mail instead.  
 Package symfony/swiftmailer-bundle is abandoned, you should avoid using it. Use symfony/mail instead.  
 Package phpunit/php-token-stream is abandoned, you should avoid using it. No replacement was suggested.  
 Package sensio/generator-bundle is abandoned, you should avoid using it. Use symfony/maker-bundle instead.

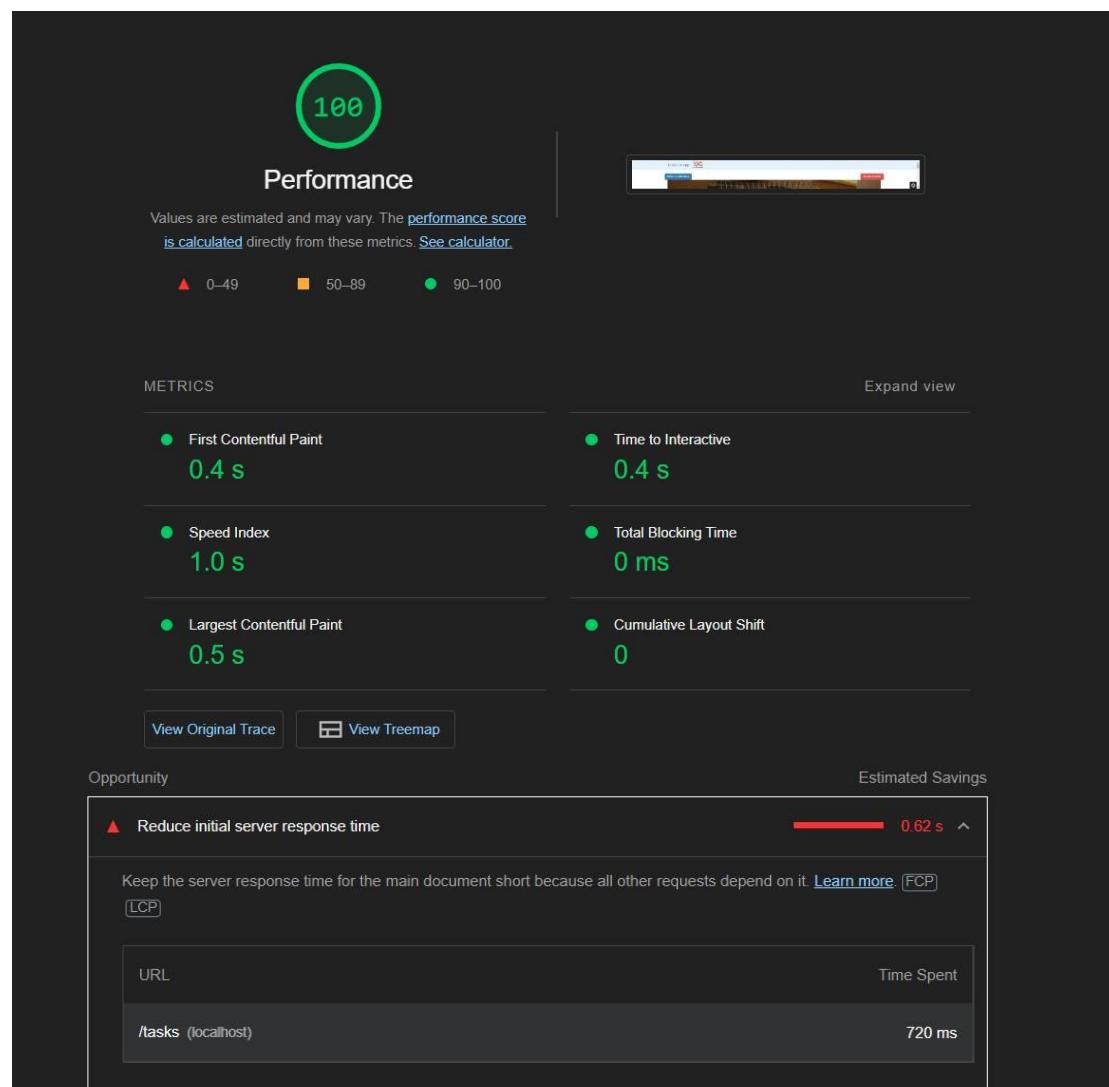
### Conclusion de l'analyse de dépréciations :

De nombreuses dépendances sont obsolètes et outre les problèmes de performance et de bug liés, elles peuvent engendrer des failles de sécurité majeures. De plus Symfony et PHP doivent être mis à jour à leur dernière version ( 6.1 pour Symfony et 8 pour PHP ). En conséquence, il est recommandé de partir d'un projet neuf puis d'installer «composer unused» afin de garantir d'effectivité des dépendances.

## Analyse de performance LightHouse :

L'analyse de performance avec le service Google LightHouse est orientée front-end et design. Cela nous donne une autre approche des problèmes liés à la performance. Cette analyse ne sera pas effectuer page par page mais de manière globale, afin de nous donner un aperçu des problèmes de performance rencontrés sur tout le site. L'analyse Google LightHouse s'effectue en plusieurs points : la Performance, l'Accessibilité, les Pratiques recommandées, le SEO et le Progressive web app. Ici l'accent sera donné sur la Performance mais afin de ne pas dénaturer l'analyse, les autres points seront également abordés dans cette section.

### Analyse de la performance :



**Enable text compression** 0.24 s ▲

Text-based resources should be served with compression (gzip, deflate or brotli) to minimize total network bytes. [Learn more](#) [FCP](#) [LCP](#)

URL	Transfer Size	Potential Savings
/css/bootstrap.min.css (localhost)	118.4 KiB	99.1 KiB
/tasks (localhost)	45.4 KiB	36.1 KiB
/js/bootstrap.min.js (localhost)	36.2 KiB	26.6 KiB
/_wdt/f78a08 (localhost)	25.8 KiB	20.4 KiB

**Reduce unused CSS** 0.16 s ▲

Reduce unused rules from stylesheets and defer CSS not used for above-the-fold content to decrease bytes consumed by network activity. [Learn more](#) [FCP](#) [LCP](#)

URL	Transfer Size	Potential Savings
/css/bootstrap.min.css (localhost)	118.5 KiB	112.7 KiB
.sf-minitoolbar { background-color: #222; border-top-left-radius: 4px; bottom: 0; ... } ...	12.0 KiB	11.0 KiB

**DIAGNOSTICS**

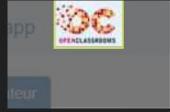
**Ensure text remains visible during webfont load**

Leverage the font-display CSS feature to ensure text is user-visible while webfonts are loading. [Learn more](#) [FCP](#) [LCP](#)

URL	Potential Savings
/fonts/glyphicons-halfings-regular.woff2 (localhost)	10 ms

**Image elements do not have explicit `width` and `height`**

Set an explicit width and height on image elements to reduce layout shifts and improve CLS. [Learn more](#) [CLS](#)

URL
 todo list <pre>todo list &lt;img class="slide-image" src="/img/todolist_content.jpg" alt="todo list"&gt;</pre>
 img <pre>img &lt;/img&gt;</pre>

▲ Serve static assets with an efficient cache policy — 6 resources found

A long cache lifetime can speed up repeat visits to your page. [Learn more](#).

URL	Cache TTL	Transfer Size
/css/bootstrap.min.css (localhost)	None	118 KiB
/img/todolist_content.jpg (localhost)	None	48 KiB
/js/bootstrap.min.js (localhost)	None	36 KiB
/fonts/glyphicons-halflings-regular.woff2 (localhost)	None	18 KiB
/img/Logo_OpenClassrooms.png (localhost)	None	7 KiB
/css/shop-homepage.css (localhost)	None	1 KiB

○ Avoid chaining critical requests — 4 chains found

The Critical Request Chains below show you what resources are loaded with a high priority. Consider reducing the length of chains, reducing the download size of resources, or deferring the download of unnecessary resources to improve page load. [Learn more](#) [FCP](#) [LCP](#)

Maximum critical path latency: **1,460 ms**

*Initial Navigation*

```

  /tasks (localhost)
    /css/bootstrap.min.css (localhost)
      /fonts/glyphicons-halflings-regular.woff2 (localhost) - 10 ms, 17.65 KiB
    /css/shop-homepage.css (localhost) - 10 ms, 0.90 KiB
    /js/jquery.js (localhost) - 710 ms, 64.08 KiB
    /js/bootstrap.min.js (localhost) - 0 ms, 36.23 KiB
  
```

○ Keep request counts low and transfer sizes small — 9 requests • 363 KiB

To set budgets for the quantity and size of page resources, add a budget.json file. [Learn more](#).

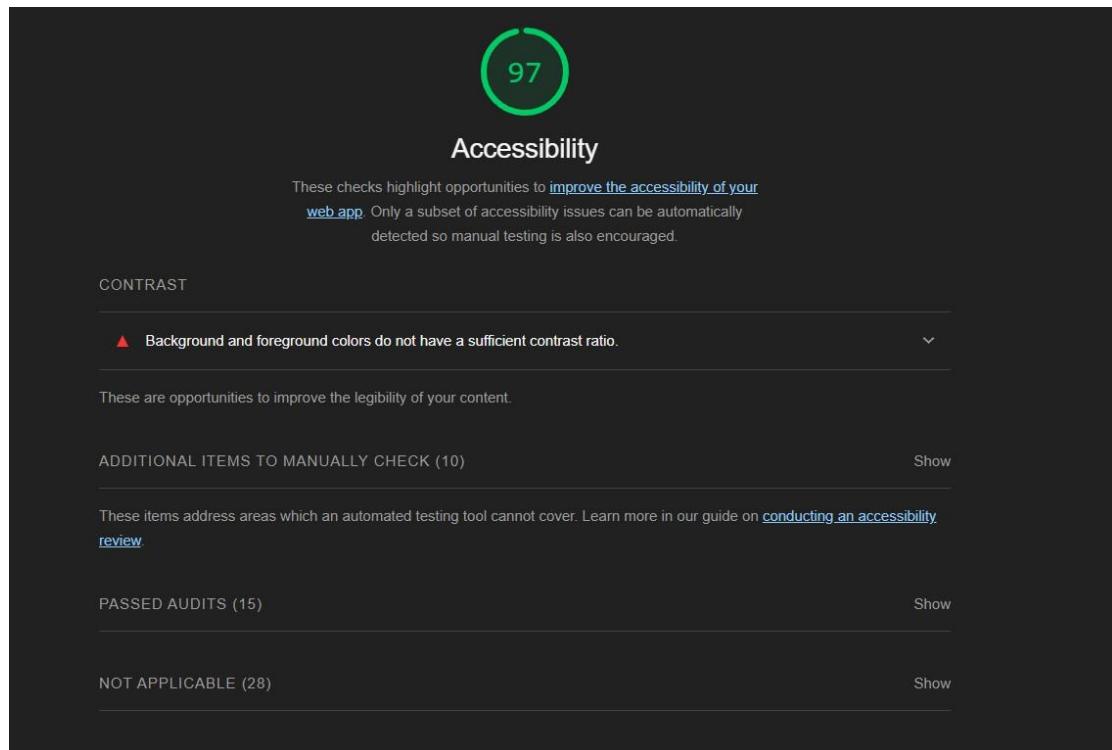
Resource Type	Requests	Transfer Size
Total	9	363.4 KiB
Stylesheet	2	119.4 KiB
Script	2	100.3 KiB
Image	2	54.7 KiB
Document	1	45.5 KiB
Other	1	25.9 KiB
Font	1	17.6 KiB
Media	0	0.0 KiB
Third-party	0	0.0 KiB

○ Largest Contentful Paint element — 1 element found

This is the largest contentful element painted within the viewport. [Learn More](#) [LCP](#)

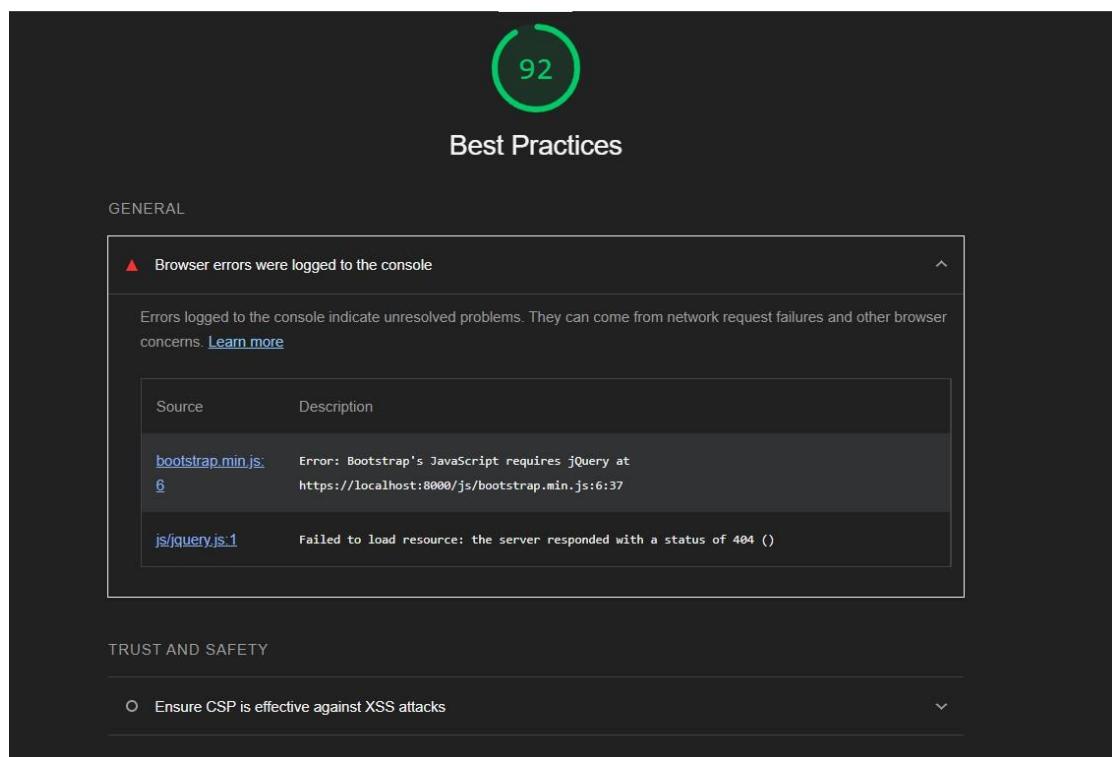
Element
 <code>img.slide-image</code>

## Analyse de l'accessibilité :



The screenshot shows an accessibility audit report with a green circular icon containing the number 97. Below it, the word "Accessibility" is displayed. A message states: "These checks highlight opportunities to [improve the accessibility of your web app](#). Only a subset of accessibility issues can be automatically detected so manual testing is also encouraged." A section titled "CONTRAST" shows a warning: "▲ Background and foreground colors do not have a sufficient contrast ratio." Below this, a note says: "These are opportunities to improve the legibility of your content." There are three main sections: "ADDITIONAL ITEMS TO MANUALLY CHECK (10)" which has a "Show" link; "PASSED AUDITS (15)" which also has a "Show" link; and "NOT APPLICABLE (28)" which also has a "Show" link.

## Analyse des Pratiques recommandées :



The screenshot shows a best practices audit report with a green circular icon containing the number 92. Below it, the words "Best Practices" are displayed. A section titled "GENERAL" contains a warning: "▲ Browser errors were logged to the console". A note explains: "Errors logged to the console indicate unresolved problems. They can come from network request failures and other browser concerns. [Learn more](#)". A table lists errors:

Source	Description
<a href="#">bootstrap.min.js: 6</a>	Error: Bootstrap's JavaScript requires jQuery at <a href="https://localhost:8000/js/bootstrap.min.js:6:37">https://localhost:8000/js/bootstrap.min.js:6:37</a>
<a href="#">js/jquery.js:1</a>	Failed to load resource: the server responded with a status of 404 ()

Below this, a section titled "TRUST AND SAFETY" contains a note: "○ Ensure CSP is effective against XSS attacks".

## Analyse du SEO :

The screenshot shows the Lighthouse SEO audit results for a website. The overall score is 82, displayed prominently at the top. Below the score, the category "SEO" is shown. A note states: "These checks ensure that your page is following basic search engine optimization advice. There are many additional factors Lighthouse does not score here that may affect your search ranking, including performance on Core Web Vitals. [Learn more](#)".

**CONTENT BEST PRACTICES**

- ⚠ Document does not have a meta description. Description text is empty.

Format your HTML in a way that enables crawlers to better understand your app's content.

**CRAWLING AND INDEXING**

- ⚠ Links are not crawlable

To appear in search results, crawlers need access to your app.

**ADDITIONAL ITEMS TO MANUALLY CHECK (1)** Show

Run these additional validators on your site to check additional SEO best practices.

## Analyse des Progressive web app :

The screenshot shows the Lighthouse PWA audit results for a website. The overall score is 100, displayed prominently at the top. Below the score, the category "PWA" is shown. A note states: "These checks validate the aspects of a Progressive Web App. [Learn more](#)".

**INSTALLABLE**

- ⚠ Web app manifest or service worker do not meet the installability requirements — 1 reason

**PWA OPTIMIZED**

- ⚠ Does not register a service worker that controls page and start\_url
- ⚠ Is not configured for a custom splash screen. Failures: No manifest was fetched.
- ⚠ Does not set a theme color for the address bar.  
Failures: No manifest was fetched, No '<meta name="theme-color">' tag found.
- Content is sized correctly for the viewport
- Has a <meta name="viewport"> tag with width or initial-scale
- ⚠ Does not provide a valid apple-touch-icon

### Conclusion de l'analyse de performance de LightHouse :

Outre la reduction du temps de reponse déjà abordé precedemment, l'analyse de Google LightHouse nous ouvre de nouveaux champs d'ameliorations. Cette analyse n'affecte que les medias et les fichiers front-end, et elle complète ainsi l'analyse de performance de Blackfire. Pour ce qui concerne les fichiers css et js, il est conseillé de les minifier et de compresser au maximum ainsi que se débarrasser du code css non utilisé. Le tout doit être mis en cache afin d'optimiser le temps de chargement. L'appel aux fichiers doit se faire de manière individualisé. L'analyse a également révélé une variable jQuery non définie dans bootstrap.min.js, en cause l'appel à un fichier jquery.js non existant. Il faudra veiller à installer la dernière version de bootstrap afin de ne pas s'exposer à des vulnérabilités connues. Même recommandations d'optimisations en ce qui concerne les medias ( les images ).

