Rapport d'activité sur le projet de développement web à base de Framework web

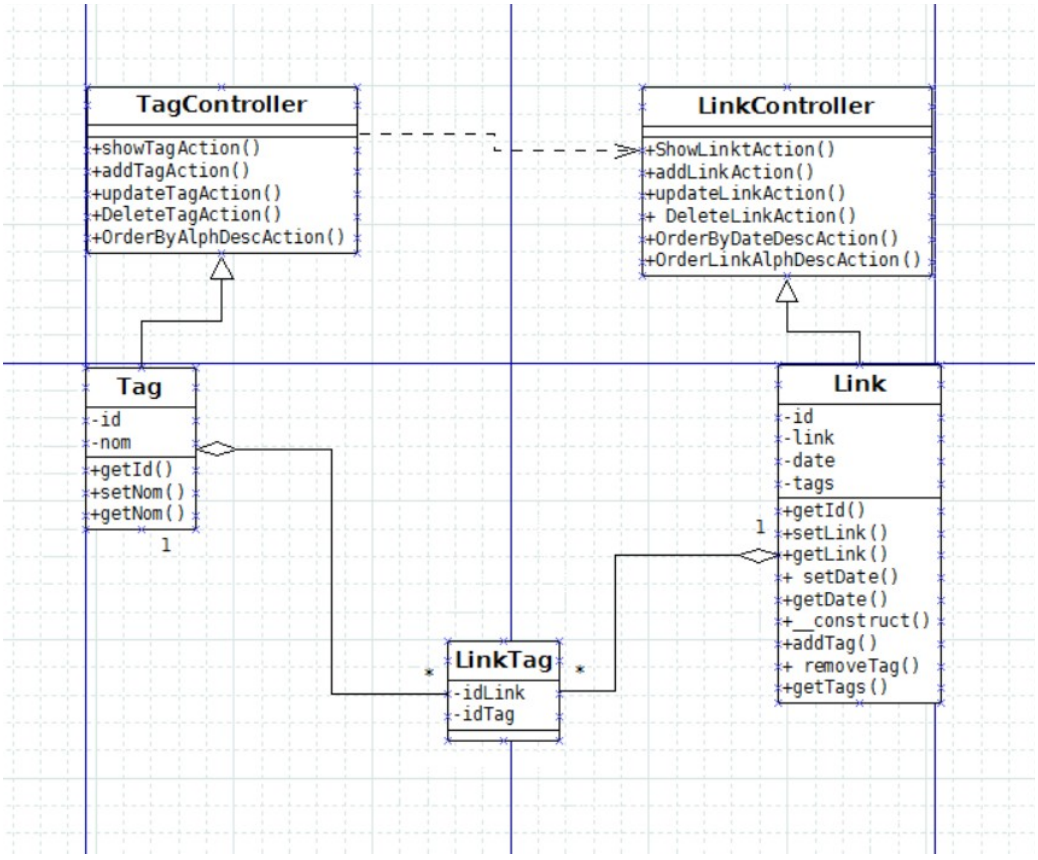
L'équipe de développement : Alex Lys , Franck Xiong

Pour ce projet, nous avons travaillé en pair-programming (Alex LYS-PHORIMAVONG et Franck XIONG). Nous nous sommes réparties les tâches, Franck s'occupait de mettre en place la mise en page de notre application, tandis que moi Alex, m'occupait de l'implémentation du code (méthodes, authentification etc). Le pair-programming nous a permis d'être efficace dans le sens où chacun intervenait, donner son opinion, son ressentie ainsi que ses connaissances (que ce soit en symfony , php , html/css) afin de faciliter développement de l'applicaton web.Nous avons atteint notre objectif, c'est-à -dire livré une application web ,développé sous le framework Symfony, fonctionnelle et opérationnelle répondant aux attentes de la mission. Cependant, certains points sont à améliorer , nous y reviendrons plus en détail plus loin dans notre compte rendu.

La couverture fonctionnelle de l'application

Sur le diagramme de cas d'utilisation ci-dessous, nous pouvons constater deux types d'utilisateurs : User qui ne peut accéder qu'au site en s'authentifiant, Admin qui accède au site en s'authentifiant et possède les droits de modification, suppression et d'ajout des liens et des tags.

Le diagrame de classe



Quelques explication de notre code :

EXPLICATION DU CODE // DESCRIPTION DU CODE

-Méthode ShowLinkAction :

```
/**
 * @Route("/links", name="links")
 * @Template("links.html.twig")
 */
public function ShowLinktAction(Request $request) {
    $repository = $this->getDoctrine()->getRepository('AppBundle:Link');

    $query = $repository->createQueryBuilder('l')
            ->orderBy('l.date', 'ASC')
            ->getQuery();

    $links = $query->getResult();
    return array('links' => $links);
}
```

Le nom de la route est « links», avec pour URL « /links »

On renvoie les éléments obtenu par cette méthode dans une vu twig

Creation d'une requête DQL qui nous retourne tout les élements de la table « Link », dans l'odre croissant de date

