



ToDo & Co

Gestion de la Sécurité

Client	ToDo & Co
Créé le	15/03/2018
Auteur	C. DUVAL
Destinataires	ToDo & Co
Fichier	ToDoList Sécurité.docx

Version	Modifié le	Par	Modifications
1	15/03/2018	C. DUVAL	Création

Table des Matières

I	Authentication	3
1-	Les Utilisateurs	4
1.1	La classe User	4
1.2	L'encodage	5
1.3	L'ORM : DOCTRINE	5
1.4	La base de données	5
2-	Le firewall.....	6
II	Autorisation.....	7
1-	Les rôles	8
2-	Access Control	8
3-	Configuration via les actions de contrôleur.....	8
	 Annexe 1 :Description générale.....	 9
	Annexe 2 : Profil des acteurs.....	10

Objet

Cette documentation expliquant comment l'implémentation de la sécurité a été faite est destinée aux développeurs qui rejoignent l'équipe.

Elle comprend les réponses aux questions suivantes :

- comment s'opère l'authentification
- où sont stockés les utilisateurs.
- comment s'opère l'autorisation
- quel(s) fichier(s) il faut modifier et pourquoi ;

[CDu]	Spécifications techniques ToDoList	V 1.0
		Page 3 / 10

I AUTHENTIFICATION

L'authentification en quelques mots

L'authentification est le processus qui va permettre l'identification de l'utilisateur auprès de l'application. Il en résulte un statut qui peut prendre 2 valeurs :

- Anonyme : vous n'êtes pas identifié
- Authentifié : vous êtes identifié et reconnu comme utilisateur du site

En fonction du résultat de l'authentification et du paramétrage des firewalls du site, l'utilisateur aura accès ou pas aux différentes pages du site.

Le processus Sécurité – 1^{ère} partie

1. L'utilisateur essaye d'accéder à une page protégée par le firewall;
2. Le firewall lance le processus d'authentification en redirigeant l'utilisateur vers le formulaire de connexion (/login);
3. La page /login renvoie le formulaire via le Security Controller;
4. L'utilisateur soumet ses identifiants via le formulaire de connexion;
5. Le système de sécurité intercepte la requête, vérifie les identifiants de l'utilisateur, authentifie l'utilisateur si ceux sont valides, le renvoie vers le formulaire de connexion sinon.

[CDu]	Spécifications techniques ToDoList	V 1.0
		Page 4 / 10

1- Les Utilisateurs

L'application ToDoList utilise une base MySQL pour stocker ses utilisateurs.

1.1 La classe User

Fichier : src/AppBundle/Entity/User.php

La table de données des utilisateurs est basée sur la classe « User » qui comprend les attributs suivants :

- **Id**
 - Identifiant unique en base de données
 - Entier. Clé primaire, unique, auto-incrémenté
- **username**
 - Identifiant de l'utilisateur, paramètre de l'authentification
 - Chaîne de caractères, doit être unique
- **password**
 - Mot de passe encodé
 - Chaîne de caractères
- **email**
 - Email de l'utilisateur
 - Type Email, doit être unique
- **role**
 - Liste des rôles de l'utilisateur
 - Tableau

Afin d'être utilisée par le système de sécurité de Symfony, cette classe User implémente l'interface **UserInterface** qui impose l'écriture des fonctions suivantes en plus, si nécessaire, de nos getters et setters :

- getRoles()
- getPassword()
- getUsername()
- getSalt()
- eraseCredentials()

1.2 L'encodage

Fichier : app/config/ security.yml

encoders:

AppBundle\Entity\User: bcrypt

Le type d'encodage choisi pour les mots de pascie est le bcrypt avec un cost par défaut (13).

1.3 L'ORM : DOCTRINE

Fichier : app/config/security.yml

providers:

doctrine:

entity:

class: AppBundle:User

property: username

L'application communique avec la base de données via Doctrine, l'ORM par défaut de Symfony. Username est l'attribut identifiant l'utilisateur au sein de la couche sécurité, c'est pourquoi il doit être unique.

