Audit de qualité du code & performance de l'application

Ce document va vous présentez les différents éléments constituant la dette technique de l'application qui devront être corrigé par la suite pour garantir un haut niveau de performance.

Qualité du code

Version obsolète de Symfony

La version actuelle du projet est Symfony 3.2, cette version ne fait actuellement plus l'objet d'un support par la société gérer par Symfony, les bugs éventuellement présent ne ferait donc pas l'objet de correction ce qui pourrait notamment poser des problèmes de sécurité.

Il serait donc préférable de faire évoluer le projet vers une version de Symfony faisant l'objet d'un support, soit les versions 3.4 ou la version 4.0.

Fichier absent du projet

Le fichier JQuery est référencé dans la vue **base.html.twig** mais n'est pas inclus dans le projet. L'absence du fichier JQuery prive le projet de la possibilité d'utilisé une partie des fonctionnalités offerte par Bootstrap qui est utilisé dans ce projet.

Il faudra donc inclure un fichier JQuery dans le projet.

Dépréciation des méthodes lors de la validation des formulaires

Depuis la version 3.2 de Symfony, il est déprécier de contrôler la validité d'un formulaire sans avoir contrôlé que le formulaire à bien été envoyé. Faire ceci dans la version 4 de Symfony déclenchera une erreur.

Pour y remédier il faudra changer dans les actions relatives à la validation des formulaires ce code :

```
if ($form->isValid())
```

Et les remplacer par ce code

```
if ($form->isSubmitted() && $form->isValid())
```

Variable non utilisé

L'analyse via Codacy révèle la présence de variable inutilisé dans le code de cette application. Par soucis de clarté du code il est bon d'éviter la présence de variable inutilisé et donc inutile.

Exemple de l'action loginAction du SecurityController :

```php
public function loginAction(Request $request)
{
    $authenticationUtils = $this->get('security.authentication_utils');

    $error = $authenticationUtils->getLastAuthenticationError();
    $lastUsername = $authenticationUtils->getLastUsername();

    return $this->render('security/login.html.twig', array(
        'last_username' => $lastUsername,
        'error'         => $error,
    ));
}
```

Dans ce cas, la variable **$request** n'est pas utilisé dans le code de l'action et devrais donc être retiré.

Sécurité du mot de passe insuffisante

L'entité User qui gère les utilisateurs ne comporte pas d'attribut salt. Hors le salt permet de renforcer le cryptage du mot de passe de manière significative en donnant une clé de cryptage unique à chaque mot de passe. En l'absence de salt, on peut considérer le cryptage comme étant insuffisant.

Pour résoudre ce problème vous devez ajouter un salt à l'entité User et veillez à ce que chaque utilisateur bénéficie d'un salt unique généré aléatoirement.

Absence de message d'erreur adéquat

Lors de la création d'un utilisateur les contraintes d'unicités sur les emails et sur les noms d'utilisateur sont bien présentent mais ces contraintes ne sont pas associé à des messages clair rendant compte de l'erreur à l'utilisateur lors de l'envoi du formulaire. En l'état, en cas de l'erreur un message standard est affiché.

Pour résoudre ce problème, il faudra ajouter des messages personnalisé pour ces erreurs dans la classe User.

Bouton non utilisé

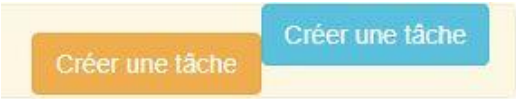
Le bouton « Consulter la liste des tâches terminées » n'est pas utilisé et ne renvoie vers aucun lien. De plus il n'y a aucune action dans le code de l'application qui corresponde au libellé de ce bouton. Il est également à noter que les taches terminé ou en cours ne sont pas séparé et toute accessible depuis le bouton « Consulter la liste des tâches à faire » ce qui est incohérent avec la présence de deux bouton.