---

### **Conclusion à l'analyse de performance :**

L'analyse de performance a été effectué via l'analyseur Blackfire pour le back-end et avec Google LightHouse pour le front-end. L'architecture Symfony/PHP utilisée dans le projet est obsolète, il est nécessaire et impératif de mettre à jour ces versions.

Globalement, l'analyse de performance Back-End est mauvaise : l'optimisation de l'autoloader Composer et la mise en cache via OPCache sont les principales améliorations à adopter pour corriger cela. Pour le front-end, l'analyse s'est révélée correcte, il faut mettre à jour la version utilisée de jQuery mais également compresser et mettre en cache les fichiers et les medias.

## **Partie 2 : Analyse de la qualité**

---

### **Introduction :**

Afin de procéder à l'analyse de qualité nous allons adopter deux approches. La première approche consiste à analyser le code avec les outils d'analyse du code et les outils d'analyse du rendu html. Chacun des outils analysant le code analyse un aspect ou un langage différent. Les outils d'analyse du rendu html sont eux un peu différents dans le sens où il s'agit d'outils en ligne analysant le rendu de nos pages web. Cette première approche est intéressante mais loin d'être suffisante, ainsi ces outils d'analyses sont capables de remonter des erreurs de typographies, de logiques de typage ou autres mais peuvent passer aisement à côté de failles de sécurité ou de mise en forme illogiques. Ainsi cette première approche sera complétée par l'analyse visuelle du développeur aguerri qui pourra, lui, remonter les illogismes, les dangers ou les pratiques à améliorer qui avaient échappé à la première analyse.

## Analyses utilisant les outils d'analyse du code :

### Analyse de la qualité du code avec PHP Stan :

L'analyseur PHPStan analyse la qualité de l'écriture du code PHP. C'est un outil intéressant qui possède plusieurs niveaux de qualités. Le niveau 1 est une analyse très basique alors que le niveau 10 est une analyse très poussée. Les analyses de bas niveaux ne seront pas retransmises ici. Seul l'analyse de niveau 9 sera ici retransmise.

### Analyse complète de PHPStan - Niveau 9 :

```
Ikan Hiu@Suranadi MINGW64 ~/Desktop/projet8-TodoList (master)
$ vendor/bin/phpstan analyse
Note: Using configuration file C:\Users\Ikan Hiu\Desktop\projet8-TodoList\phpstan.neon.
10/10 [======] 100%

Line  src\AppBundle\Controller\DefaultController.php
13   Method AppBundle\Controller\DefaultController::indexAction() has no return type specified.

Line  src\AppBundle\Controller\SecurityController.php
14   Method AppBundle\Controller\SecurityController::loginAction() has no return type specified.
30   Method AppBundle\Controller\SecurityController::loginCheck() has no return type specified.
38   Method AppBundle\Controller\SecurityController::logoutCheck() has no return type specified.

Line  srcAppBundle\Controller\TaskController.php
16   Method AppBundle\Controller\TaskController::listAction() has no return type specified.
18   Parameter #1 $persistentObject of method
        Doctrine\Persistence\ManagerRegistry::getRepository() expects class-string<AppBundle:Task>,
        string given.
24   Method AppBundle\Controller\TaskController::createAction() has no return type specified.
48   Method AppBundle\Controller\TaskController::editAction() has no return type specified.
71   Method AppBundle\Controller\TaskController::toggleTaskAction() has no return type
        specified.
84   Method AppBundle\Controller\TaskController::deleteTaskAction() has no return type
        specified.

Line  srcAppBundle\Controller\UserController.php
16   Method AppBundle\Controller\UserController::listAction() has no return type specified.
18   Parameter #1 $persistentObject of method
        Doctrine\Persistence\ManagerRegistry::getRepository() expects class-string<AppBundle:User>,
        string given.
24   Method AppBundle\Controller\UserController::createAction() has no return type specified.
50   Method AppBundle\Controller\UserController::editAction() has no return type specified.

Line  srcAppBundle\Entity\Task.php
19   Property AppBundle\Entity\Task::$id has no type specified.
24   Property AppBundle\Entity\Task::$createdAt has no type specified.
30   Property AppBundle\Entity\Task::$title has no type specified.
36   Property AppBundle\Entity\Task::$content has no type specified.
41   Property AppBundle\Entity\Task::$isDone has no type specified.
49   Method AppBundle\Entity\Task::getId() has no return type specified.
54   Method AppBundle\Entity\Task::getCreatedAt() has no return type specified.
59   Method AppBundle\Entity\Task::setCreatedAt() has no return type specified.
59   Method AppBundle\Entity\Task::setCreatedAt() has parameter $createdAt with no type
        specified.
```

```

specified.
64 Method AppBundle\Entity\Task::getTitle() has no return type specified.
69 Method AppBundle\Entity\Task::setTitle() has no return type specified.
69 Method AppBundle\Entity\Task::setTitle() has parameter $title with no type specified.
74 Method AppBundle\Entity\Task::getContent() has no return type specified.
79 Method AppBundle\Entity\Task::setContent() has no return type specified.
79 Method AppBundle\Entity\Task::setContent() has parameter $content with no type specified.
84 Method AppBundle\Entity\Task::isDone() has no return type specified.
89 Method AppBundle\Entity\Task::toggle() has no return type specified.
89 Method AppBundle\Entity\Task::toggle() has parameter $flag with no type specified.

-----
Line src\AppBundle\Entity\User.php

22 Property AppBundle\Entity\User::$id has no type specified.
28 Property AppBundle\Entity\User::$username has no type specified.
33 Property AppBundle\Entity\User::$password has no type specified.
40 Property AppBundle\Entity\User::$email has no type specified.
42 Method AppBundle\Entity\User::getId() has no return type specified.
52 Method AppBundle\Entity\User::getUsername() has no return type specified.
52 Method AppBundle\Entity\User::setUsername() has parameter $username with no type specified.
67 Method AppBundle\Entity\User::setPassword() has no return type specified.
67 Method AppBundle\Entity\User::setPassword() has parameter $password with no type specified.
72 Method AppBundle\Entity\User::getEmail() has no return type specified.
77 Method AppBundle\Entity\User::setEmail() has no return type specified.
77 Method AppBundle\Entity\User::setEmail() has parameter $email with no type specified.
87 Method AppBundle\Entity\User::eraseCredentials() has no return type specified.

-----
Line src\AppBundle\Form\TaskType.php

11 Method AppBundle\Form\TaskType::buildForm() has no return type specified.

-----
Line src\AppBundle\Form\UserType.php

14 Method AppBundle\Form\UserType::buildForm() has no return type specified.

-----
Line tests\AppBundle\Controller\DefaultControllerTest.php

9 Method Tests\AppBundle\Controller\DefaultControllerTest::testIndex() has no return type
specified.
15 Call to an undefined method
Tests\AppBundle\Controller\DefaultControllerTest::assertEquals().
15 Cannot call method getStatusCode() on Symfony\Component\HttpFoundation\Response|null.
16 Call to an undefined method
Tests\AppBundle\Controller\DefaultControllerTest::assertContains().

[ERROR] Found 51 errors

```

Niveaux 1 - 2 - 3 - 4 ==> 2 erreurs  
 Niveau 5 ==> 4 erreurs  
 Niveaux 6 - 7 ==> 50 erreurs  
 Niveaux 8 - 9 ==> 51 erreurs

Conclusion de l'analyse PhpStan :

Le rapport est plutôt bon jusqu'à niveau 5 mais ne résiste pas au passage des niveaux 6 et 7, en cause l'absence des types d'arguments passés aux fonctions et aux méthodes ainsi que l'absence de typehint. Le code doit être revu et doit être plus strict, les types d'argument et de retour doivent être implementés pour respecter les normes d'écriture et éviter les bugs de fonctionnement. Le code doit être corrigé et maintenu au niveau 9 de PhpStan.

## Analyse de la qualité du code avec PHP-CS-FIXER :

```
Ikan Hiu@Suranadi MINGW64 ~/Desktop/projet8-TodoList (master)
$ vendor/bin/php-cs-fixer fix src -vvv --dry-run --show-progress=dots
Loaded config default.
Using cache file ".php_cs.cache".
FFFFFF
Legend: ?-unknown, I-invalid file syntax, file ignored, S-Skipped, .-no changes, F-fixed, E-error
  1) src\\AppBundle\\AppBundle.php (line_ending)
  2) src\\AppBundle\\Controller\\DefaultController.php (line_ending)
  3) src\\AppBundle\\Controller\\SecurityController.php (line_ending)
  4) src\\AppBundle\\Controller\\TaskController.php (line_ending, braces)
  5) src\\AppBundle\\Controller\\UserController.php (line_ending)
  6) src\\AppBundle\\Entity\\Task.php (line_ending)
  7) src\\AppBundle\\Entity\\User.php (line_ending)
  8) src\\AppBundle\\Form\\TaskType.php (line_ending)
  9) src\\AppBundle\\Form\\UserType.php (line_ending)

Checked all files in 0.339 seconds, 10.000 MB memory used
```

L'analyseur Php-Cs-Fixer est un autre outil d'analyse côté back-end. Php-Cs-Fixer est un correcteur d'erreurs d'écriture du PHP. L'ensemble des fichiers doivent être corrigés ( Fixed marqué F ) par le correcteur Php-Cs-Fixer.

## Analyse de la qualité du code avec TwigCS :

```
Ikan Hiu@Suranadi MINGW64 ~/Desktop/Symfony/P8-ToDoApp (main)
$ vendor/bin/twigcs "C:\Users\Ikan Hiu\Desktop\Symfony\P8-ToDoApp\templates"
\templates\task\create.html.twig
l.5 c.37 : ERROR There should be 0 space between the key and ":".
\templates\task\edit.html.twig
l.5 c.37 : ERROR There should be 0 space between the key and ":".
l.5 c.59 : ERROR There should be 0 space before the hash values.
l.5 c.64 : ERROR There should be 0 space between the key and ":".
l.5 c.74 : ERROR There should be 0 space after the hash values.
\templates\task\list.html.twig
l.15 c.59 : ERROR There should be 0 space between the key and ":".
l.15 c.69 : ERROR There should be 0 space after the hash values.
l.19 c.62 : ERROR There should be 0 space between the key and ":".
l.19 c.72 : ERROR There should be 0 space after the hash values.
l.24 c.62 : ERROR There should be 0 space between the key and ":".
l.24 c.72 : ERROR There should be 0 space after the hash values.
\templates\user\create.html.twig
l.8 c.37 : ERROR There should be 0 space between the key and ":".
\templates\user\edit.html.twig
l.8 c.37 : ERROR There should be 0 space between the key and ":".
l.8 c.63 : ERROR There should be 0 space between the key and ":".
\templates\user\list.html.twig
l.30 c.63 : ERROR There should be 0 space between the key and ":".
15 violation(s) found
```

TwigCs est un analyseur pour le modèle de templating Twig côté Front-End. L'analyse est satisfaisante. On corrigera cependant les violations minimes remontées ci-dessus.

### Analyse de la qualité du code avec ESLint:

```
Ikan Hiu@Suranadi MINGW64 ~/Desktop/Symfony/P8-ToDoApp (main)
$ yarn run eslint "public/js/bootstrap.js"
yarn run v1.22.17
$ "C:\Users\Ikan Hiu\Desktop\Symfony\P8-ToDoApp\node_modules\.bin\eslint" public
/js/bootstrap.js

C:\Users\Ikan Hiu\Desktop\Symfony\P8-ToDoApp\public\js\bootstrap.js
  7:23  error  Strings must use doublequote          quotes
  8:19  error  Strings must use doublequote          quotes
  8:61  error  Missing semicolon                   semi
 12:3   error  Strings must use doublequote          quotes
 13:35  error  Strings must use doublequote          quotes
 13:49  error  Strings must use doublequote          quotes
 13:53  error  Missing semicolon                   semi
 15:21  error  Strings must use doublequote          quotes
 15:113  error  Missing semicolon                   semi
 17:3   error  'jQuery' is not defined            no-undef
 29:3   error  Strings must use doublequote          quotes
 35:14  error  'document' is not defined           no-undef
 35:37  error  Strings must use doublequote          quotes
 35:49  error  Missing semicolon                   semi
 38:26  error  Strings must use doublequote          quotes

.....
 2345:23  error  Missing semicolon                   semi
 2347:34  error  Missing semicolon                   semi
 2348:33  error  Missing semicolon                   semi
 2355:21  error  Missing semicolon                   semi
 2356:16  error  Missing semicolon                   semi
 2357:4   error  Missing semicolon                   semi
 2363:5   error  'window' is not defined            no-undef
 2363:16  error  Strings must use doublequote          quotes
 2364:7   error  Strings must use doublequote          quotes
 2365:25  error  Missing semicolon                   semi
 2366:29  error  Missing semicolon                   semi
 2368:38  error  Missing semicolon                   semi
 2370:76  error  Missing semicolon                   semi
 2371:73  error  Missing semicolon                   semi
 2373:30  error  Missing semicolon                   semi
 2374:7   error  Missing semicolon                   semi
 2375:5   error  Missing semicolon                   semi
 2377:3   error  'jQuery' is not defined            no-undef

✖ 1597 problems (1597 errors, 0 warnings)
 1529 errors and 0 warnings potentially fixable with the '--fix' option.

error Command failed with exit code 1.
info Visit https://yarnpkg.com/en/docs/cli/run for documentation about this command.
```

L'analyseur ESLint est un outil d'analyse du Javascript côté Front-End. L'analyse nous remonte ici, les problèmes que nous avons déjà mis en avant lors de l'analyse avec Google LightHouse que sont les variables indefinies ( ici jQuery et window ). Les autres problèmes sont des erreurs d'écriture dans la qualité du code. Le code Javascript devra être remis à jour.

## Analyse de la qualité du code avec StyleLint:

```
Ikan Hiu@Suranadi MINGW64 ~/Desktop/Symfony/P8-ToDoApp (main)
$ yarn run stylelint "public/css/shop-homepage.css"
yarn run v1.22.17
$ "C:\Users\Ikan Hiu\Desktop\Symfony\P8-ToDoApp\node_modules\.bin\stylelint" public/css/shop-homepage.css

public/css/shop-homepage.css
  8:5   ✘ Expected indentation of 2 spaces           indentation
     8:144 ✘ Expected line length to be no more than 120 characters max-line-len
gth
  12:5   ✘ Expected indentation of 2 spaces           indentation
  16:5   ✘ Expected indentation of 2 spaces           indentation
  21:5   ✘ Expected indentation of 2 spaces           indentation
  25:5   ✘ Expected indentation of 2 spaces           indentation
  26:5   ✘ Expected indentation of 2 spaces           indentation
  30:5   ✘ Expected indentation of 2 spaces           indentation
  34:5   ✘ Expected indentation of 2 spaces           indentation
  38:5   ✘ Expected indentation of 2 spaces           indentation
  39:5   ✘ Expected indentation of 2 spaces           indentation
  40:5   ✘ Expected indentation of 2 spaces           indentation
  44:5   ✘ Expected indentation of 2 spaces           indentation
  48:5   ✘ Expected indentation of 2 spaces           indentation
  49:5   ✘ Expected indentation of 2 spaces           indentation
  53:5   ✘ Expected indentation of 2 spaces           indentation
  54:1   ✘ Unexpected missing end-of-source newline no-missing-e
nd-of-source-newline
```

L'analyseur StyleLint est un outil d'analyse du CSS coté Front-End.  
L'analyse nous remonte ici, des problèmes d'indentations.  
Il s'agit cependant d'erreurs minimes.

## Conclusion de l'analyse avec les outils d'analyse du code :

Ces différents outils d'analyses revelent tous des erreurs d'écriture. En ce qui nous concerne, l'accent doit être mis sur la mise à jour du code PHP. L'analyseur PHPStan a mis en relief l'absence de typage, cela devra être implementé, ainsi que les standards d'écriture disponible depuis PHP8 comme par exemple indiquer le type de retour d'une méthode. Les erreurs remontées par les autres analyseurs s'inscrivent davantage dans la forme du code : indentation, respect des espaces, respect de la ligne vierge à la fin d'une classe PHP...

## Analyses utilisant les outils d'analyse du rendu html:

Il existe en ligne de nombreux outils d'analyse du code HTML. Certains de ces outils poussent l'analyse afin d'intégrer une ébauche critique sur le rendu du CSS, sur le design. Nous allons ici en utiliser 3 : AChecker (<https://achecker.achecks.ca/checker/index.php>) est un outil directement disponible à son adresse url. W3C MarkUp Validation Service (<https://validator.w3.org/>) est également un outil directement disponible à son adresse url. Wave Web Accessibility Evaluation Tool est une extension Chrome à rajouter. Nous ne partageons que les analyses trouvant des problèmes.

