



Attendu :

Les fondateurs souhaitent pérenniser le développement de l'application. Cela dit, ils souhaitent dans un premier temps faire un état des lieux de la dette technique de l'application.

Au terme de votre travail effectué sur l'application, il vous est demandé de produire un audit de code sur les deux axes suivants : la qualité de code et la performance.

Partie : Présentation

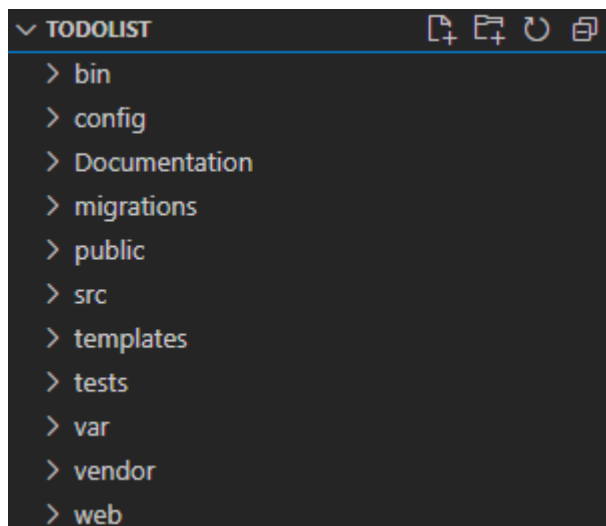
Partie : Audit de qualité du code

Partie : Performance de l'application

Partie : Tests automatisés

Partie : Comment contribuer

- **Architecture.**



Bin: Contient les fichiers de commandes permettant d'effectuer des actions sur un projet Symfony.

Config : Configuration des packages, services et routes (YAML)

Migration : Contient les fichiers de migrations Doctrine -> BDD

Public : Point d'entrée de l'application, index.php. Contient les images.

SRC : Cœur du projet ! Dossier qui contient la logique de votre application.

Templates : Contient nos Views. Symfony utilise le moteur de Template Twig par défaut.

Tests : Contient les fichiers de test pour PHPUNIT

Var : Cache et fichiers de log.

WEB : Contient les résultats (rapport de couverture) des tests PHPUNIT

Vendor : Packages de Symfony listés dans le Fichier Composer.json

- **Principaux Paquet :**
 - **Annotation** : Permet de définir les routes en commentant le code avec #
 - **Doctrine** : Doctrine est l'ORM (object-relational mapping) par défaut du framework Symfony.
 - **EazyAdmin** : Générateur de page d'administration
 - **Email-validator** : Permet la validation d'un compte par l'envoi d'un mail (via google-mailer)
 - **Flex** : Pluggin Composer permettant l'installation de paquets supplémentaire
 - **Form** : Permet de créer, traiter et réutiliser facilement des formulaires HTML
 - **Google-Mailer** : Permet l'envoi de mail via un compte Gmail
 - **Security-Bundle** : Intégration des composants de sécurité qui permet notamment la création des Controller d'enregistrement et d'authentification des utilisateurs
 - **Password-hasher** : Permet l'encryptage des mots de passe.
 - **PHPUnit** : Paquet permettant d'effectuer les tests unitaires.
 - **TWIG** : Le moteur de templates pour le langage de programmation PHP, utilisé par défaut par le framework Symfony.

La liste complète des paquets ainsi qu'une description est disponible dans le fichier `Paquet.md` ou peut être obtenu par la commande *Composer info*

AUDIT DE QUALITE DU CODE


- Analyse du code.

Repository name		Grade	CATEGORY	
TODOLIST		B	All	7086
			Code Style	7059
			Error Prone	19
			Security	3
			Unused Code	5

Le projet n'obtient que la note de B sous Codacy. Il y a donc des améliorations à faire.

Le gros des issues concerne le code style, avec un trop grand usage du « !important » dans les CSS.

Les Issues concernant « Security » et « Unused code » concernent les packages tel que EAZYADMIN et les librairies tels que BOOTSTRAP et JQUERY.