Pour résoudre ce problème, il y a deux possibilités :

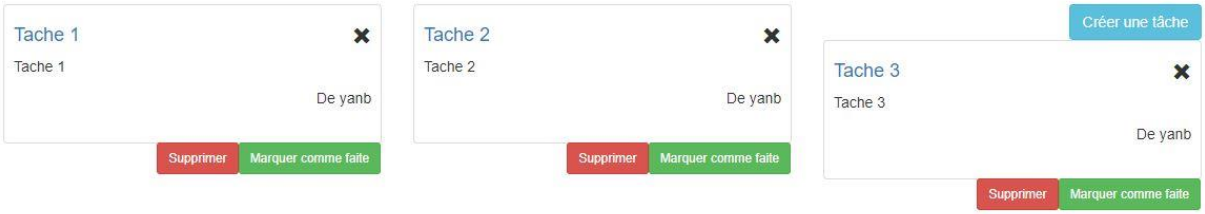
- Avoir un seul bouton avec un libellé cohérent indiquant la présence de toutes les tâches
- Avoir deux boutons renvoyant chacun vers des listes de tâches distinctes

Mauvais positionnement des boutons dans la liste des tâches

Quand la liste des tâches ne contient aucune tâche il y a 2 boutons de créations d'une tâche cote à cote qui sont mal aligné.



Quand la liste contient au moins 3 tâches, le bouton de création de tâche n'est plus correctement aligné.



Pour résoudre ce problème, il faudra supprimer l'un des boutons de création d'une tâche en cas de liste vide. Dans le cas d'une liste contenant au moins 3 tâches, il faudra veiller à ce que le bouton de création d'une tâche ne vienne pas casser l'alignement.

Navigation difficile

Le site web ne contient pas de menu accessible sur toutes les pages du site, ce qui rend parfois extrêmement difficile la navigation, il par exemple pas possible depuis la page d'édition d'une tâche de revenir à la listes des tâches avec les liens fourni par le site web.

Pour résoudre ce problème, il faudrait d'abord ajouter un menu avec des liens permettant de se rendre très facilement sur les principales pages du site web, ensuite il faudra ajouter des liens sur certaine page permettant de revenir à une page parente (Exemple : depuis les formulaires de création et d'édition d'une tache il doit être possible de revenir sur la page listant les tâches).

Mauvaise organisation du formulaire d'édition d'un utilisateur

Le formulaire d'édition d'un utilisateur ne sépare pas la modification du mot de passe de la modification du reste des informations. Cela créez une difficulté, car pour modifier une information il faut entrer son mot de passe à deux reprise même si on ne souhaite pas modifier ce dernier. Cela rend la modification des informations plus lourde que nécessaire.

Pour résoudre ce problème, il faudrait mettre en place 2 formulaires séparé : un formulaire permettant de modifier le mot de passe et un formulaire permettant de modifier le reste des informations de l'utilisateur.

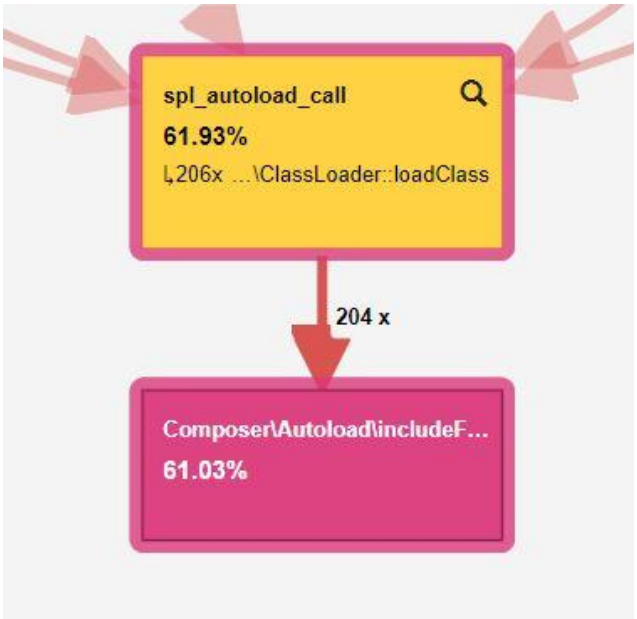
Performance

Le poids de l'autoloader de composer

Pour optimiser les performances de l'autoloader vous devez utiliser cette commande :

composer dump-autoload --optimize --no-dev --classmap-authoritative

Malgré l'optimisation de l'autoloader de composer pour l'environnement de production, l'autoloader de composer mobilise autour de 60% du temps d'exécution sur l'ensemble des pages testé.



L'une des manières possible de minimiser ce problème est de limiter le nombre de dépendance.