Cette méthode nous permet de renvoyer tout les liens présent dans notre base de données sur notre vue Twig (dans un tableau plus précisément).

```
{% for news_link in links %}
</tr>
<tr>
    <td id="link"> <a href="{{news_link.link}}" target="_blank"> {{news_link.link}} </a> </td>
```

Pour chaque liens de la collection d'objet « links », on affiche la valeur du champ désiré (ici on veut afficher la valeur du camp « link »)

-Fichier sécurity.yml

```
# http://symfony.com/doc/current/book/security.html#where-do-users-come-from-user-providers
providers:
    in_memory:
        memory:
            users:
                admin:
                    password: admin
                    roles: 'ROLE_SUPER_ADMIN'
                user:
                    password: user
                    roles: 'ROLE_ADMIN'
access_control:
        - { path: ^/login, roles: IS_AUTHENTICATED_ANONYMOUSLY }
        - { path: ^/update, roles: ROLE_SUPER_ADMIN }
        - { path: ^/delete, roles: ROLE_SUPER_ADMIN }
        - { path: ^/, roles: ROLE_ADMIN}


firewalls:
    secured_area:
        anonymous: ~
        logout:
            path:   /logout
            target: /
```

N'ayant pas réussi à implémenter les méthodes nous permettant d'exploiter des données issus de notre base de données pour pouvoir nous authentifier, nous avons mis nous même , en mémoire, des login et password (respectivement pour chaque rôle : admin / user)

Ici, nous donnons à l'utilisateur (admin ou user) l'accès a certaines partie de notre application par sécurité. Par exemple, seul l'utilisateur possédant le statut de Super Admin peut accéder au URLs commençant par « /delete ». De plus, nous n'autorisons pas les anonymes sur notre site. Ils doivent obligatoirement s'identifier.

De ce coté, symfony s'occupe lui méme de fermer notre session lorsque qu'on l'on se dirige vers l'URL « /login ».

C'est dans ce fichier que l'on peut paramétrer les systèmes d'authentifications, de déconnexion, répartir les rôles, donner des autorisations etc...

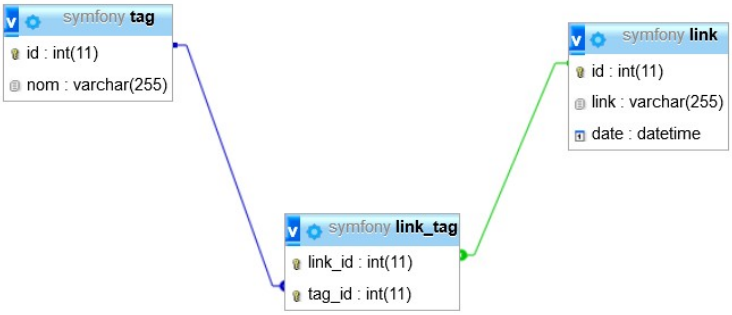
-Extrait d'une de nos vues Twig about,html.twig

```
{% extends "layout.html.twig" %}
{% block container %}

    <h1> Voici l'équipe du projet ! </h1>

{% endblock %}
```

Définir des blocks , nous permet d'y ajouter tout sortes d'éléments. On peut disposer d'autant de block qu'on le souhaite. Ici, «Voici l' équipe ... » s'affichera dans le block « container » qui est lui même contenu dans le block « body » définie dans le layout.html.twig

Un extrait simple d'une de nos vues twig. Notre vue « Layout » dispose de la barre de navigation, du titre, des footer etc.. « Extend layout.html.twig » permet d'étendre notre vue actuelle avec celle de layout. Elle disposera elle aussi de tout le contenu de la vue twig. Cela permet d'alleger le code, de le rendre plus lisible. Plus besoin de mettre tout le html.

-Schéma relationnel de notre base de données



Relation ManyToMany : Un lien peut être associer à plusieurs tag. Et un tag peut avoir plusieurs liens.

Notre ressentie :

Ce projet nous a permis de découvrir de façon plus objective le framework Symfoy. L'application du modèle MVC nous a beaucoup aidés, notamment pour débuger, corriger nos problèmes. L'API symfony étant très bien faite (anglais facile, illustrations, explications claire) , notre prise en main de ce framework a été plus simple que ce nous pensions.

Nous avions de gros doutes sur le « comment » des opérations CRUD qui devait obligatoirement être intégré à notre application web. Notre principale difficulté (non résolu) fut l'accès à notre site via l'utilisation de login et de password issu de notre base de données. Nous avons perdu énormément de temps sur ce problème. C'est pour cela que nous avons préféré mettre ses données directement en mémoire, revenir sur ce problème si on le pouvait. De plus, il reste des détails à affiner pour notre application : redéfinir des url plus juste, changer la mise en page du formulaire d'ajout et de modification, ajouter et implémenter l'option « se souvenir de moi » et « mot de passe oublié »pour la page de login, mettre d'éventuels messages flash (de confirmation d'ajout, de suppression , modification), de nouvelles fonctions de tri (trier par tag), afficher pour chaque tags sa liste de liens associé.