65/100

4.25 hours
to get the Platinum Medal

Search


Severity
5 Minor
1 Info

Risk
4 Productivity
2 Reputation

Developer
5 Avamdui
1 Collective

Stats
Lines of code: 37,059
Nb of suggestions: 6

Last commit
Update P8-
Documentation
technique.docx by avam-
dui 5 minutes ago. main

 SentryInsight

Your project is running on our new infrastructure!
Builds are now more reliable and faster.
You may need to adapt the `pre_composer_script` and `post_composer_script` options in your configuration. If you filter Git access by IP addresses, you will also need to update those: contact us for more details.
Give us your feedback or report issues at symfony.com/support

Changes: +27 suggestions 0 critical 27 major 0 minor 0 info

SymfonyInsight detected 1 upgrade issue in your project.
Upgrade your plan to see the details and how to upgrade

Suggestions (6) Fixed Ignored (38) Stats

▼ Your project templates should not be too long

Read doc Ignore all Productivity Minor

10% of all your templates have more than 200 lines, the threshold is 5%.
Time to fix: about 1 hour

7

2

0

1

0

0

0 - 50 51 - 100 101 - 200 201 - 400 401 - 800 801 +

90% 10%

• [templates/task/list.html.twig](#) is 239 lines long

Le projet obtient la médaille d'argent sous Symfony insight, qui nous fait remarquer que nos Templates sont trop gros, ceci étant principalement dus à l'affichage des Taches sous forme de KANBAN qui représente la grande partie du site.

ToDo & Co

- Sécurité.

Commande : *symfony security:check*

```
PS C:\Users\VMROOT\Documents\OPENCLASSROOM\p8\TODOLIST> symfony security:check

INFO A new Symfony CLI version is available (5.4.3, currently running 5.3.3).

If you installed the Symfony CLI via a package manager, updates are going to be automatic.
If not, upgrade by downloading the new version at https://github.com/symfony-cli/symfony-cli/releases
And replace the current binary (symfony.exe) by the new one.

Symfony Security Check Report
=====

No packages have known vulnerabilities.

Note that this checker can only detect vulnerabilities that are referenced in the security advisories database.
Execute this command regularly to check the newly discovered vulnerabilities.
```

Pas de vulnérabilité détecté par le rapport de sécurité Symfony.

- Mise à jour.