### Analyse AChecker :

Problèmes trouvés sur le path : /users

The screenshot shows the AChecker interface with the following details:

- Accessibility Review** tab is selected.
- Accessibility Review (Guidelines: WCAG 2.0 (Level AA))**
- Known Problems(2)**, **Likely Problems (0)**, **Potential Problems (35)**, **HTML Validation (0)**, **CSS Validation**.
- 1.3 Adaptable: Create content that can be presented in different ways (for example simpler layout) without losing information or structure.**
- Success Criteria 1.3.1 Info and Relationships (A)**
- Check 245: Data table with more than one row/column of headers does not use id and headers attributes to identify cells.**
- Repair:** Add `id` and `headers` attributes to table cells so they identify the cells that relate to the headers.
- Line 60, Column 13:**

```
<table class="table">
    <thead>
        <tr>
            <th>#</th> ...
```
- Check 244: Data table with both row and column headers does not use scope to identify cells.**
- Repair:** Add `scope` attributes to header cells so they identify the cells that relate to the header.
- Line 60, Column 13:**

```
<table class="table">
    <thead>
        <tr>
            <th>#</th> ...
```

### Analyse de W3C MarkUp Validation :

Problème trouvé sur tous les paths :

**Warning** The `navigation` role is unnecessary for element `nav`.  
 From line 28, column 9; to line 28, column 111  
`>--<nav class="navbar navbar-light navbar-fixed-top" style="background-color: #e3f2fd;" role="navigation">`

## Problème trouvé sur le path : /tasks

**Warning** The `navigation` role is unnecessary for element `nav`.  
 From line 29, column 9; to line 29, column 111  
`>--<nav class="navbar navbar-light navbar-fixed-top" style="background-color: #e3f2fd;" role="navigation">`

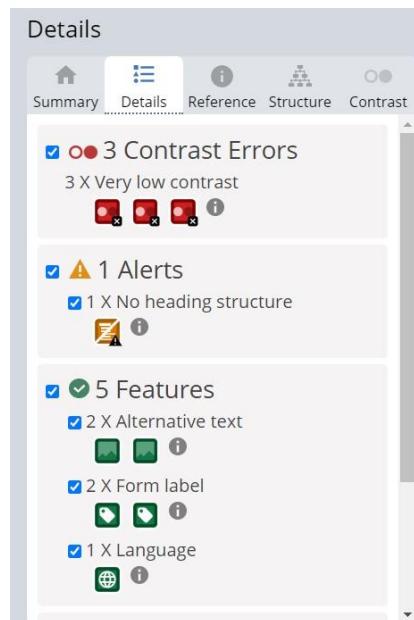
**Warning** Empty heading.  
 From line 64, column 21; to line 64, column 43  
`<h4 class="pull-right">`

**Warning** Empty heading.  
 From line 83, column 21; to line 83, column 43  
`<h4 class="pull-right">`

**Warning** Empty heading.  
 From line 102, column 21; to line 102, column 43  
`<h4 class="pull-right">`

## Analyse de Wave Web Accessibility Evaluation Tool :

Problèmes trouvés sur les path : /tasks/create et /tasks/{id}/edit



Problèmes trouvés sur le path : /tasks

**Details**

Summary Details Reference Structure Contrast

- 3 Errors**
  - 3 X Empty heading
    - h-
    - h-
    - h-
    - i
- 9 Contrast Errors**
  - 9 X Very low contrast
    - 
    - 
    - 
    - 
    - 
    - 
    - 
    - 
    -
- 1 Alerts**
  - 1 X Missing first level heading
    - h1
- 3 Features**
  - 2 X Alternative text
    - 
    - 
    -

Problèmes trouvés sur les path : /users/create et /users/{id}/edit

**Details**

Summary Details Reference Structure Contrast

- 3 Contrast Errors**
  - 3 X Very low contrast
    - 
    - 
    - 
    -
- 6 Features**
  - 1 X Alternative text
    - 
    -
  - 4 X Form label
    - 
    - 
    - 
    - 
    -
  - 1 X Language
    - 
    -
- 3 Structural Elements**
  - 1 X Heading level 1
    - h1
  - 1 X Navigation

Problèmes trouvés sur les path : /login

The screenshot shows the 'Details' panel of the WAVE tool. It lists three categories of findings:

- Contrast Errors:** 2 items, both marked as 'Very low contrast'. Icons: red square with a black 'X'.
- Alerts:** 1 item, marked as 'No heading structure'. Icon: yellow triangle with an exclamation mark.
- Features:** 5 items, all marked as valid:
  - 2 X Alternative text (green square icon)
  - 2 X Form label (hand icon)
  - 1 X Language (globe icon)

Problèmes trouvés sur les path : /users

The screenshot shows the 'Details' panel of the WAVE tool for the '/users' path. It lists several categories of findings:

- Contrast Errors:** 6 items, all marked as 'Very low contrast'. Icons: red square with a black 'X'.
- Features:** 2 items, both marked as valid:
  - 1 X Alternative text (green square icon)
  - 1 X Language (globe icon)
- Structural Elements:** 12 items, including:
  - 1 X Data table (blue circle icon)
  - 4 X Table header cell (blue square icon)
  - 4 X Row header cell (blue square icon)

Conclusion des analyses en ligne :

Les outils en ligne d'analyses mettent en avant des erreurs minimes mais recurrentes dans le code HTML. On veillera à les corriger.

## Analyse visuelle du développeur :

Plusieurs erreurs ont été notées à la lecture du code pouvant nuire à la qualité du code mais pouvant également créer d'éventuelles failles de sécurité et des bugs de fonctionnement. On verra premierement les failles majeures de sécurité, puis les améliorations possibles pour suivre les principes SOLID et les erreurs graphiques.

### Failles majeures :

Premierement comme on l'avait souligné lors de l'analyse Blackfire, les pages Users/, Users/Create, Users/{id}/Edit sont ouvert à tous, or seul Users/Create peut être accessible à tout le monde. La page Users/ peut être accessible on le souhaite à l'administrateur et le Users/{id}/Edit ne doit être accessible en théorie qu'à l'utilisateur connecté également titulaire de cet id.

Ensuite la classe User est incomplète, ainsi on utilise l'username pour se connecter mais il n'y a pas d'UniqueEntity dessus, il faut donc soit le rajouter soit utiliser l'email pour se connecter. La validation côté back-end a été baclée ainsi dans le code on pourrait enregistrer un utilisateur sans mot de passe, en réalité la validation côté front-end l'empeche, mais ce n'est pas suffisant car on peut la contourner et finalement on peut enregistrer un utilisateur avec un mot de passe d'un seul caractère...

Une faille de sécurité importante est également l'absence de token lors de la connexion : le site est donc vulnérable à la faille CSRF.

De même côté code, on a remarqué l'absence de vérification de la soumission d'un formulaire, on traite le formulaire seulement si il est valide alors qu'il faut le traiter si il a été soumis et si il est valide.

### Respect des principes SOLID :

Il est fortement conseillé de suivre les principes dits «SOLID» afin de garantir la qualité du code et permettre à l'application de se perenniser dans le temps. Dans ce projet, nous avons relevé deux améliorations possibles.

La première correspond à l'architecture même du projet : le code se concentre dans les classes Controller alors que celles-ci devraient en théorie être épurées.

Pour cela, nous recommandons de créer des classes Repository et des classes Manager afin de satisfaire le principe d'une seule responsabilité par fonction.

Concrètement lors de la création d'une tâche, on a ce code dans le taskController :

```
if ($form->isValid()) {
    $em = $this->getDoctrine()->getManager();
    $em->persist($task);
    $em->flush();
```

Ce type d'opérations ne devraient pas figurer dans un Controller mais dans une fonction create() ( ou new()) d'un TaskManager.

La seconde amélioration concerne le principe d'inversion de dépendances selon lequel les classes de hauts niveaux ne devraient pas dépendre des classes de bas niveaux mais d'abstractions.

Concrètement lors de la création d'un utilisateur, on a ce code dans le UserController ( qui devrait d'ailleurs se situer dans un manager ) :

```
if ($form->isValid()) {
    $em = $this->getDoctrine()->getManager();
    $password = $this->get('security.password_encoder')->encodePassword($user,
$user->getPassword());
    $user->setPassword($password);
```

On va donc chercher d'encodeur pour encoder le mot de passe. Mais que se passe-t-il si un jour, on souhaite changer d'encodeur ? Il faut alors changer le code. On préférera alors utiliser une abstraction comme UserPasswordHasherInterface comme cela si un jour on décide de changer d'encodeur on aura juste à implémenter cette interface sur notre nouveau encodeur sans changer notre code.

Un autre principe à respecter est la nomenclature des méthodes, il est ainsi inutile de rappeler le nom de la classe dans ses propres méthodes, si on va dans le TaskController, on a : listTask, createAction, editAction, toggleTaskAction et deleteTaskAction. Beaucoup de répétitions pour peu de chose : create, edit et delete sont de toutes façons des actions, de plus nous sommes dans le TaskController qui s'occupent de l'entité Task : on peut donc les renommer : list, create, edit, toggle et delete.

### Erreurs graphiques :

On ne rappellera pas les erreurs d'écritures relevées par les analyses précédentes et l'on va uniquement se concentrer sur les erreurs visuelles de design et d'expérience utilisateur concrètes.

Sur la page d'accueil (/login ), le bouton «Creer un utilisateur» n'est ni ajuster ni à sa place, il devrait se situer sous la section «Se Connecter» ( qui devrait être elle aussi remise en forme et aérer ) avec un scénario de lecture logique comme : « Se connecter , par encore inscrit ? Creer un compte».

Sur la page de création d'un utilisateur (/users/create), le bouton «Créer un utilisateur» doit être enlevé. En effet, à sa soumission, on est réorienté sur ... la même page. Le bouton de soumission gagnerais à s'appeler «Créer mon compte», en effet du point de vue de l'admin il s'agit d'ajouter un utilisateur mais du point de vue de l'utilisateur il s'agit avant tout de se créer un compte.

Comme on l'a précisé la page de consultation de la liste des utilisateurs (/users) ne devrait être accessible qu'à l'admin c'est-à-dire à un utilisateur connecté : on peut donc enlever les boutons «Créer un utilisateur» et «Se Connecter». Même remarque pour la page d'édition d'un utilisateur (/users/{id}/edit).

Lorsqu'on se connecte on remarque encore une fois le bouton «Créer un utilisateur». Ainsi il faudra supprimer ce bouton sur toutes les pages connectées. Le lien «Se deconnecter» gagnerais à être dans la barre de menu.

Au moment de la connection on arrive sur une page d'accueil (/). On a 3 boutons : «Creer une nouvelle tâche», «Consulter la liste des tâches à faire» et «Consulter la liste des tâches terminées». Ici plusieurs problèmes se posent : premierement visuellement il n'y a aucune mise en forme, deuxiemement une grosse incohérence du point de vue même du design des boutons : les deux premiers ressemblent à de vrais boutons alors que le troisième ressemble à un lien. Troisiemement, une incohérence fonctionnelle : le bouton «Consulter la liste des tâches à faire» ne renvoie pas du tout à la liste des tâches à faire mais à la liste de

toutes les tâches ( faite ou pas ) alors que le bouton «Consulter la liste des tâches terminées» ne renvoie à rien du tout ... ( faux lien ). Du point de vue de l'expérience utilisateur cette page d'accueil est inutile, autant mettre la liste des toutes les tâches faites ou non, et renvoyés les boutons à leurs rôles : créer une tâche, consulter les tâches faites, consulter les tâches non faites.

Lorsqu'on arrive sur la liste des tâches ( /tasks ) ( donc toutes les tâches faites ou non ), on remarque une problème de mise en forme dû au bouton «Créer une tâche», en effet on a 3 tâches par rang, mais la troisième tâche se décale verticalement et rompt l'alignement avec les autres à cause du bouton. Le design des tâches va à l'encontre du bon sens : la symbole croix (x) signifie en général «supprimer» mais pas ici, ici il signifie que la tâche n'est pas terminée : l'utilisateur a tendance à essayer de cliquer dedans alors que ce n'est pas un lien. Au contraire il ne sait pas où cliquer pour éditer la tâche car le lien se cache dans le nom de la tâche. Habituellement le nom de la tâche peut comporter un lien pour renommer la tâche, mais l'édition de la tâche possède un bouton «Edit» voire encore mieux une icone ( stylo pour éditer , corbeiller ou croix pour supprimer ).

---

## Conclusion de l'analyse de qualité :

Les multiples analyses ont mises au jour des erreurs d'écriture dans le code à la fois côté back-end et côté front-end. Cependant c'est davantage l'analyse du développeur qui a permis de découvrir des graves erreurs de sécurité, des failles importantes pouvant entraîner des bugs et le non respect de certains principes SOLID côté back-end, ainsi que des incohérences, des erreurs graphiques et des erreurs mises en forme côté front-end. Toutes les erreurs back-end doivent être impérativement comprises et corrigées afin de proposer un site fonctionnel et sécurisé.

## Partie 3 : Améliorations apportées

---

### Introduction :

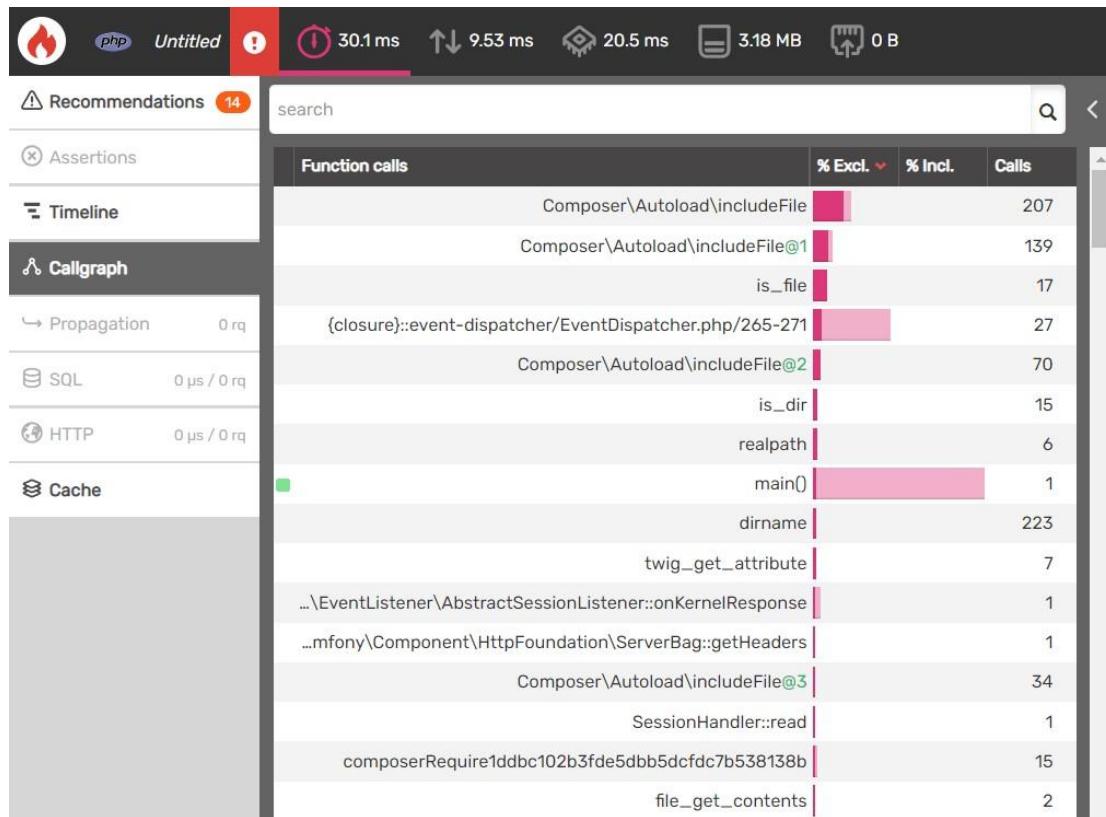
Dans le cadre de ce projet, il a été spécifié que le développeur n'avait pas l'obligation de corriger les erreurs relevées dans l'audit mais que la correction si elle avait lieu serait appréciée. C'est en gardant ces considérations à l'esprit que le développeur a choisi de reprendre et d'améliorer les points qui lui ont semblé les plus importants : correction des erreurs majeures de fonctionnement, implementation et respect des normes PSR ( PHP Standard Recommandation ) et des principes SOLID, implementation des tests unitaires et fonctionnels, et améliorations de la performance. Le côté Front-end et l'aspect visuel du site a été revue de manière très basique afin de se concentrer sur les tâches nommées ci-dessus.

Cette partie comprend le comparatif de l'analyse de performance de blackfire, l'implementation des tests avec PHPUnit et le taux de couverture du code et les améliorations mises en oeuvre pour respecter les principes SOLID.

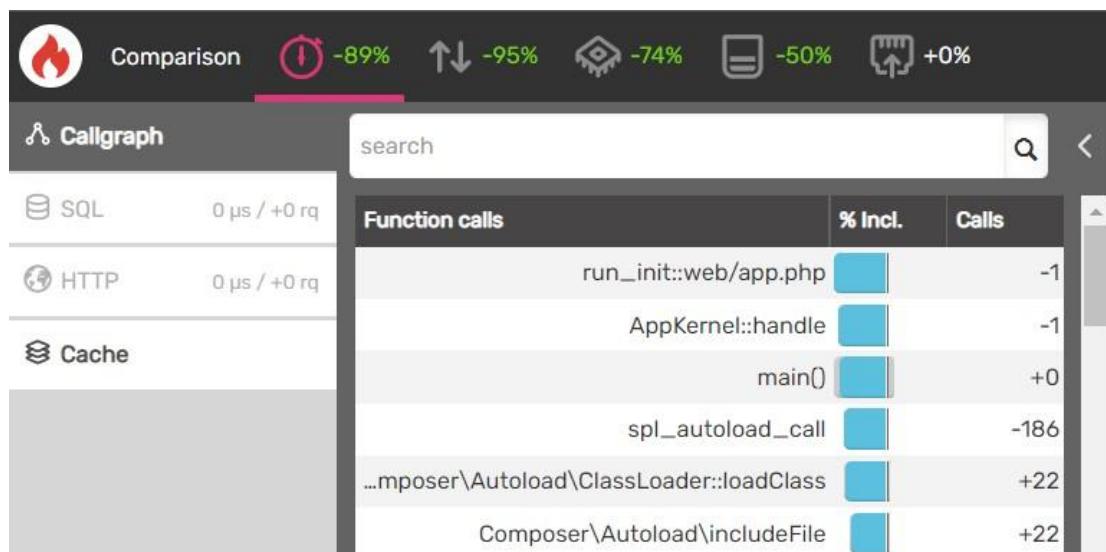
## Comparatif de l'analyse de performance Blackfire :

Dans cette partie nous allons montrer pour chaque requête analysée dans la partie 1, les améliorations apportées en terme de performance.

