AUDIT DE QUALITE DE CODE

Avant-Propos
Pour améliorer la qualité de code d'une application il existe des outils spécialement conçu à cet effet tels que **Codacy,** Code Climate ou CodeCov.
L'un des plus accessible de par son caractère gratuit est Codacy.
C'est celui qui a été utilisé dans le cadre de l'analyse du projet 8 ToDoList.

Un second outil utilisé en ligne de commande pour détecter des erreurs dans la rédaction du code est **PHP_CodeSniffer**
https://github.com/squizlabs/PHP_CodeSniffer

Présentation de Codacy
Codacy est un outil automatisé d'audit de code qui fournit une aide afin d'améliorer la qualité de de code et réduire la dette technique.
Codacy est utilisable via un repository distant tel que Github.

Utilisation de Codacy
Pour utiliser Codacy on doit s'enregistrer sur la page du site grâce à un compte github.
A partir du tableau de bord on va définir le ou les repository qui devront être testés par l'outil d'analyse.

Une fois que l'analyse du repository est exécuté, différents rapport sous forme de graphique ou de recommandations sont disponibles.
En sélectionnant l'analyse par fichiers, Codacy fournit une description des anomalies rencontrées au sein du code contenu dans le fichier analysé.
Pour se faire Codaci utilise un code couleur :

Rouge pour les anomalies critiques
Orange pour celles de niveau intermédiaires
Bleue pour les mineures

Lorsque les anomalies des catégories rouge et oranges sont traitées votre code sera qualifié avec un badge A ou B, ce qui est recherché.

https://github.com/damirmales/p8_todoAndCo

https://app.codacy.com/manual/d.males/p8_todoAndCo/dashboard

Utilisation de PHP_CodeSniffer

Pour installer PHP_CodeSniffer avec composer :

```
composer global require "squizlabs/php_codesniffer=*"
```

Pour détecter les erreurs il faut utiliser la commande :

```
phpcs  /path/to/code/myfile.php
```

Ensuite pour réparer (fix) les erreurs

```
phpcbf  /path/to/code/myfile.php
```

Réalisations de tests (unitaires,fonctionnels)

Ces tests se feront avec PHPUnit
Les tests seront rédigés dans un dossier dédié nommé tests et situé au même niveau que le dossier src qui contiendra les codes métiers
Les classes de test seront nommées par le mot de la classe présente dans le dossier src auquel on ajoute le suffixe “Test”, Task controller Test()
Les méthodes au sein de ces classes utilisent le préfixe “test”, test Edit Task().

Lancement des tests :
“vendor/bin/phpunit” tests/appbundle

Lancement de l’analyse de couverture de code (code coverage)
“vendor/bin/phpunit” --coverage-html tests/appbundle

Réalisation de code reviews

Faire du code review au fur et à mesure de l'avancée du projet afin de s'assurer que les règles établies ont bien été respectées.