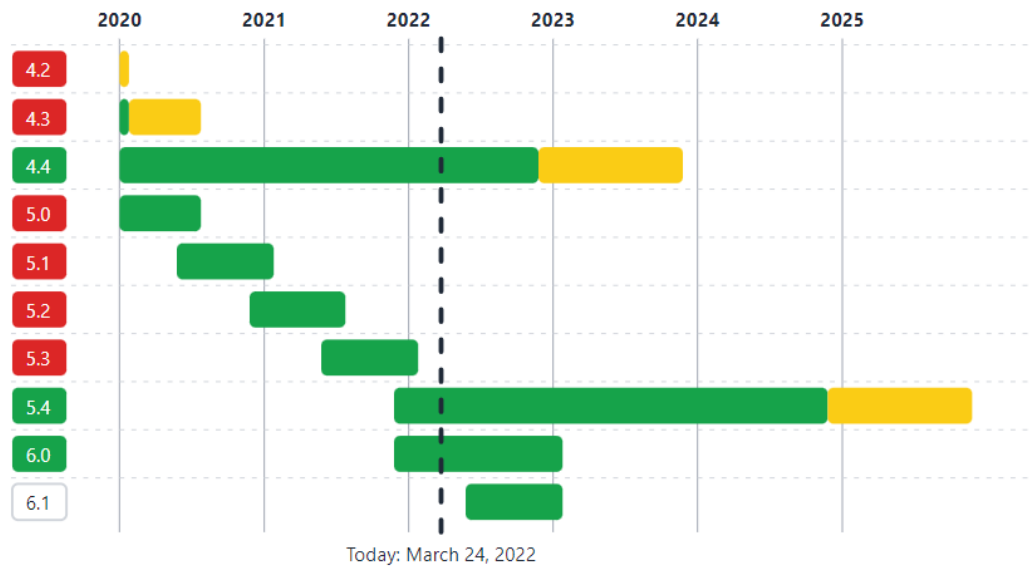
Commande : *composer outdated*

```
PS C:\Users\VMROOT\Documents\OPENCLASSROOM\p8\TODOLIST> composer outdated
Info from https://repo.packagist.org: #StandWithUkraine
Color legend:
- patch or minor release available - update recommended
- major release available - update possible
psr/cache 2.0.0 3.0.0 Common interface for caching libraries
psr/container 1.1.2 2.0.2 Common Container Interface (PHP FIG PSR-11)
psr/link 1.1.1 2.0.1 Common interfaces for HTTP links
psr/log 2.0.0 3.0.0 Common interface for logging libraries
symfony/asset v5.4.3 v6.0.3 Manages URL generation and versioning of web assets such as CSS stylesheets, JavaScript files and image files
symfony/browser-kit v5.4.3 v6.0.3 Simulates the behavior of a web browser, allowing you to make requests, click on links and submit forms programmatically
symfony/cache v5.4.6 v6.0.6 Provides an extended PSR-6, PSR-16 (and tags) implementation
symfony/cache-contracts v2.5.0 v3.0.0 Generic abstractions related to caching
symfony/config v5.4.5 v6.0.5 Eases the creation of beautiful and testable configuration values of any kind
symfony/console v5.4.3 v6.0.5 Eases the creation of beautiful and testable command line interfaces
symfony/css-selector v5.4.3 v6.0.3 Converts CSS selectors to XPath expressions
symfony/debug-bundle v5.4.3 v6.0.3 Provides a tight integration of the Symfony VarDumper component and the ServerLogCommand from MonologBridge into the Symfony full-...
symfony/dependency-injection v5.4.6 v6.0.6 Allows you to standardize and centralize the way objects are constructed in your application
symfony/doctrine-bridge v5.4.6 v6.0.6 Provides integration for Doctrine with various Symfony components
symfony/doctrine-messenger v5.4.3 v6.0.3 Symfony Doctrine Messenger Bridge
symfony/dotenv v5.4.5 v6.0.6 Eases DOP navigation for HTML and XML documents
symfony/dotenv v5.4.5 v6.0.5 Registers environment variables from a .env file
symfony/error-handler v5.4.3 v6.0.3 Provides tools to manage errors and ease debugging PHP code
symfony/event-dispatcher v5.4.3 v6.0.3 Provides tools that allow your application components to communicate with each other by dispatching events and listening to them
symfony/expression-language v5.4.3 v6.0.3 Provides an engine that can compile and evaluate expressions
symfony/filesystem v5.4.6 v6.0.6 Provides basic utilities for the filesystem
symfony/finder v5.4.3 v6.0.3 Finds files and directories via an intuitive fluent interface
symfony/form v5.4.5 v6.0.5 Allows to easily create, process and reuse HTML forms
symfony/framework-bundle v5.4.3 v6.0.6 Eases integration between Symfony components and the Symfony full-stack framework
symfony/google-mailer v5.4.3 v6.0.3 Symfony Google Mailer Bridge
symfony/http-client v5.4.5 v6.0.5 Provides powerful methods to fetch HTTP resources synchronously or asynchronously
symfony/http-client-contracts v2.5.0 v3.0.0 Generic abstractions related to HTTP clients
symfony/http-foundation v5.4.6 v6.0.6 Defines an object-oriented layer for the HTTP specification
symfony/http-kernel v5.4.6 v6.0.6 Provides a structured process for converting a Request into a Response
symfony/intl v5.4.5 v6.0.5 Provides a PHP replacement layer for the C intl extension that includes additional data from the ICU library
symfony/mailer v5.4.5 v6.0.5 Helps sending emails
symfony/messenger v5.4.3 v6.0.3 Helps applications send and receive messages to/from other applications or via message queues
symfony/mime v5.4.3 v6.0.3 Allows manipulating MIME messages
symfony/monolog-bridge v5.4.3 v6.0.3 Provides integration for Monolog with various Symfony components
symfony/notifier v5.4.6 v6.0.6 Sends notifications via one or more channels (email, SMS, ...)
symfony/options-resolver v5.4.3 v6.0.3 Provides an improved replacement for the array_replace PHP function
symfony/password-hasher v5.4.3 v6.0.3 Provides password hashing utilities
symfony/process v5.4.5 v6.0.5 Executes commands in sub-processes
symfony/property-access v5.4.5 v6.0.5 Provides functions to read and write from/to an object or array using a simple string notation
symfony/property-info v5.4.3 v6.0.3 Extracts information about PHP class' properties using metadata of popular sources
symfony/proxy-manager-bridge v5.4.6 v6.0.6 Provides integration for ProxyManager with various Symfony components
symfony/routing v5.4.3 v6.0.5 Maps an HTTP request to a set of configuration variables
symfony/runtime v5.4.5 v6.0.5 Enables decoupling PHP applications from global state
symfony/security-bundle v5.4.5 v6.0.5 Provides a tight integration of the Security component into the Symfony full-stack framework
symfony/security-core v5.4.5 v6.0.5 Symfony Security Component - Core Library
symfony/security-csrf v5.4.5 v6.0.3 Symfony Security Component - CSRF Library
symfony/security-http v5.4.5 v6.0.5 Symfony Security Component - HTTP Integration
symfony/serializer v5.4.6 v6.0.6 Handles serializing and deserializing data structures, including object graphs, into array structures or other formats like XML an...
```