1.4 La base de données

Fichier : app/config/parameters.yml

Ce fichier contient les paramètres de connexion à la base de données.

2- Le firewall

Fichier : app/config/ security.yml

```
firewalls:
  main:
    anonymous: ~
    pattern: ^/
    form_login:
      login_path: login
      check_path: login_check
      always_use_default_target_path: true
      default_target_path: /
    logout: ~
```

Notre pare-feu principal s'appelle Main :

➤ anonymous: true

Le site est accessible pour les utilisateurs anonymes (ce qui permet de pouvoir se logger). Les ressources seront protégées par les rôles.

➤ pattern: ^/

Toutes les URL commençant par / (soit l'ensemble du site) sont protégées par le firewall

➤ form_login: Représente la méthode d'authentification

login_path: login	Route du formulaire de connexion
check_path: login_check	Route de validation du formulaire
always_use_default_target_path: true	Redirige systématiquement vers la page par défaut si non connecté
default_target_path: /	Définit la page par défaut : /

➤ logout: ~ : Permet de se délogger

Un firewall « Dev » est également défini, il n'est utilisé que pour certains outils de développement de symfony : ne pas s'en préoccuper.

[CDu]	Spécifications techniques ToDoList	V 1.0
		Page 7 / 10

II AUTORISATION

L'autorisation en quelques mots

L'autorisation est le processus qui va permettre de déterminer si l'utilisateur a le droit d'accéder à telle ou telle ressource de l'application. L'Access Control va comparer les droits de l'utilisateur avec les droits requis pour accéder à la ressource.

En fonction du rôle de l'utilisateur et des paramétrages de l'application, l'utilisateur aura accès ou pas aux différentes pages et/ou informations du site.

Le processus Sécurité – 2^{ème} partie

6. L'utilisateur authentifié génère une requête
7. Le système de sécurité intercepte la requête, vérifie les droits de l'utilisateur en comparaison des droits requis, autorise l'utilisateur si ceux-ci sont suffisant, le renvoie vers une page par défaut sinon.

[CDu]	Spécifications techniques ToDoList	V 1.0
		Page 8 / 10

1- Les rôles

Fichier : app/config/ security.yml

```
role_hierarchy:
    ROLE_ADMIN: ROLE_USER
```

L'application utilise 2 rôles définis ROLE_ADMIN et ROLE_USER.

2- Access Control

Fichier : app/config/ security.yml

```
access_control:
    - { path: ^/login, roles: IS_AUTHENTICATED_ANONYMOUSLY }
    - { path: ^/users, roles: ROLE_ADMIN }
    - { path: ^/tasks, roles: ROLE_USER }
    - { path: ^/, roles: ROLE_USER }
```

La page /login est accessible par tous.

La page d'accueil et les pages « tâches » sont accessibles aux ROLE_USER.