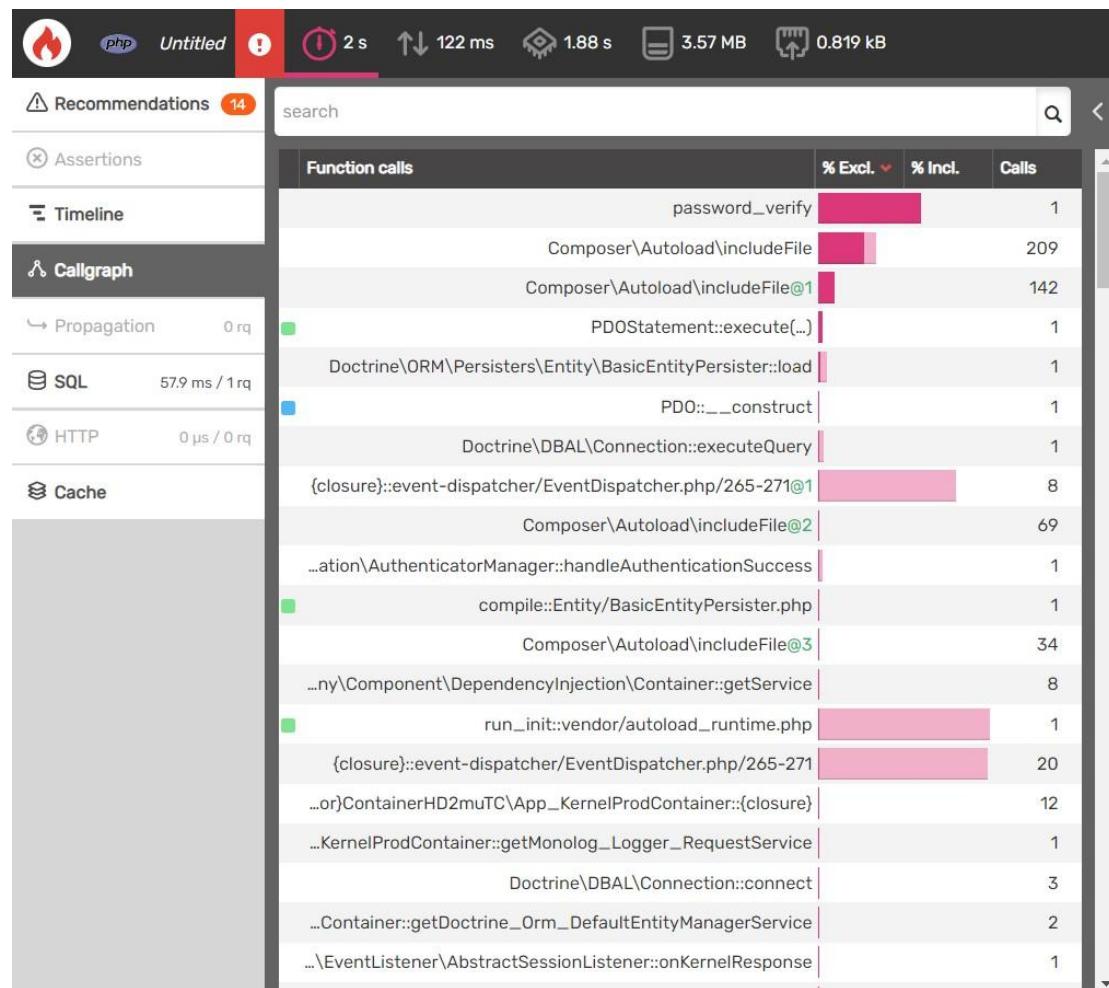
Statut 200, Method GET, Path /login :



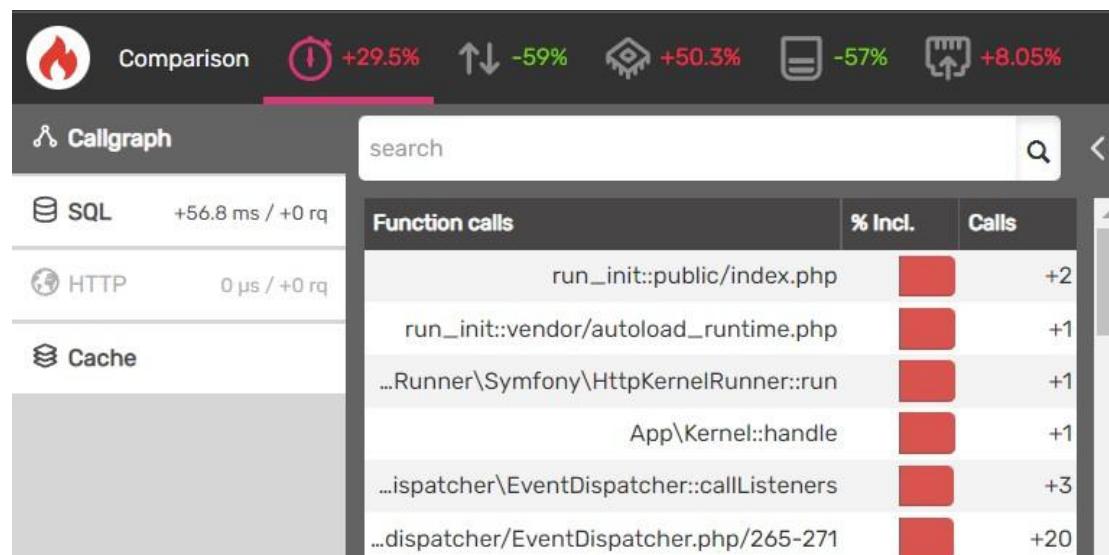
Comparaison avec le premier profil :



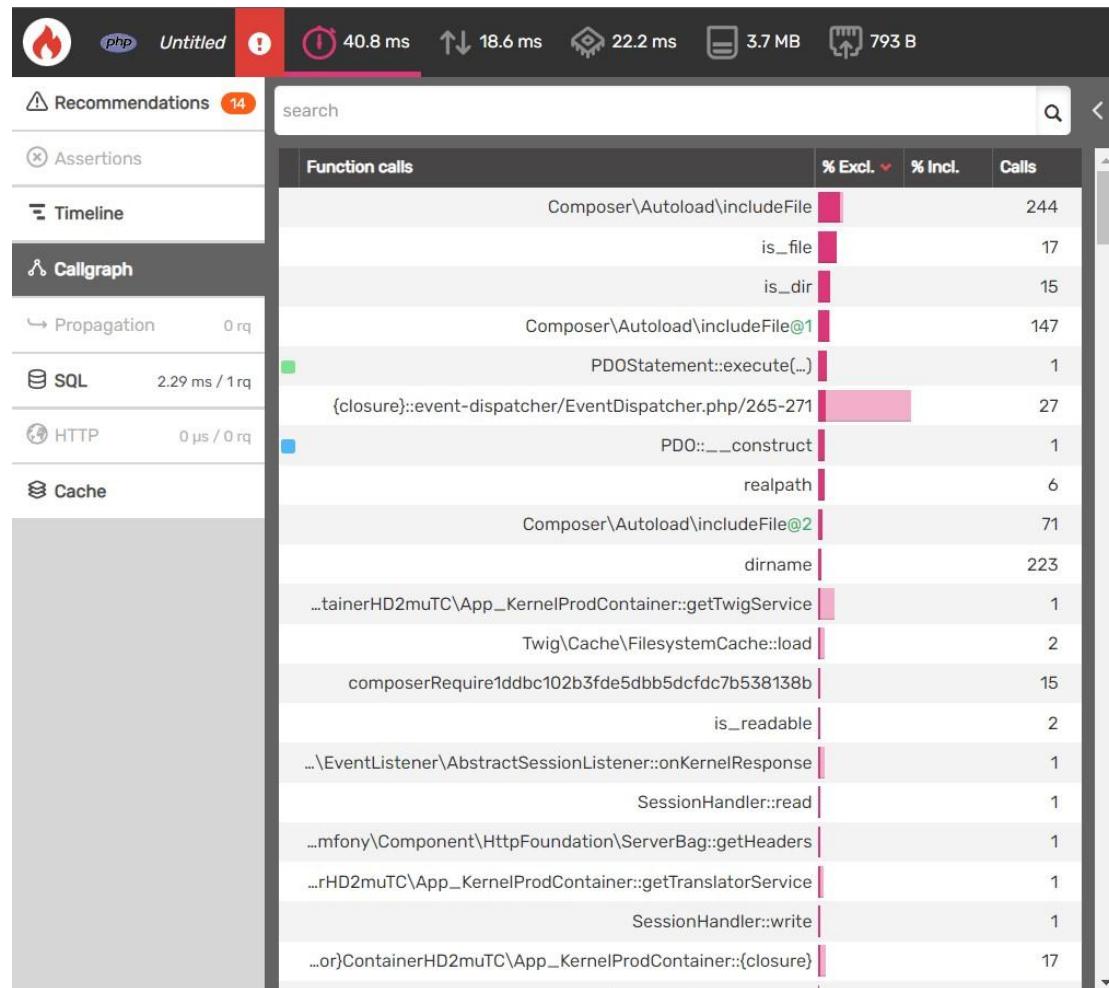
Statut 302, Method POST, Path /login :



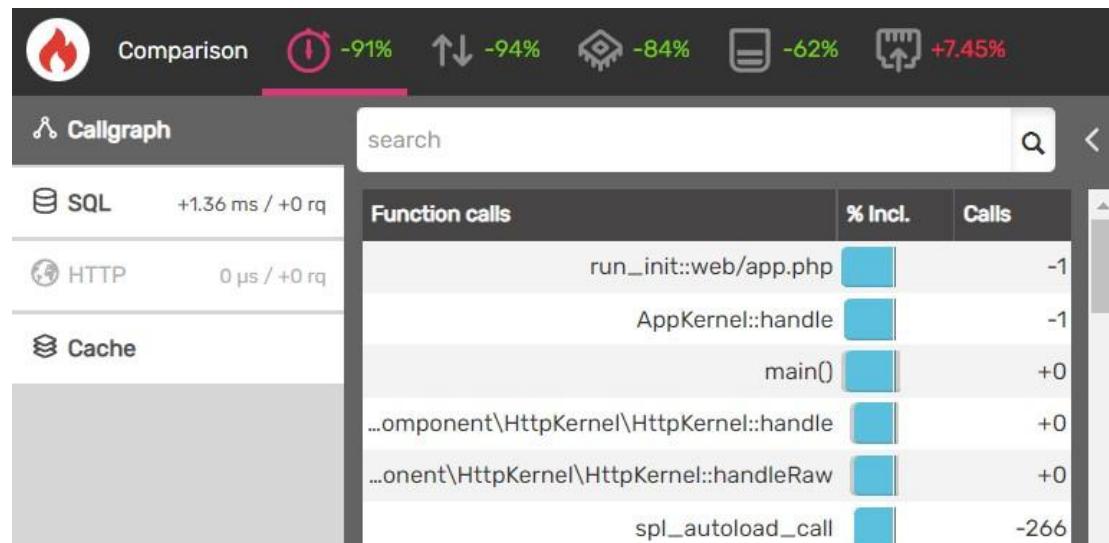
Comparaison avec le premier profil ( nom du path : /login\_check) :



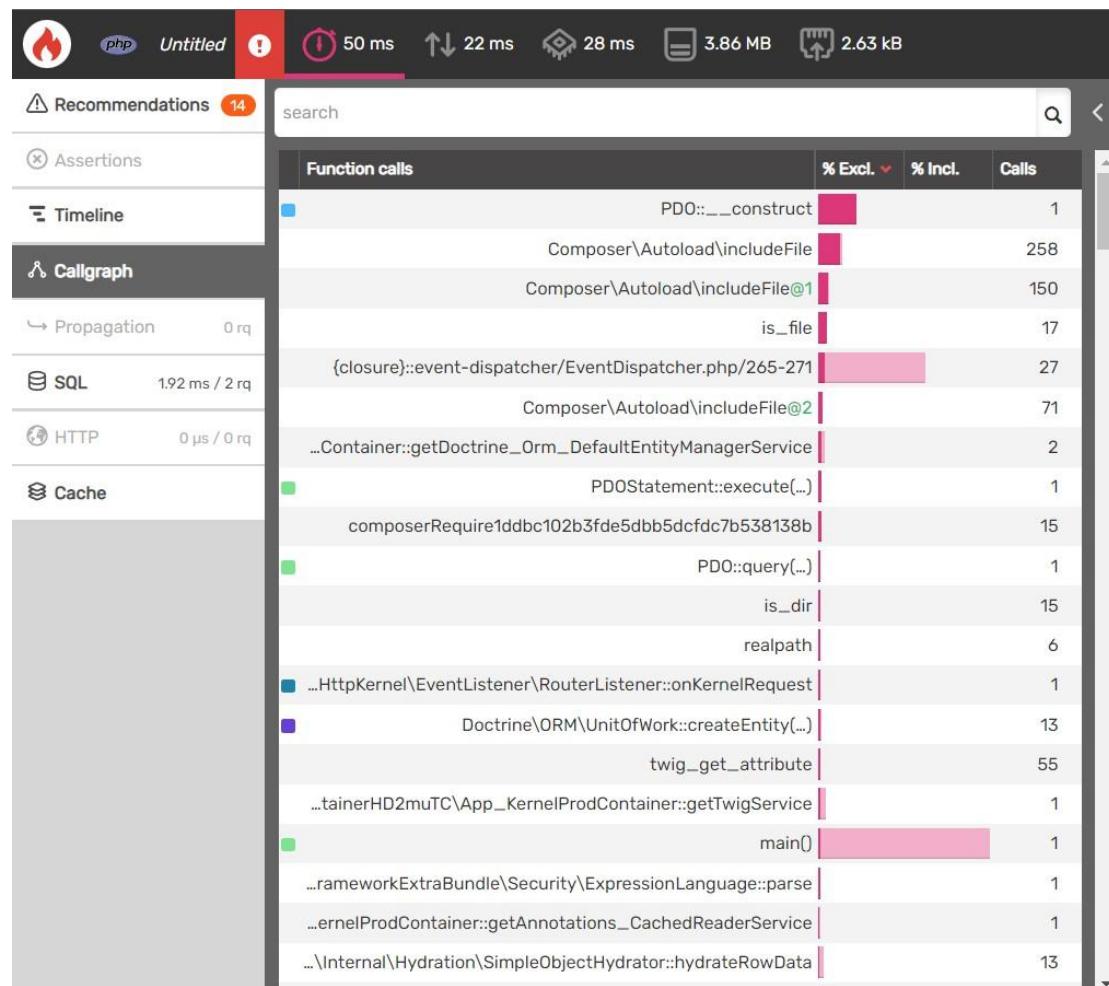
Statut 200, Method GET, Path / :



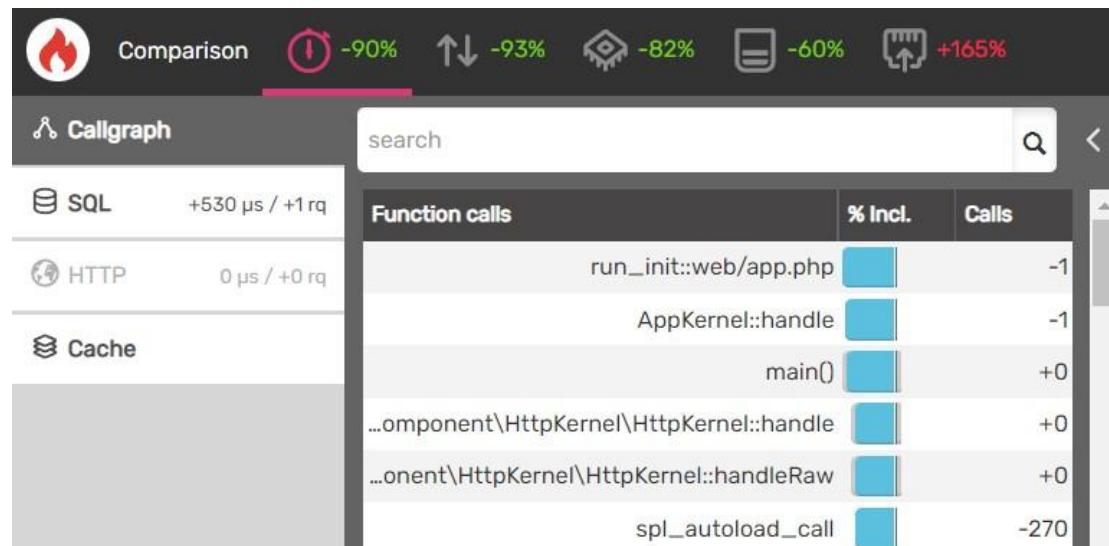
Comparaison avec le premier profil :



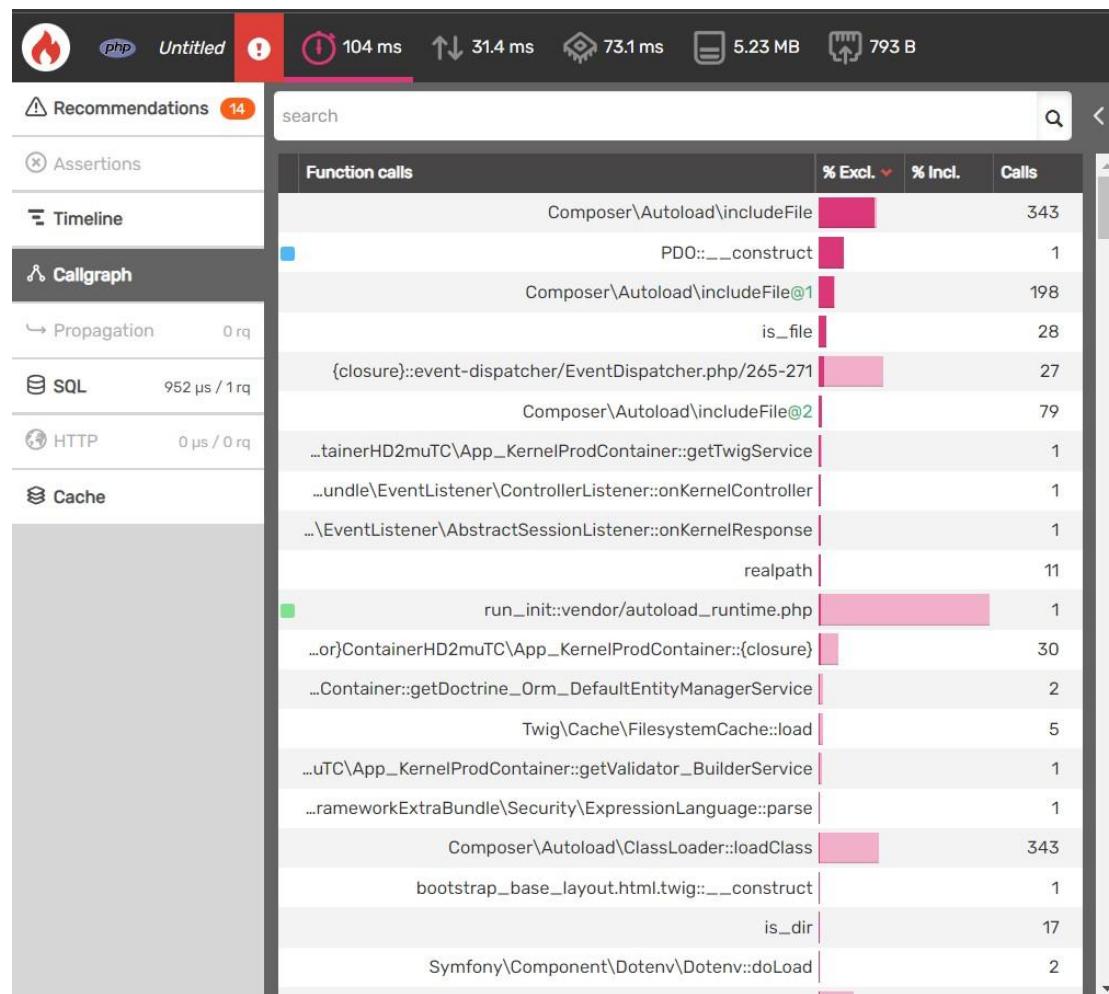
Statut 200, Method GET, Path /users :