Mise à jour possible vers Symfony 6. De nombreux paquets peuvent ainsi être mise à jours se qui tendrai à renforcer la sécurité et la qualité du code.

Cependant nous resterons sur la versions 5.4 qui est la LTS (Long-Term Support Release).

Symfony Releases Calendar



ToDo & Co

- **Test fonctionnel et unitaire**

Écriture d'une série de tests unitaires et fonctionnels afin de garantir le fonctionnement de l'application.

Ci-dessous l'exemple du test de création de tâche.

```
•  
• public function testCreateAction()  
• {  
•     // Login grâce a la fonction loginuser()  
•     $userRepository = static::getContainer()->get(UserRepository::class);  
•     $userTest = $userRepository->findOneByEmail('test@test.fr');  
•     $this->client->loginUser($userTest);  
•     // Nous accédons à la page "mes tâches" de l'user ayant l'ID 1  
•     $crawler = $this->client->request('GET', '/tasks/1');  
•     $this->assertEquals(200, $this->client->getResponse()->getStatusCode());  
•  
•     //Recherche du bouton "ajouter"  
•     $buttonCrawlerNode = $crawler->selectButton("Ajouter");  
•     // Sélection du formulaire associé a ce bouton  
•     $form = $buttonCrawlerNode->form();  
•     // On remplit le formulaire  
•     $form['task[title]'] = 'PHPUNIT';  
•     $form['task[content]'] = 'Faire les test unitaires';  
•     //On soumet le formulaire  
•     $this->client->submit($form);  
•     // Vérification de la redirection et du code retour 200  
•     $this->assertTrue($this->client->getResponse()->isRedirect());  
•     $crawler = $this->client->followRedirect();  
•     $this->assertEquals(200, $this->client->getResponse()->getStatusCode());  
• }  
•
```

A l'issue des tâches, PHPUNIT génère un rapport de couverture. Nous couvrons ainsi près de 92% des fonctions et méthodes du site. La couverture de 100% des cas n'est pas nécessaires, mais il est important de couvrir l'intégralité des fonctions clefs du site, à savoir la gestion des tâches.

		Code Coverage			
		Lines		Functions and Methods	
Total		88.52%	185 / 209		91.53% 54 / 59
Controller		84.62%	110 / 130		80.00% 16 / 20
Entity		100.00%	46 / 46		100.00% 30 / 30
Form		100.00%	13 / 13		100.00% 3 / 3
Repository		100.00%	3 / 3		100.00% 3 / 3
Security		76.47%	13 / 17		66.67% 2 / 3
Kernel.php		n/a	0 / 0		n/a 0 / 0













PERFORMANCE DE L'APPLICATION

- Tests de performance

Comme conseillé par Symfony, la performance de l'application « To Do List » est analysée grâce à l'application professionnelle Blackfire.io.













Blackfire Profiler est un outil qui instrumente les applications PHP pour collecter des données sur les ressources serveur consommées comme la mémoire, le temps CPU et les opérations d'E / S

- En Dev avec profiler :

