Rapport de projet

Dashboard pour gestionnaire de tâches et de ressources

Matthieu CROUZET & Tanguy MATHIEU

Contents

I- Introduction

Ce projet consiste à développer une interface de type Single Page Application pour le gestionnaire de ressources et de tâches OAR. Nous avons codé notre application avec le Framework AngularJS qui utilise les technologies HTML et JS. Nous nous sommes servi du Bootstrap SB-Admin pour avoir une interface agréable. Le but de cette interface est de simplifier l'interaction avec OAR. L'objectif est de gérer et de visualiser aisément un ensemble de ressources et de tâches.

II- Cahier des charges

Nous avons pour objectif de développer un portail de type Dashboard.

Pour cela nous devrons utiliser le Bootstrap SB Admin v2.0 pour avoir une interface « user-friendly »

Nous utiliserons l'API REST d'OAR pour gérer notre cluster.

Il y a 3 types d'utilisateurs différents :

- **Non authentifié** : un utilisateur qui pourra seulement voir l'état des ressources et des tâches.
- **Authentifié « docker »** : un utilisateur qui pourra créer et supprimer des tâches. Il pourra aussi regarder l'état des ressources et des tâches.
- **Authentifié « oar »** : un utilisateur qui pourra apporter des modifications aux ressources, ou bien ajouter de nouvelles ressources ou les supprimer. Il pourra aussi regarder l'état des ressources et des tâches.

Notre application devra être ergonomique et simple d'utilisation. Pour cela on utilisera le système d'alerte pour donner des informations aidant l'utilisateur à se servir de notre portail.

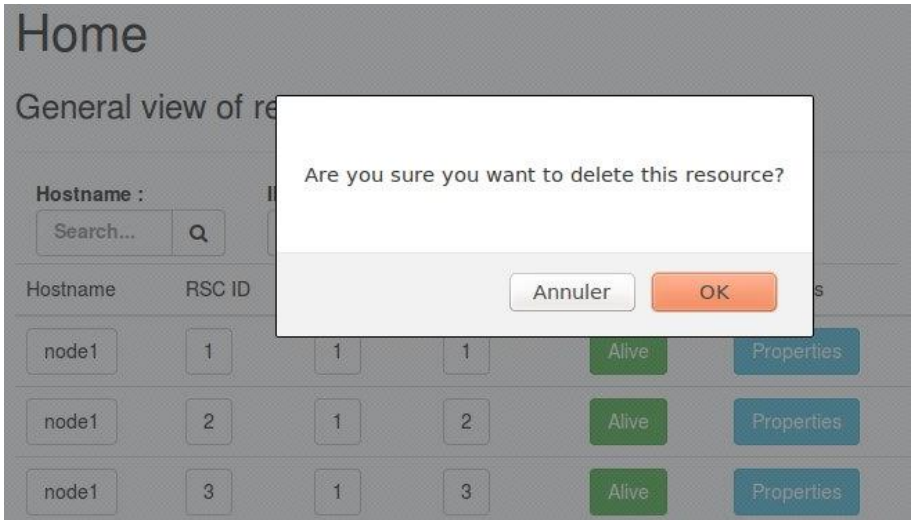


Figure 1 - Exemple de retour pour informer le client

III- Technologies utilisées

1) OAR

OAR est un gestionnaire de ressource et de tâches pour les clusters HPC, et d'autres infrastructures informatiques. L'architecture OAR est établie sur une base de données (PostgreSQL ou MySQL), un langage de script (Perl) et un outil d'administration évolutive en option (par exemple Taktuk).

L'API REST OAR permet d'interagir avec OAR sur http en utilisant une bibliothèque REST. La plupart des opérations généralement fait avec des commandes Unix peuvent être faites en utilisant cette API à partir de n'importe quel langage.

2) AngularJS

AngularJS est un Framework open-source de JavaScript construit et entretenu par Google, ce qui facilite efficacement le développement d'applications web, et standardise les applications côté client en offrant une structure solide et facilement adaptable.