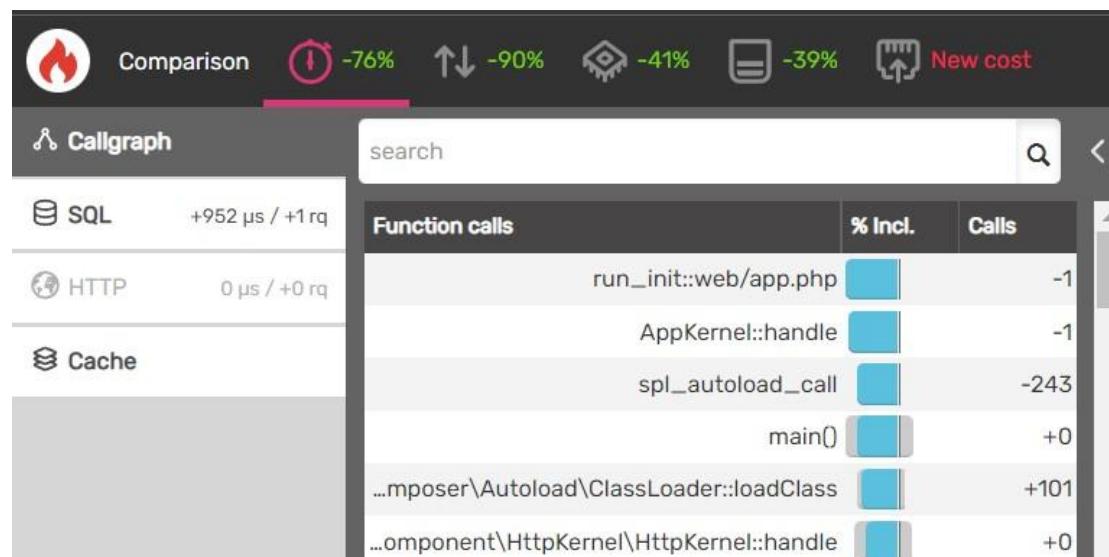
Comparaison avec le premier profil :



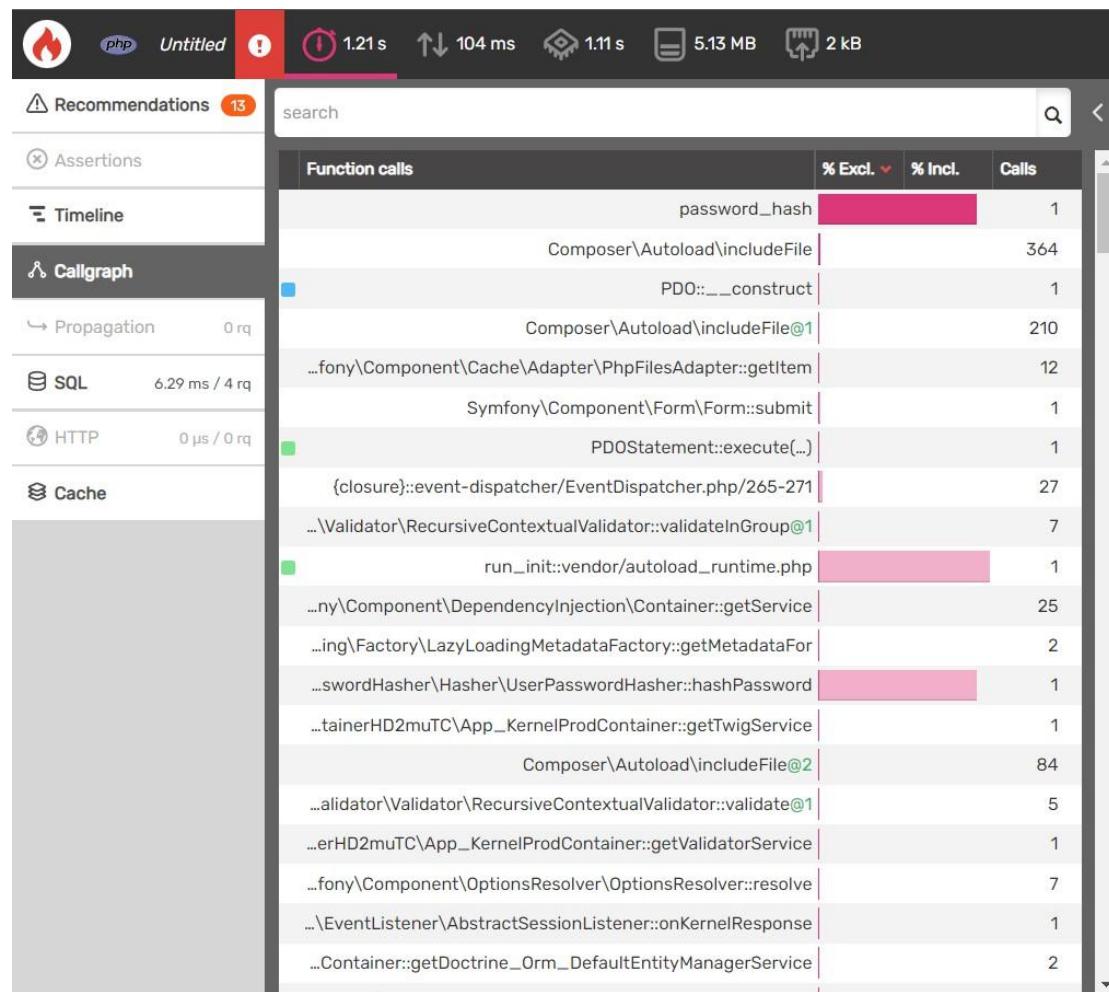
Statut 200, Method GET, Path /users/create :



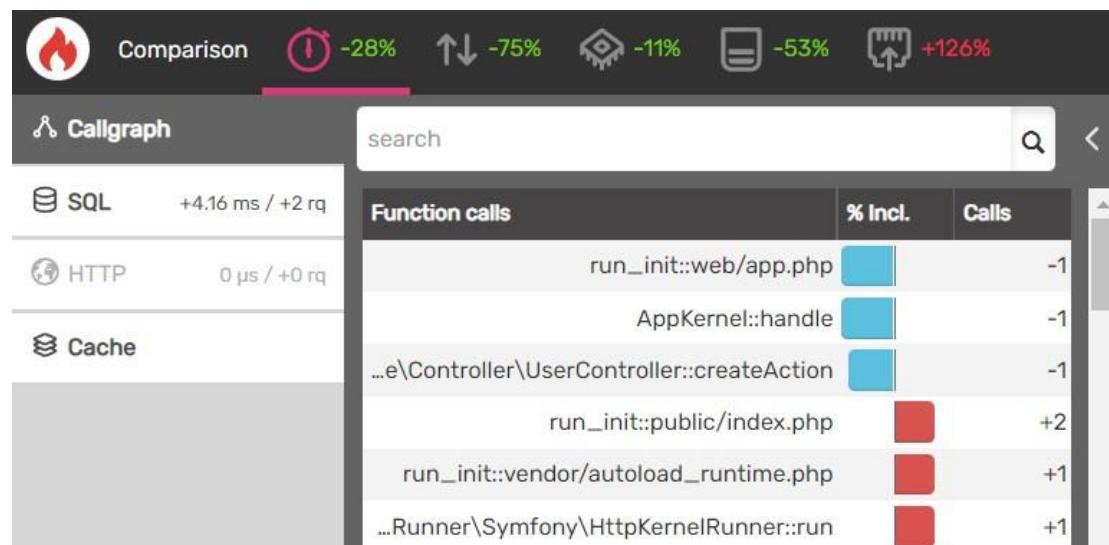
Comparaison avec le premier profil :



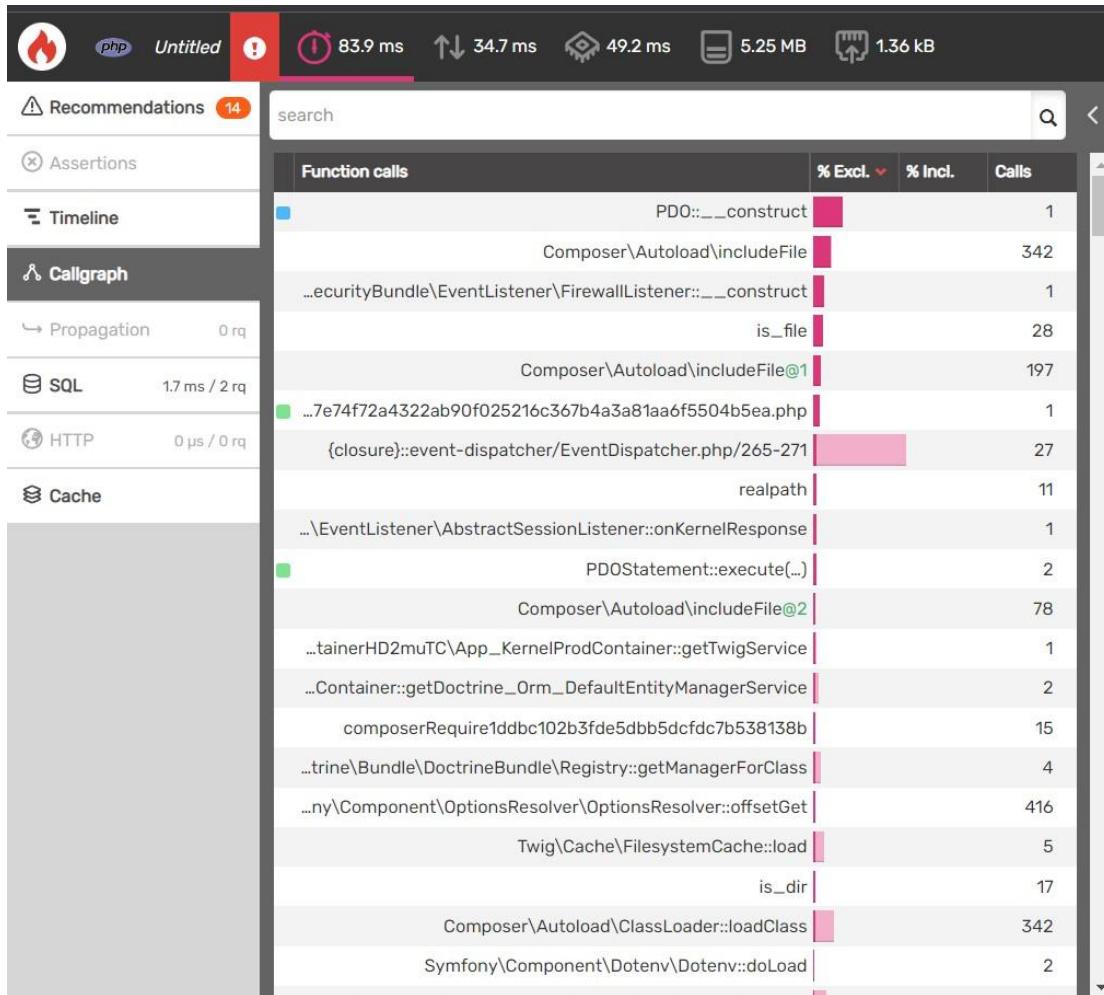
Statut 302, Method POST, Path /users/create :



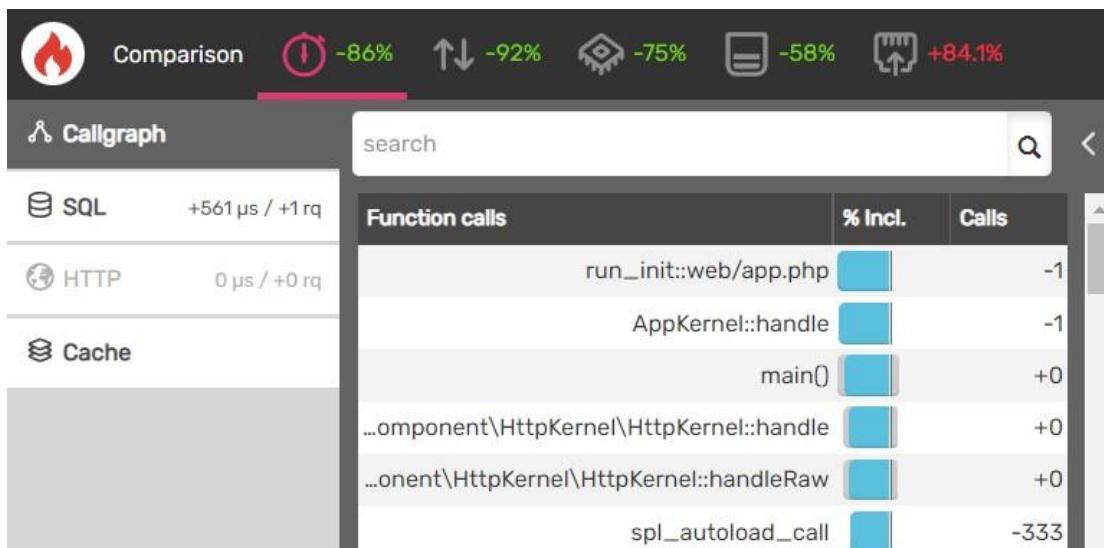
Comparaison avec le premier profil :



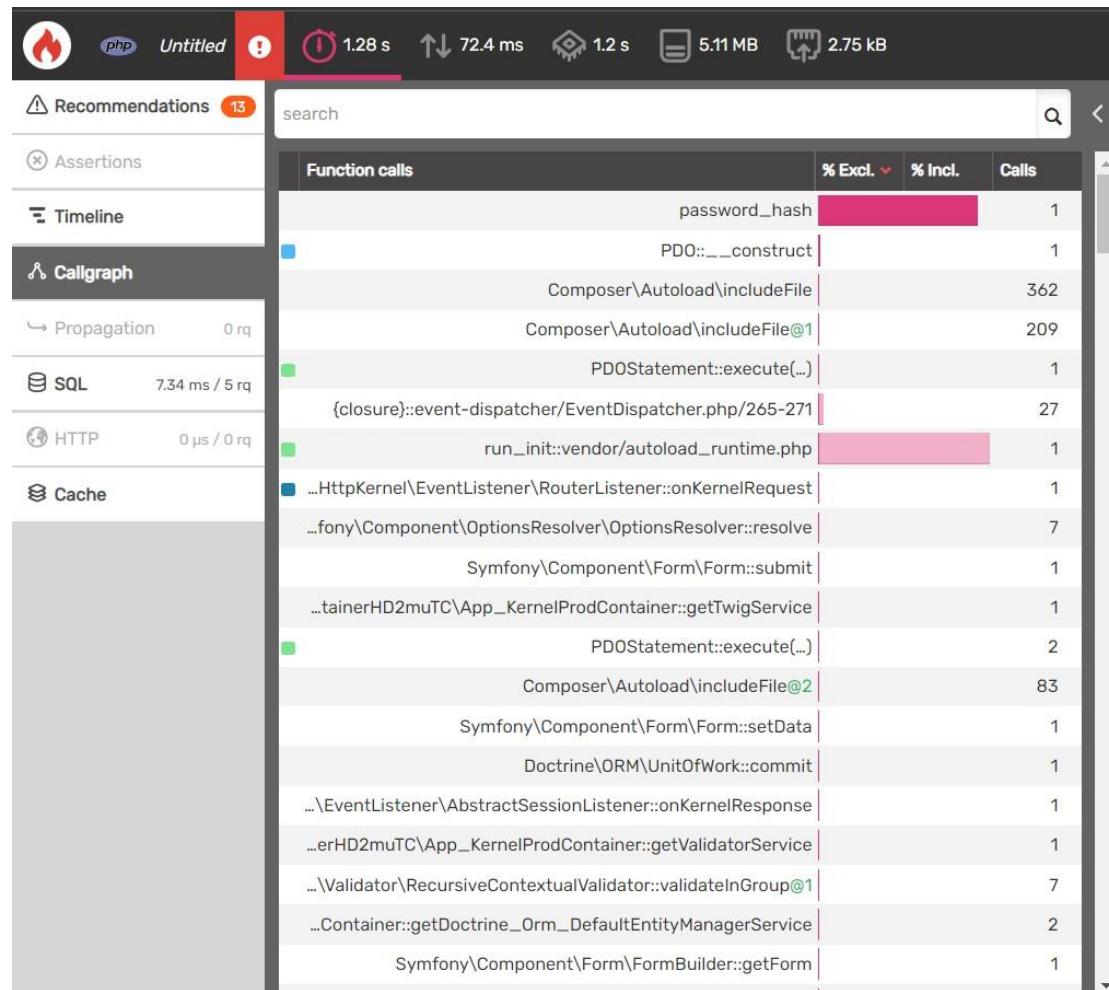
Statut 200, Method GET, Path /users/{id}/edit :



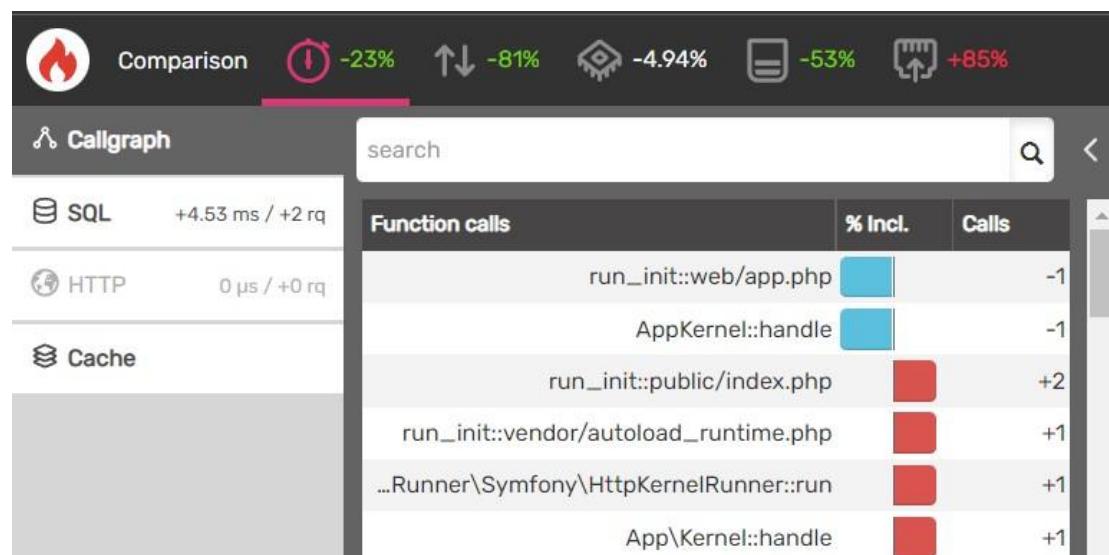
Comparaison avec le premier profil :