200 GET https://127.0.0.1:8000/register <i>TODOLIST - REGISTER</i> Created 22 hours ago by vincent gabrych    ↳ 0 rq ⌚ 602 ms 📄 41 MB 🌐 0 µs / 0 rq 🗑️ 0 µs / 0 rq
200 GET https://127.0.0.1:8000/tasks <i>TODOLIST - LISTES DES TACHES</i> Created 22 hours ago by vincent gabrych    ↳ 0 rq ⌚ 634 ms 📄 40.3 MB 🌐 0 µs / 0 rq 🗑️ 0 µs / 0 rq
200 GET https://127.0.0.1:8000/tasks/3 <i>TODOLIST - MES TACHES</i> Created 22 hours ago by vincent gabrych    ↳ 0 rq ⌚ 567 ms 📄 40.1 MB 🌐 0 µs / 0 rq 🗑️ 0 µs / 0 rq
200 GET https://127.0.0.1:8000/login <i>TODOLIST - HOMEPAGE - LOGIN</i> Created 22 hours ago by vincent gabrych    ↳ 0 rq ⌚ 433 ms 📄 32.4 MB 🌐 0 µs / 0 rq 🗑️ 0 µs / 0 rq

ToDo & Co

- Passage en Production :

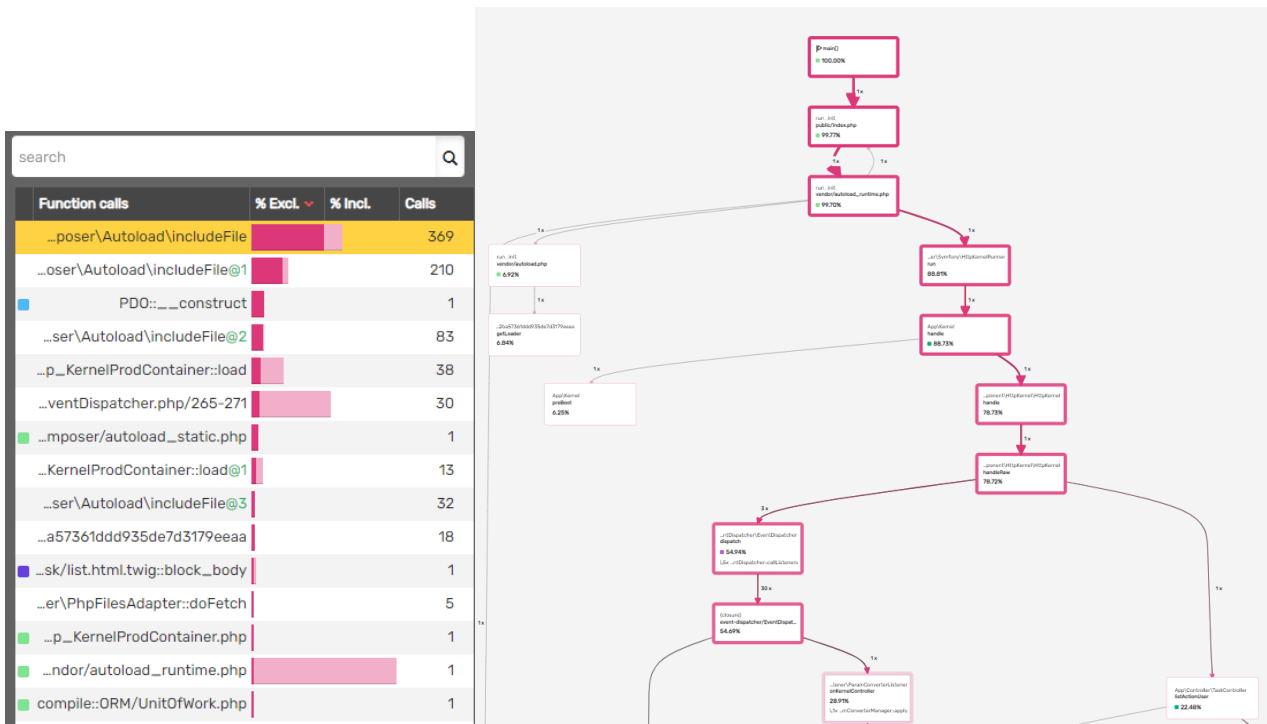
200 GET https://127.0.0.1:8000/register TODOLIST - REGISTER - PROD OPT Created 21 hours ago by vincent gabrych    ↳ 0 rq ⌚ 141 ms 📄 22.2 MB 🌐 0 µs / 0 rq 📊 0 µs / 0 rq
200 GET https://127.0.0.1:8000/login TODOLIST - LOGIN - TACHES - PROD OPT Created 21 hours ago by vincent gabrych    ↳ 0 rq ⌚ 104 ms 📄 17.3 MB 🌐 0 µs / 0 rq 📊 0 µs / 0 rq
200 GET https://127.0.0.1:8000/tasks TODOLIST - LISTES DES TACHES - PROD OPT Created 21 hours ago by vincent gabrych    ↳ 0 rq ⌚ 172 ms 📄 23.1 MB 🌐 0 µs / 0 rq 📊 0 µs / 0 rq
200 GET https://127.0.0.1:8000/tasks/3 TODOLIST - MES TACHES - PROD OPT Created 22 hours ago by vincent gabrych    ↳ 0 rq ⌚ 161 ms 📄 23 MB 🌐 0 µs / 0 rq 📊 0 µs / 0 rq