Les pages « users » ne sont accessibles qu'aux ROLE_ADMIN

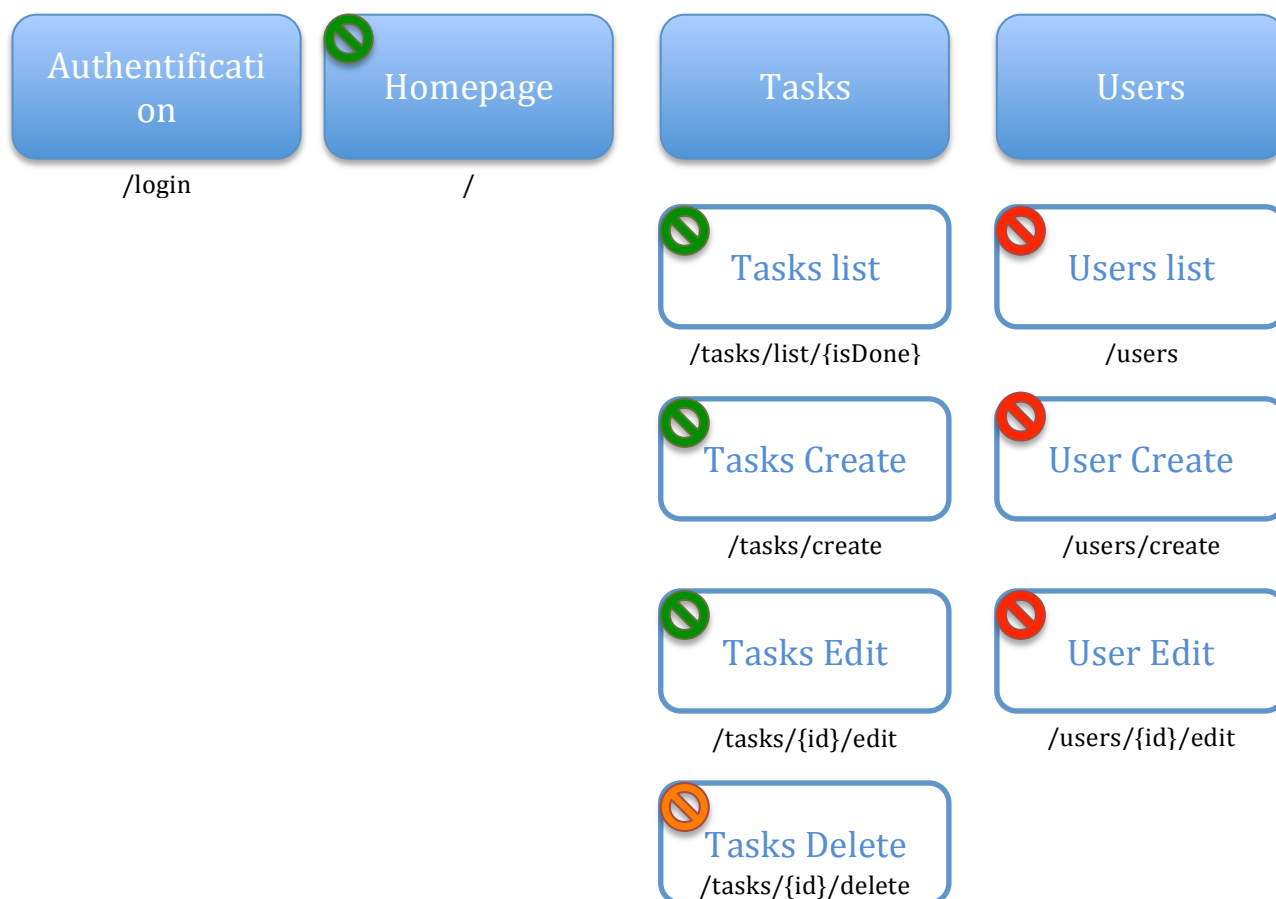
3- Configuration via les actions de contrôleur


Fichier : src/AppBundle/Controller/TaskController.php

Pour répondre à la contrainte « *Les tâches ne peuvent être supprimées que par les utilisateurs ayant créé les tâches en questions.* », une annotation @Security a été ajoutée à l'action deleteAction


```
/**
 * @Route("/tasks/{id}/delete", name="task_delete")
 * @Security("user == task.getAuthor()")
 */
public function deleteTaskAction(Task $task)
```


Annexe 1 :Description générale



 Accès autorisé aux utilisateurs ROLE_USER

 Droits particuliers requis ROLE_USER minimum + auteur de la tâche

 Accès autorisé aux utilisateurs ROLE_ADMIN

Annexe 2 : Profil des acteurs

	Visiteur	Utilisateur	Administrateur
Authentification	X	X	X
Consultation de la liste des taches		X	X
Création d'une tache		X	X
Edition d'une tâche		X	X
Suppression d'une tâche		Si auteur	Si auteur
Création d'un utilisateur			X
Consultation de la liste des utilisateurs			X
Edition d'un utilisateur			X