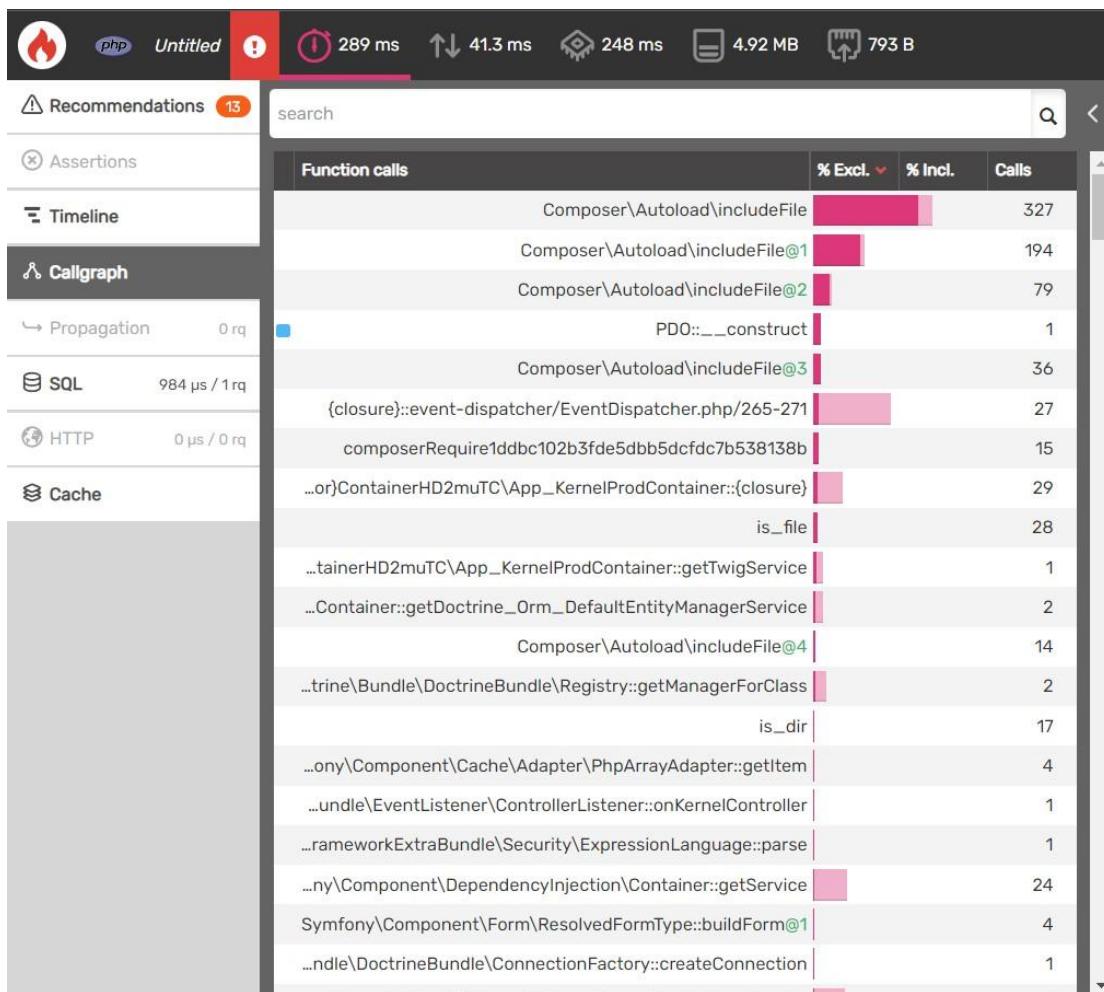
Statut 302, Method POST, Path /users/{id}/edit :



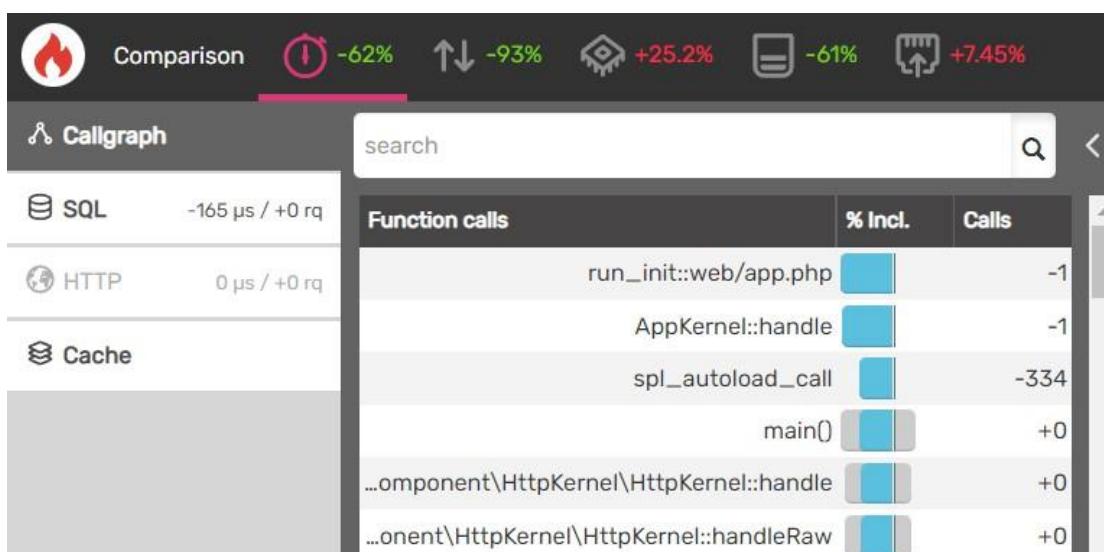
Comparaison avec le premier profil :



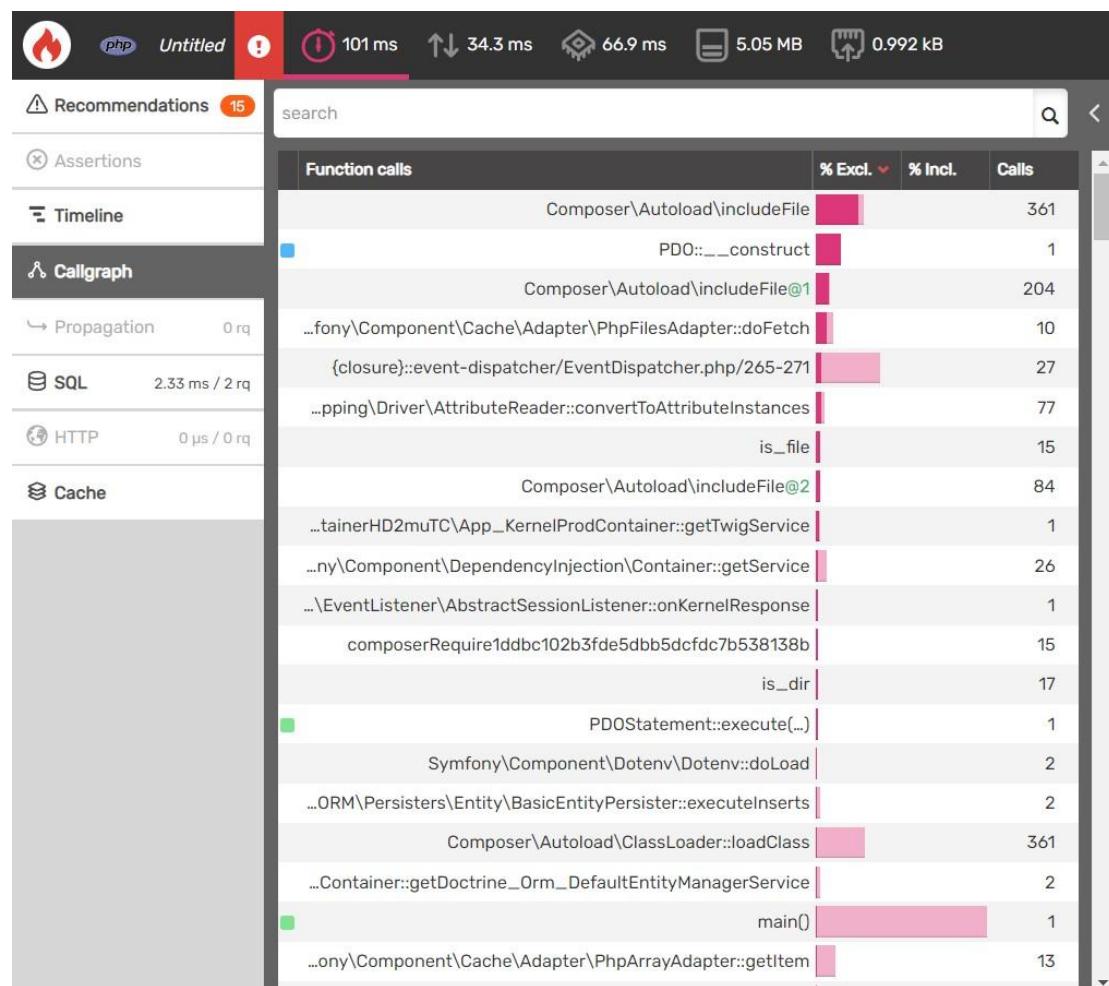
Statut 200, Method GET, Path /tasks/create :



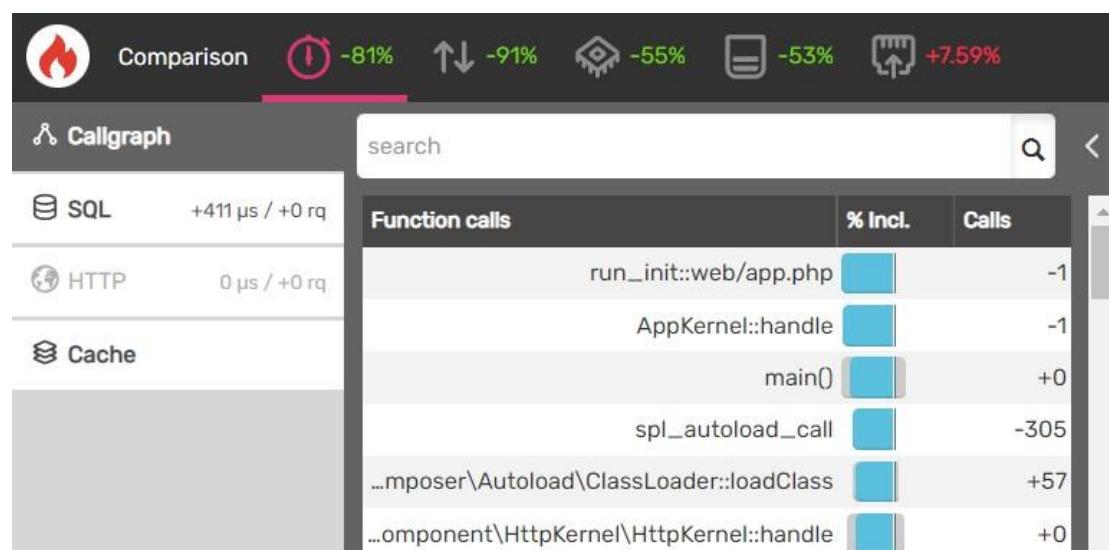
Comparaison avec le premier profil :



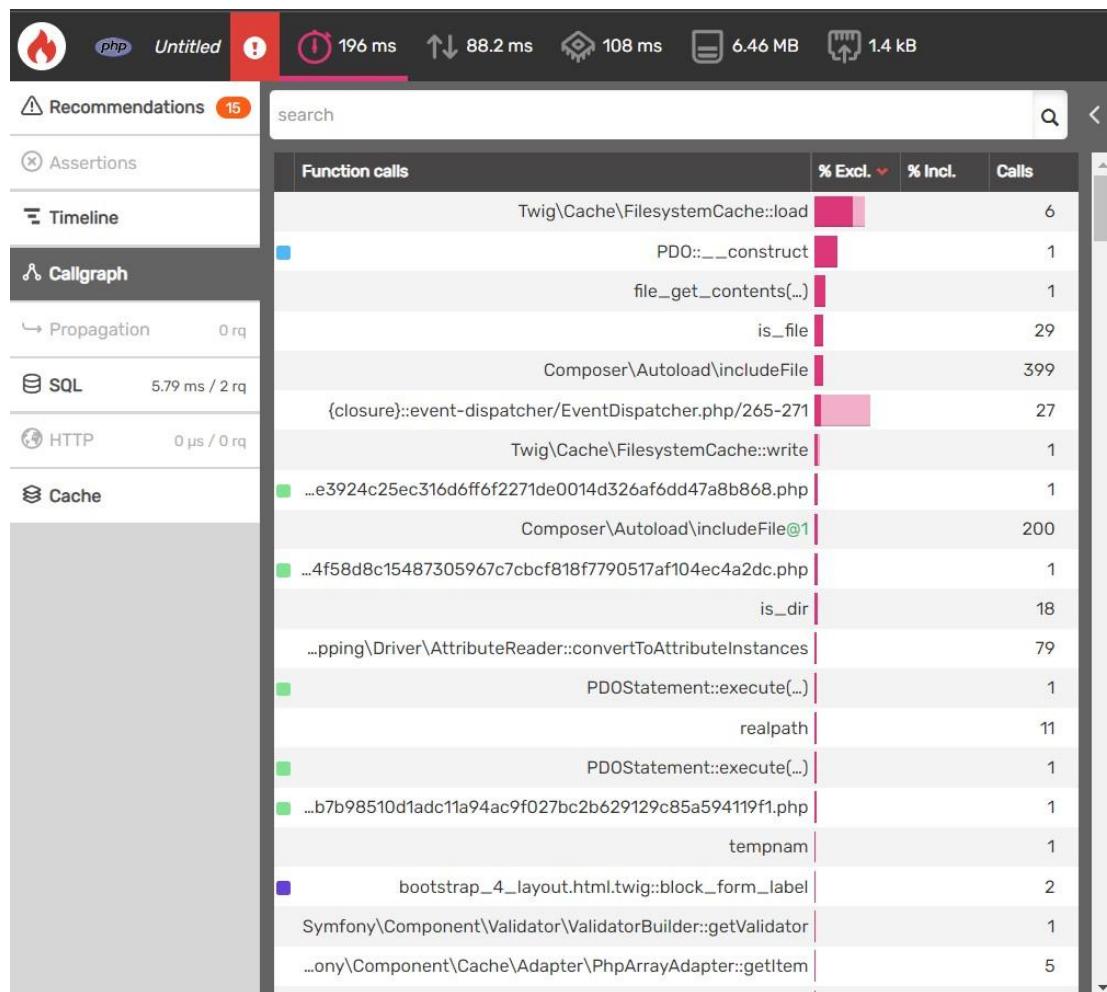
Statut 302, Method POST, Path /tasks/create :



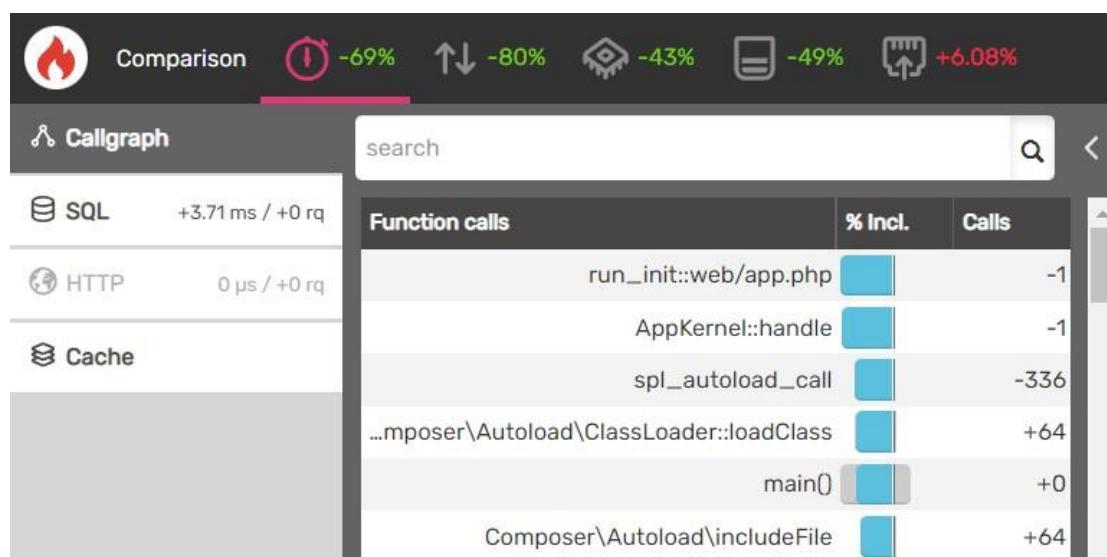
Comparaison avec le premier profil :