Le passage en production, avec la désactivation du Profiler à un impact majeur sur les performances.

Ainsi la page MES TACHES passe de 567ms à 161ms en temps de chargement.

ToDo & Co

- Analyse :



Suite à une analyse des graph Blackfire, on constate que le temps de chargement est principalement dû à l'utilisation de l'autoload de Composer.

En effet, l'autoloader de Composer utilise lors du développement de l'application est optimisé pour rechercher les classes nouvelles et modifiées. Et cela suppose d'aller chercher à chaque requête toutes les classes de l'application.

La documentation de Composer offre une solution à ce problème via la génération d’une “Class Map”.

Cela convertit les namespaces PSR-4 en ClassMap ce qui évite de faire appel au filesystem pour vérifier l'existence des classes.

Cette optimisation se fait via la commande **composer dump-autoload --optimize**.

De plus nous pouvons également mettre en place un système de cache HTTP afin d'augmenter les performance global.

CONTRIBUER

TODOLIST

Avant propos

- Le projet fonctionne sur PHP 8.0.15
- Le projet est basé sur le framework symfony 5.4.4 (Doctrine, Twig et PHPUnit)
- Git du projet : ``git clone https://github.com/avamdui/TODOLIST``

Comment Contribuer au projet

1. Cloner et Installer le repository sur votre serveur (voir le README.md)
 - Modifier le .env avec vos informations.
 - Installez les dépendances : `composer install`
 - Mettre en place la BDD :
`php bin/console doctrine:database:create`
`php bin/console doctrine:migrations:migrate`
2. Créez une branche à partir de **master** : `git checkout -b nom de la branche`
3. Ecrivez un Issue sur les modifications que vous allez apporter
4. Ecrivez votre code EN RESPECTANT LES BONNES PRATIQUES
5. Ecrivez vos Commits avant de faire un Push de la branche : `git push origin ma-Branche`
6. Mettez à jour vos issues
7. Faites un **Pull Request** et attendez sa validation

Les bonnes pratiques

1. le code

Vous devez respecter :

- Les standards du code de Symfony (``https://symfony.com/doc/current/contributing/code/standards.html``)
- Les conventions du code de Symfony (``https://symfony.com/doc/5.2/contributing/code/conventions.html``)

2. les bundles

- Les installation de bundle PHP doit se faire avec "Composer"

3. Git

Vous devez faire les choses dans cet ordre :

- Nouvelle branche à partir de master dument nommée
- Commit Correctement commentés
- Issue Correctement commentées et documentées
- Faire un update sur le code principal : `git pull origin master`

4) Tests unitaires et fonctionnels

- Les nouvelles fonctionnalité doivent avoir des tests associés (taux de couverture de 70% minimum)
- PHPUnit est à votre disposition pour créer vos tests
 - * Ecrivez vos tests dans le dossier `/test`
 - * Utiliser MakerBundle's `make:test` pour créer un squelette de test!
 - * Lancer les tests avec la commande `vendor/bin/phpunit --coverage-html web/test-coverage` pour générer un rapport de couverture

5) Architecture de fichier

- Respectez l'architecture de Symfony 5 pour vos fichiers PHP (`src\Controller\...`)
- Les vues devront être dans le répertoire `Templates`.

6) Suggestion d'amélioration

- Ajouter un espace membre / profil utilisateur, avec statistique des taches de l'utilisateur
- Permettre l'ajout de nouvelle catégorie (ex : Taches mises en attente, taches abandonnés, ...etc)
- Ajout Date de fin prévisionnelle, date de fin effective, archivage ancienne taches
- Passage à Symfony 6
- Optimisation des Templates (Trop de lignes de code)
- Mise en place d'un système de cache http