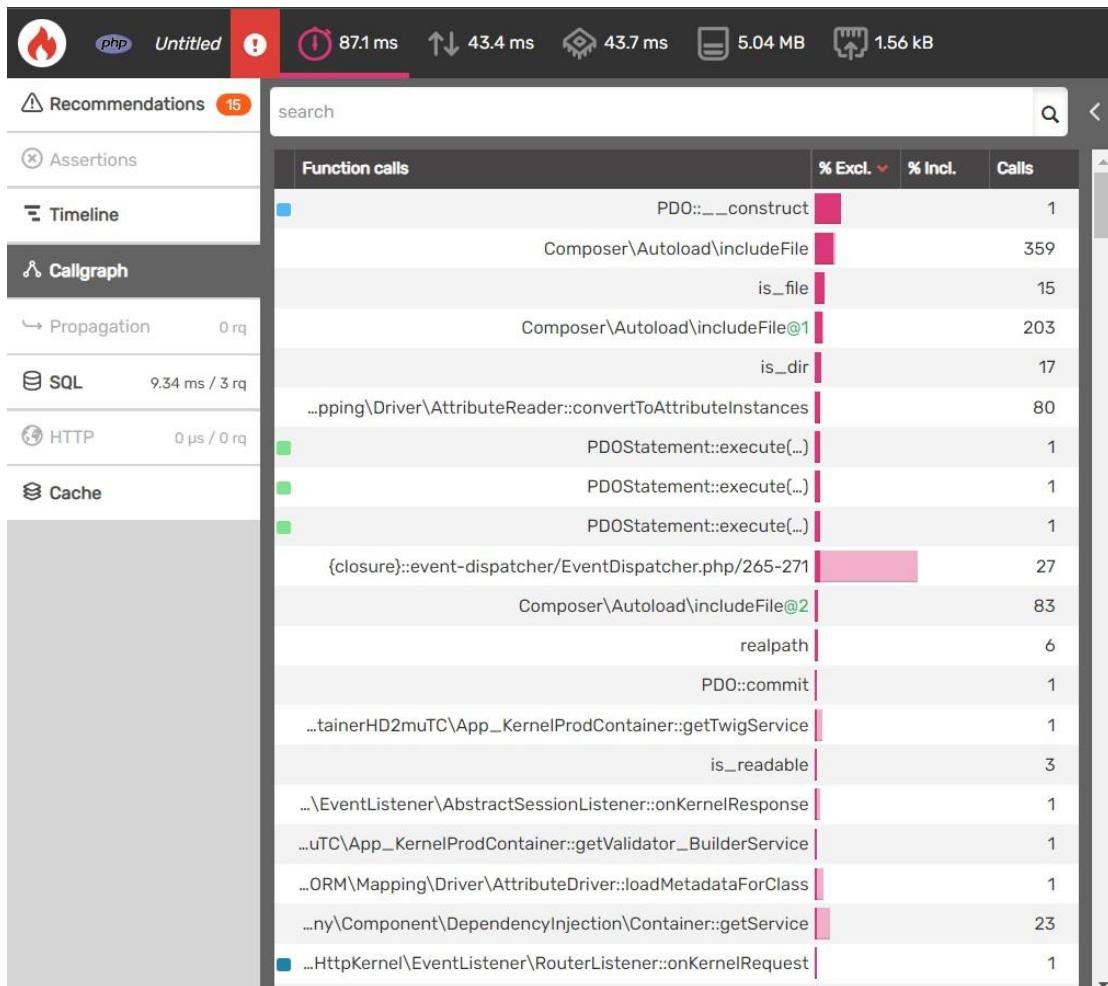
Statut 200, Method GET, Path /tasks/{id}/edit :



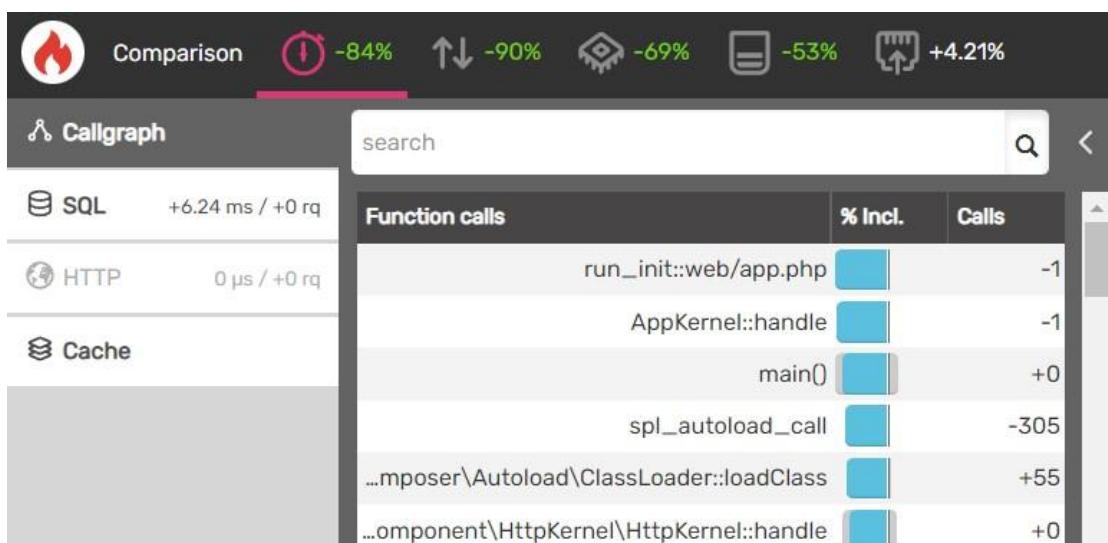
Comparaison avec le premier profil :



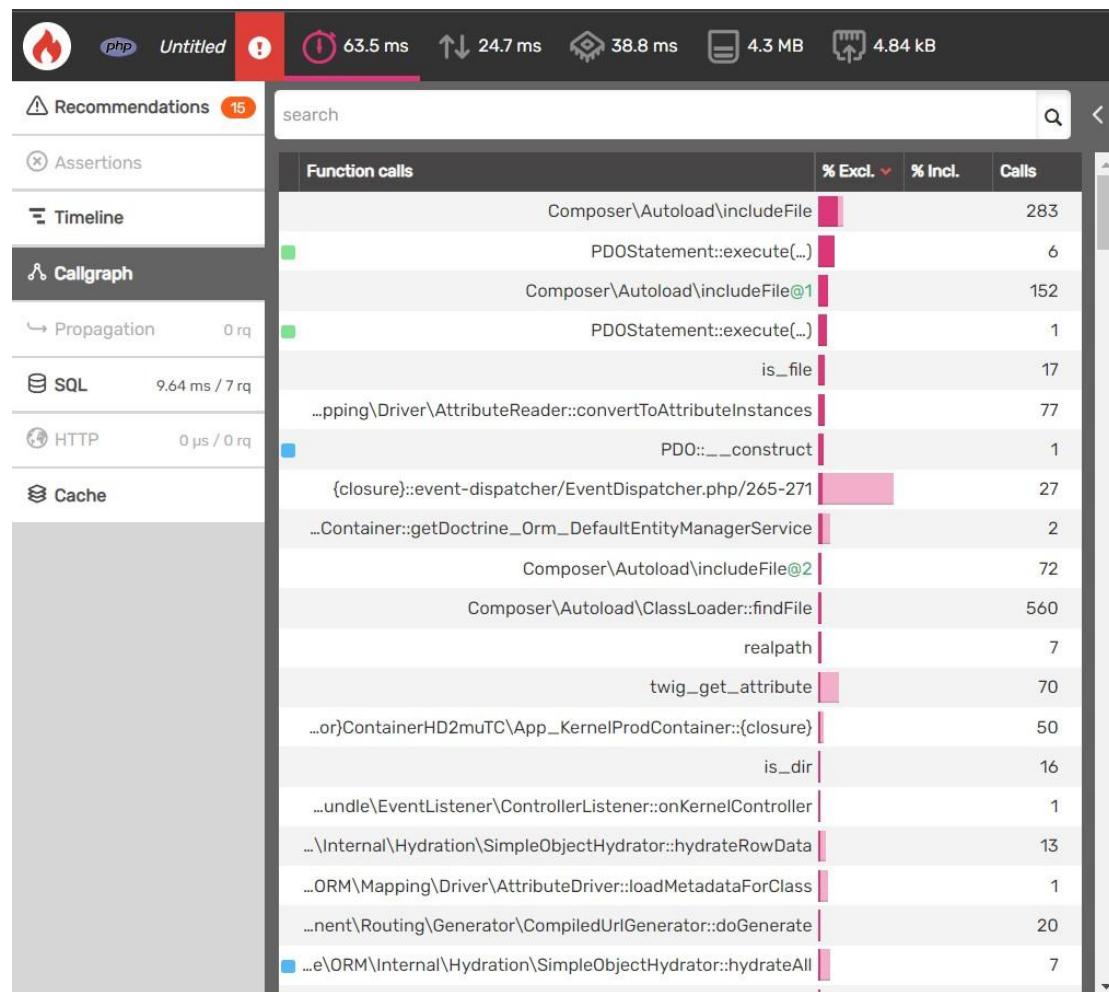
Statut 302, Method POST, Path /tasks/{id}/edit :



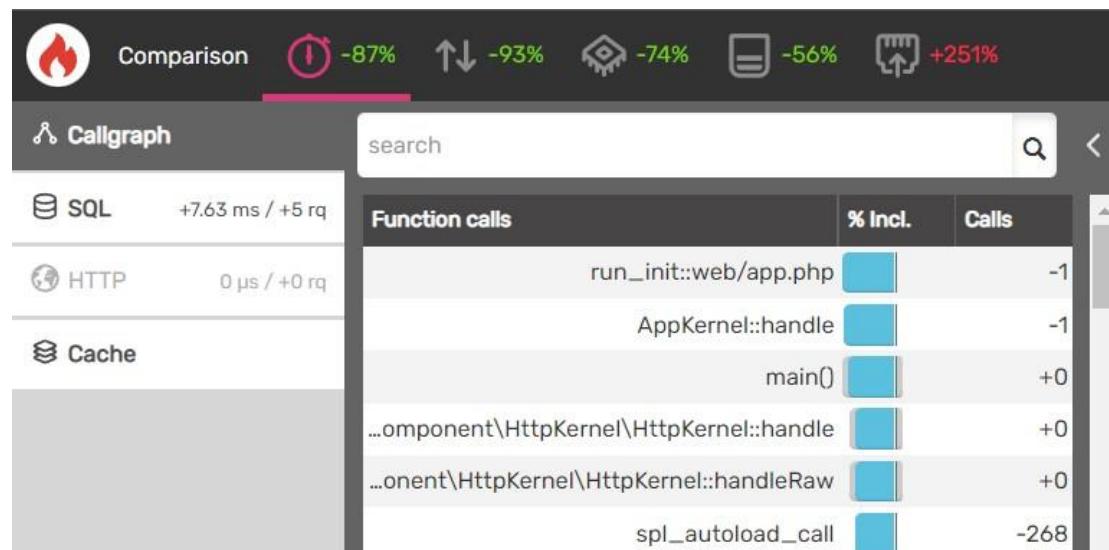
Comparaison avec le premier profil :



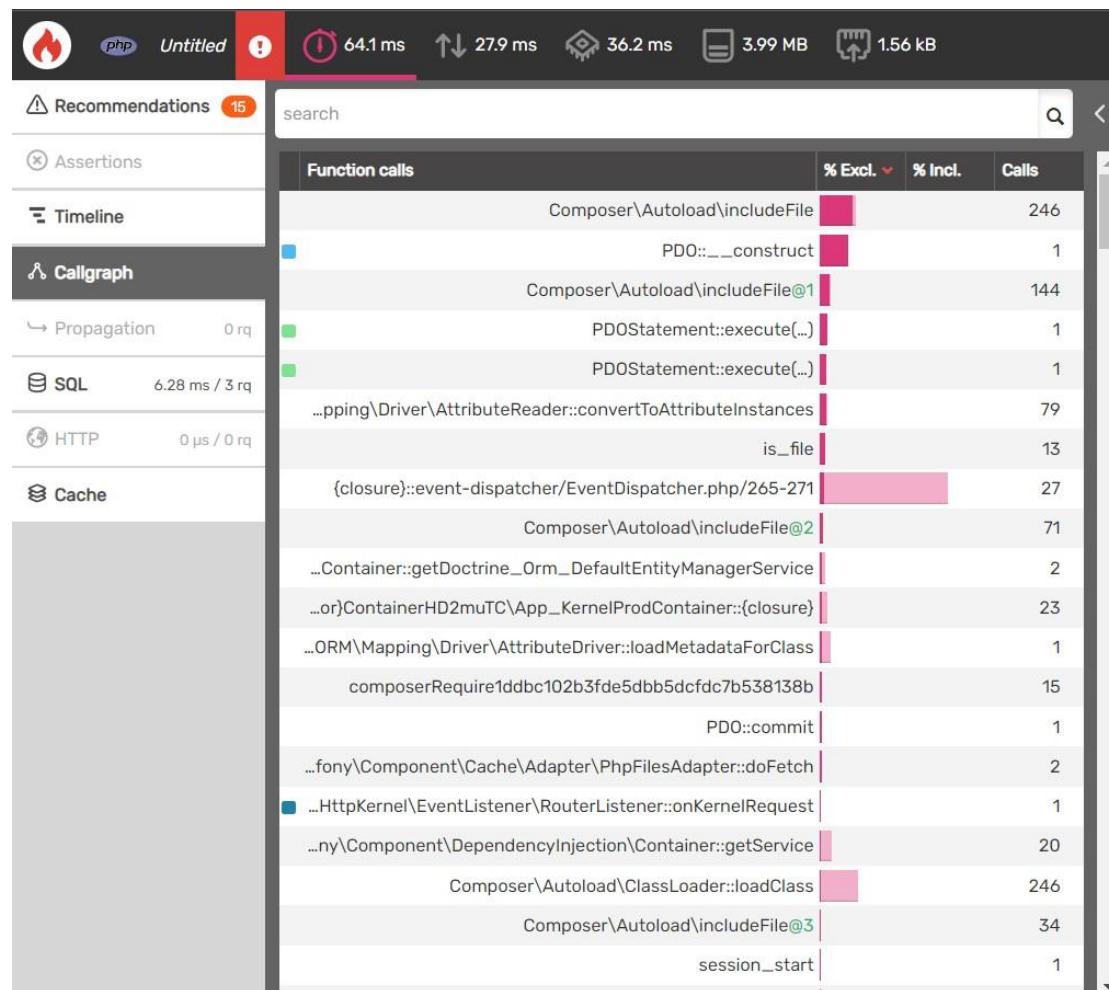
Statut 200, Method GET, Path /tasks/undone :



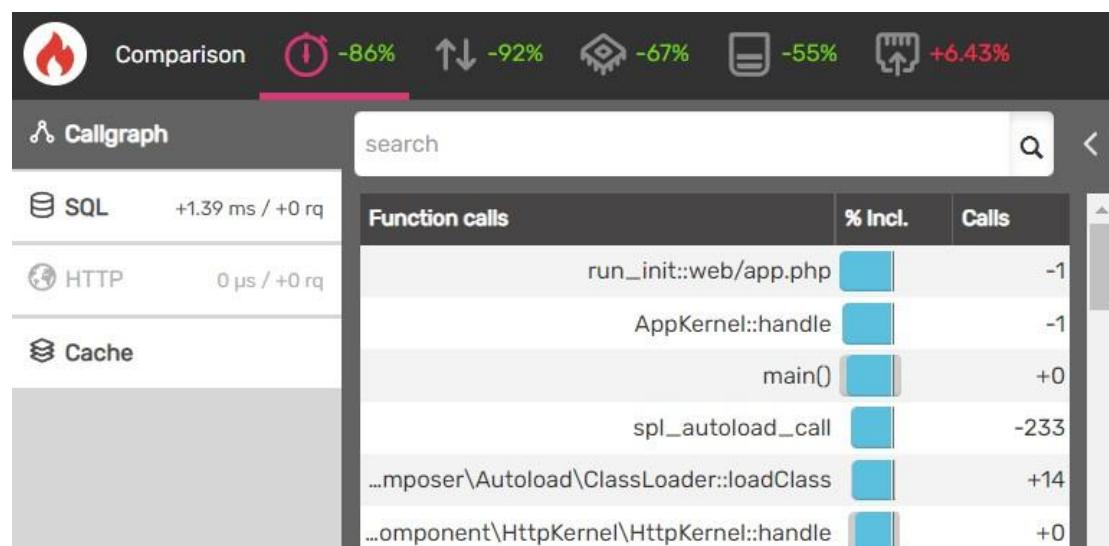
Comparaison avec le premier profil ( nom du path : /tasks ) :



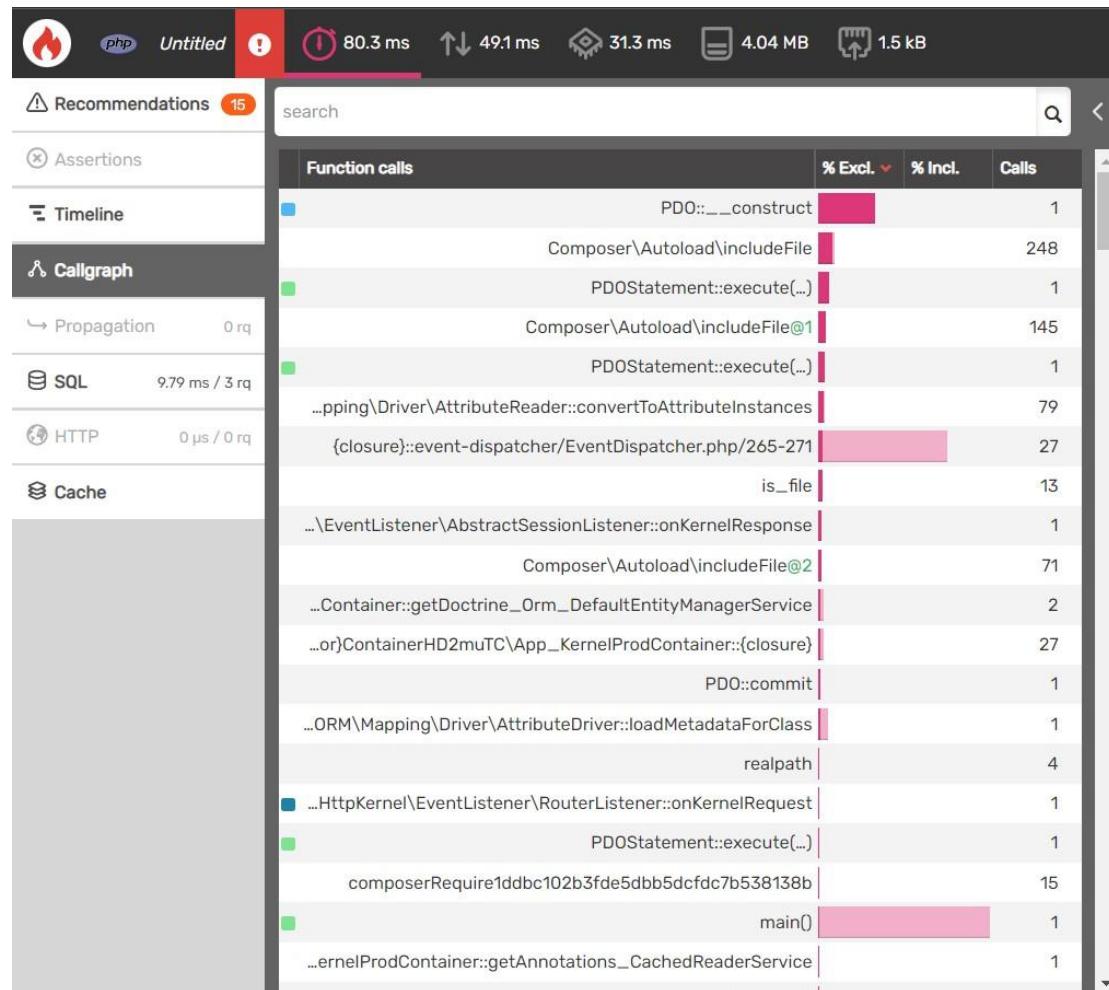
Statut 302, Method GET, Path /tasks/{id}/toggle :



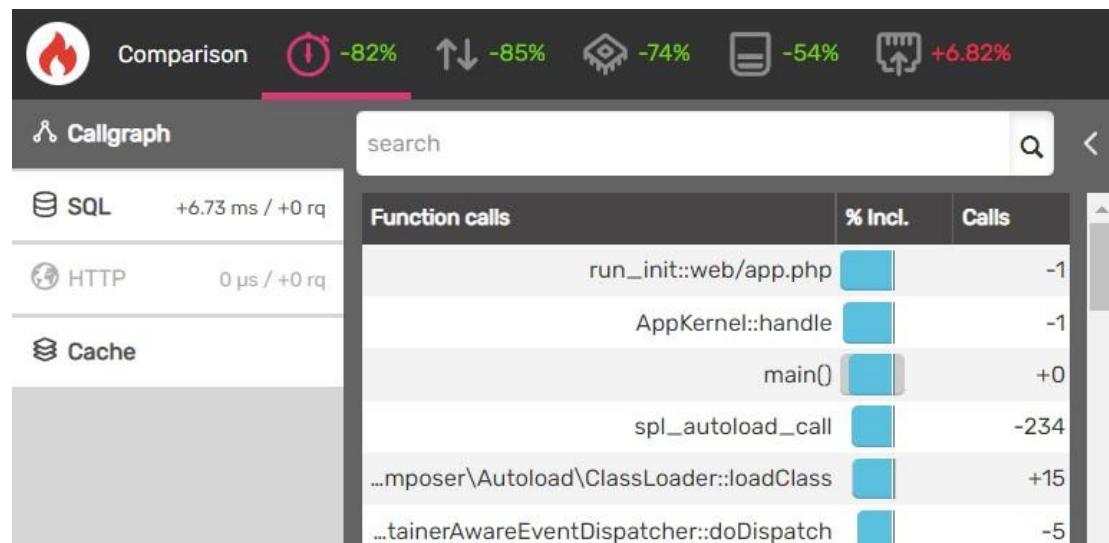
Comparaison avec le premier profil :



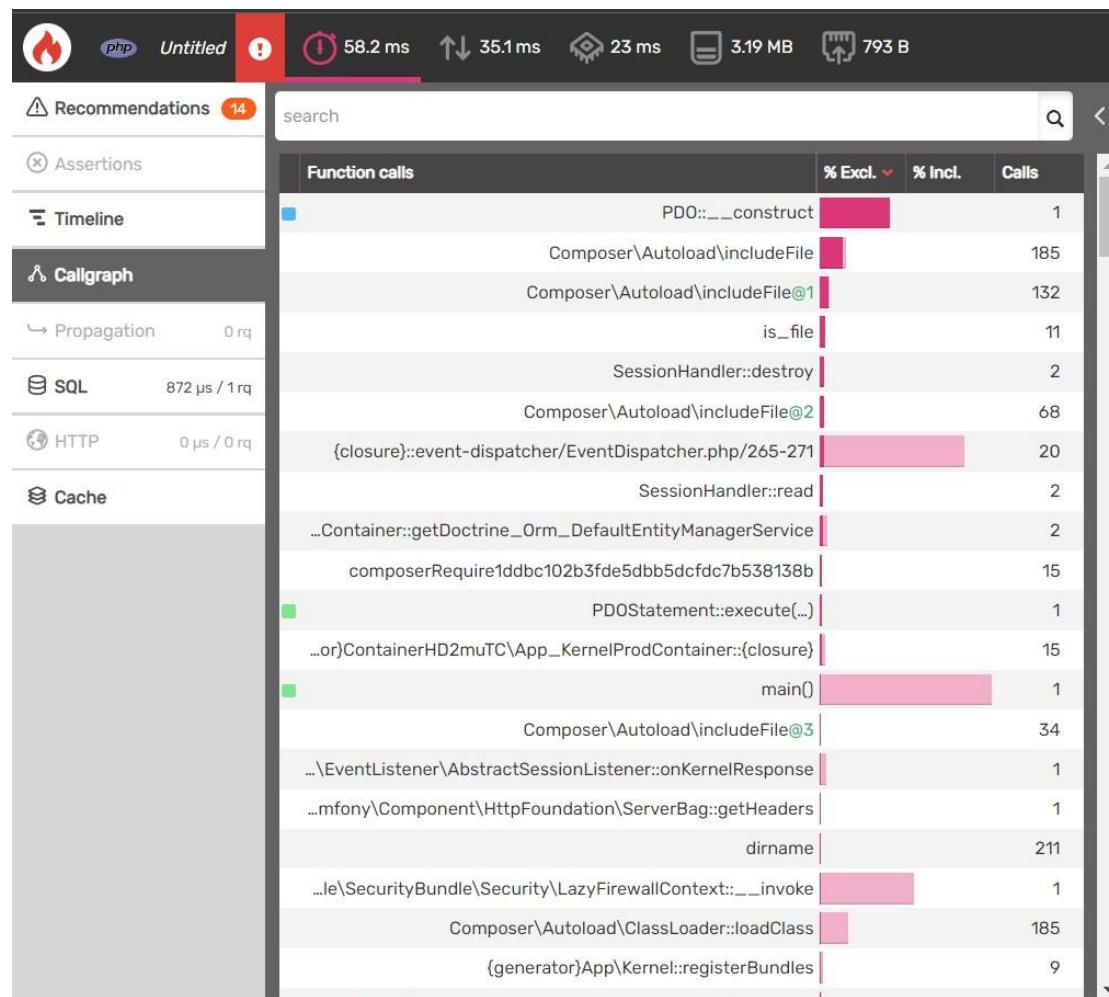
Statut 302, Method GET, Path /tasks/{id}/delete :



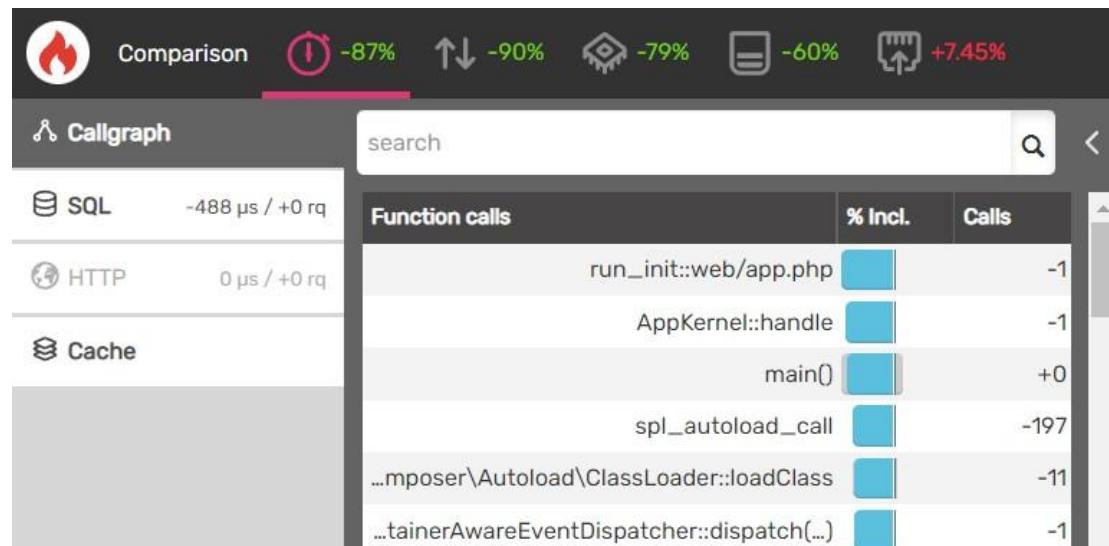
Comparaison avec le premier profil :



Statut 302, Method GET, Path /logout :



Comparaison avec le premier profil :



## Résumé du comparatif des analyses de performance :

Les différentes requêtes possibles	Temps - avant	Temps - après	Différentiel en pourcentage (%)				
			Temps	I/O Time	Processeur	Mémoire	Network
Statut 200, GET, Path /login :	321 ms	30,1 ms	-89%	-95%	-74%	-50%	0%
Statut 302, POST, Path /login : ( login_check )	1,55 sec	2 sec	29,50%	-59,00%	50,30%	-57%	8,05%
Statut 200, GET, Path /users :	496 ms	50 ms	-90%	-93%	-82%	-60%	165%
Statut 200, GET, Path /users/create :	440 ms	104 ms	-76%	-90%	-41%	-39%	0%
Statut 302, POST, Path /users/create :	1,67 sec	1,21 sec	-28%	-75%	-11%	-53%	126%
Statut 200, GET, Path /users/{id}/edit :	618 ms	83,9 ms	-86%	-92%	-75%	-58%	84,10%
Statut 302, POST, Path /users/{id}/edit :	1,66 sec	1,28 sec	-23%	-81%	-4,94%	-53%	85%
Statut 200, GET, Path / :	477 ms	40,8 ms	-91%	-94%	-84%	-62%	7,45%
Statut 200, GET, Path /tasks/undone : ( tasks )	503 ms	63,5 ms	-87%	-93%	-74,00%	-56%	251,00%
Statut 200, GET, Path /tasks/create :	763 ms	289 ms	-62%	-93%	25,20%	-61%	7,45%
Statut 302, POST, Path /tasks/create :	546 ms	101 ms	-81%	-91%	-55%	-53%	7,59%
Statut 200, GET, Path /tasks/{id}/edit :	642 ms	196 ms	-69%	-80%	-43%	-49%	6,08%
Statut 302, POST, Path /tasks/{id}/edit :	557 ms	87,1 ms	-84%	-90%	-69%	-53%	4,21%
Statut 302, GET, Path /tasks/{id}/toggle :	448 ms	64,1 ms	-86%	-92%	-67%	-55%	6,43%
Statut 302, GET, Path /tasks/{id}/delete :	439 ms	80,3 ms	-82%	-85%	-74%	-54%	6,82%
Statut 302, Method GET, Path /logout :	448 ms	58,2 ms	-87%	-90%	-79%	-60%	7,45%

## Conclusion du comparatif des analyses de performance :

Afin d'améliorer la performance, plusieurs actions ont été entreprises : on a réunis les services container dans un seul fichier, l'autoloader de composer a été optimisé pour la production et enfin nous avons mis en place le système de cache d'OPCache. Le résultat en terme de performance est très significatif, ainsi l'accès à la page d'accès /login s'effectue en 30 ms contre 321ms précédemment, c'est-à-dire en 10 fois moins de temps. Ce gain très significatif en terme de temps touche l'ensemble des pages sauf celles les plus lentes que nous avons repérées dans la partie 1 : POST login\_check, POST users/create et POST users/id/edit. Cela s'explique facilement par l'intervention des fonctions d'encodage de mot de passe très énergivore ( encodePassword(), passwordVerify()), ces fonctions prennent le même temps pour effectuer leur travail, avec ou pas cache. Outre ce gain de temps observé, on observe un gain du I/O Time c'est-à-dire le temps d'attente du système, temps durant lequel l'activité du processus n'est pas destiné à faire des calculs mais à attendre les opérations de lecture et d'écriture, de l'ordre de 90% sur l'ensemble de toutes les pages ainsi qu'un gain de mémoire consommé de l'ordre de 60% et un gain de temps lié au processeur CPU aux valeurs plus disparates. Seules les données envoyées sur le réseau sont plus importantes.

## Tests et test-coverage avec PHPUnit :

Afin de garantir le fonctionnement du site, les tests unitaires et fonctionnels ont été implémentés. 57 tests et 214 assertions ont été écrits.

```
Ikan Hiu@Suranadi MINGW64 ~/Desktop/Symfony/P8-ToDoApp (main)
$ make tests
PHPUnit 9.5.21 #StandWithUkraine

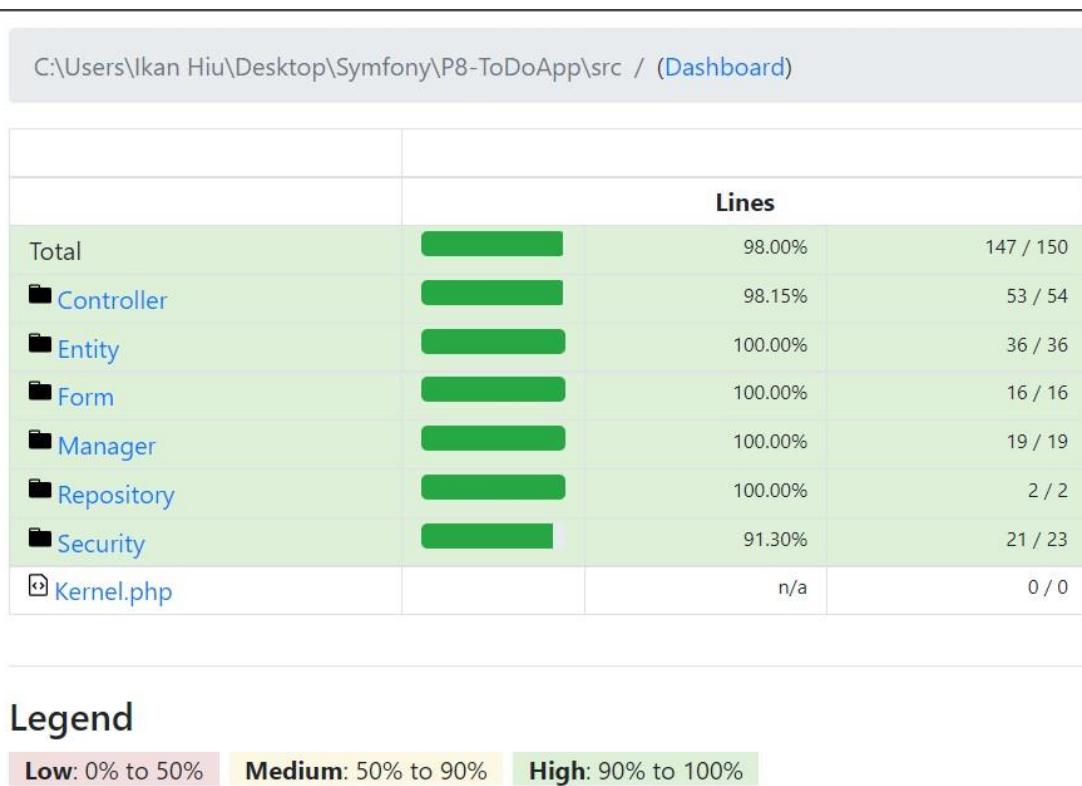
Testing
..... 57 / 57 (100%)

Time: 00:14.854, Memory: 72.00 MB

OK (57 tests, 214 assertions)

Ikan Hiu@Suranadi MINGW64 ~/Desktop/Symfony/P8-ToDoApp (main)
$
```

Le rapport du taux de couverture du code à été généré pour le dossier src/ ( à l'exclusion des fixtures ). Le taux de couverture du code s'établit à 98.00% des lignes de code écrites avec 147 lignes testées sur 150.



Generated by `php-code-coverage 9.2.15` using `PHP 8.0.14` and `PHPUnit 9.5.21` at Sat Jul 16 9:25:28 UTC 2022.

## Respect des principes SOLID :

Les améliorations apportées visent également à conformer les code aux principes SOLID.

Afin de suivre le principe de Responsabilité, nous avons individualisé les multiples actions effectuées dans les controllers. Ainsi nous avons créer les classes Manager afin d'effectuer des opérations sur nos objects et les classes Repository afin de communiquer avec la base de donnée. Chaque fonction ne fait maintenant qu'une seule et unique chose.

Afin de suivre le principe de principe d'inversion de dependances, nous avons injecter des interfaces en argument afin de ne pas avoir à dépendre de nos entités. Dans la partie 2, nous avions conseillé d'utiliser une interface afin de recuperer notre encodeur. Voici les améliorations apportées :

The screenshot shows two code editor tabs. The top tab is titled 'TaskController.php' and contains PHP code for a controller action. The bottom tab is titled 'UserManager.php' and contains PHP code for a service class. Both files use a color-coded syntax highlighting scheme.

```
TaskController.php
78
79     #[Route('/tasks/{id}/delete', name: 'task_delete', methods: ['GET', 'POST'])]
80     public function delete(Task $task, TaskManagerInterface $taskManager): Response {
81         $this->denyAccessUnlessGranted(attribute: 'delete', $task);
82         $taskManager->delete($task);
83         $this->addFlash(type: 'success', message: 'La tâche a bien été supprimée.');
84
85         return $this->redirectToRoute(route: 'homepage');
86     }
87 }
88

UserManager.php
8
9     class UserManager implements UserManagerInterface
10    {
11        private UserPasswordHasherInterface $hasher;
12        private EntityManagerInterface $em;
13
14        public function __construct(EntityManagerInterface $em, UserPasswordHasherInterface $hasher) {
15            $this->em = $em;
16            $this->hasher = $hasher;
17        }
18
19        public function new(string $plainPassword, User $user): void {
20            $password = $this->hasher->hashPassword($user, $plainPassword);
21            $user->setPassword($password);
22        }
23    }
24
```

Ici lorsque nous encodons le mot de passe nous utilisons un hasher issu de la `UserPasswordHasherInterface`, de plus lorsque nous appelons nos managers dans le controller nous utilisons des objects managers issus

d'interfaces que nous avons créés. Ainsi nous respectons le principe d'inversion de dépendance qui préconise de passer des abstractions en paramètre plutôt que les objets eux-mêmes.

Afin de suivre le principe de ségrégation des interfaces, nous n'avons pas créer une seule interface EntityManagerInterface mais deux petites interfaces : UserManagerInterface et TaskManagerInterface afin que les interfaces collent au plus près aux actions de la classe.

---

## **Conclusion aux améliorations apportées :**

Le project a été mis à jour sur les versions actuelles de Symfony et PHP, Les failles de sécurité et les risques de bugs ont été corrigés, la performance a été nettement améliorée. La qualité du code a été mise à niveau afin de suivre les recommandations des principes SOLID.

Du point de vue front-end, nous avons mis en place les appels aux bootstraps css et js dernières versions, et nous avons corriger les incohérentes et les illogismes qui se trouvaient dans les views. Les améliorations sont malgré tout restées de ce point de vue là, basique.

Enfin les tests unitaires et fonctionnels ont été implémentés afin de pouvoir péréniser l'application sur le long-terme. Un rapport de couverture de code a été généré qui stipule que le code est couvert à hauteur de 98%.