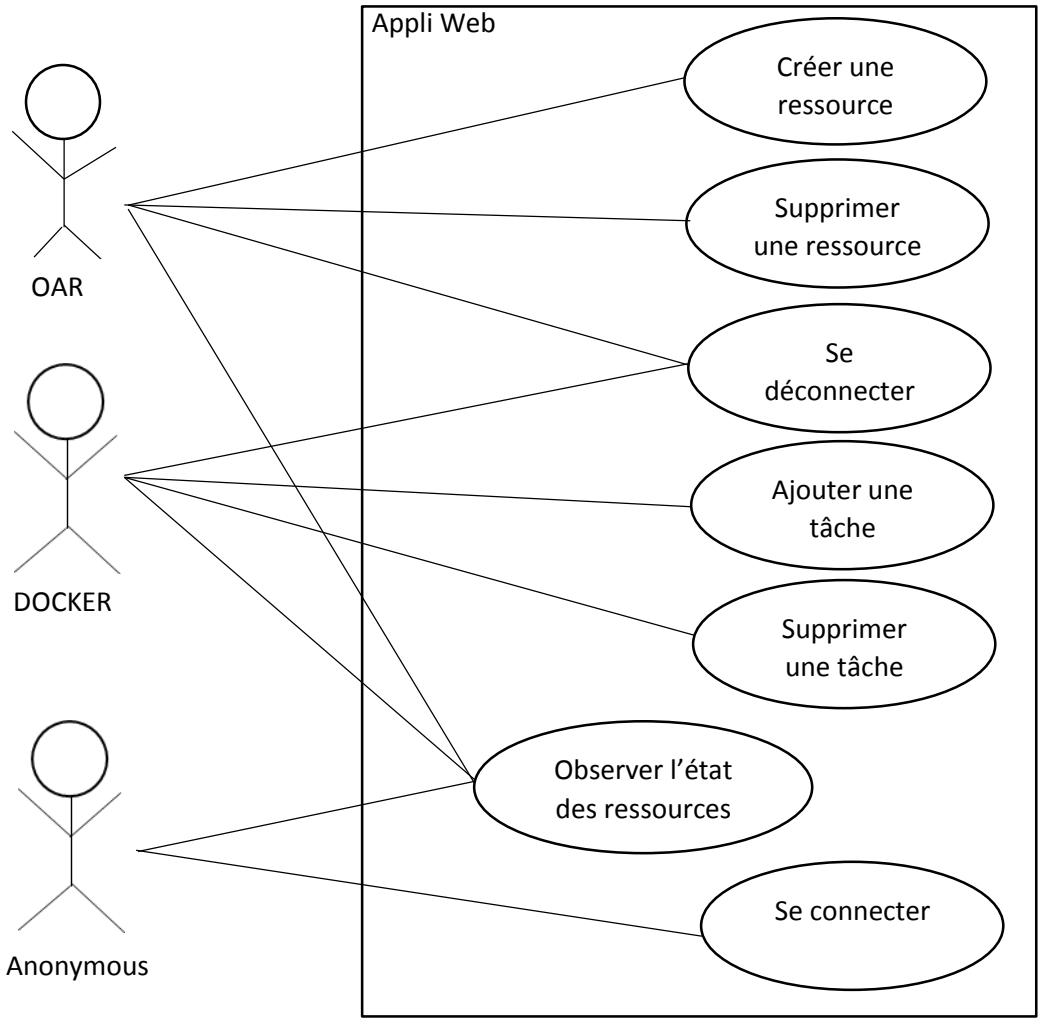
AngularJS est le meilleur choix pour une application web, il favorise particulièrement la création d'éléments visuels, le résultat étant une navigation fluide et rapide sur le site. Ceci explique sa parfaite adaptation pour les Single Page Application, un des objectifs de notre projet.

La plateforme impose un développement selon la structure MVVM (Modèle-Vue-Vue Modèle), design pattern qui reprend la modularité des concepts MVC (Modèle-Vue-Contrôleur). AngularJS adapte et étend le langage HTML traditionnel pour servir le contenu dynamique de manière améliorée grâce à un data-binding bidirectionnel qui permet la synchronisation automatique des modèles et des vues.

3) SB Admin v2.0

SB Admin v2.0 est un Bootstrap gratuit basé sur un thème d'administration ou de tableau de bord. Il est codé en AngularJS ce qui est parfait pour notre application. Il a un design responsive, c'est-à-dire qu'il s'adapte à tous les appareils aussi bien ordinateur fixe que smartphone.
SB Admin v2.0 nous permettra d'avoir une interface « user-friendly » sur laquelle le client pourra travailler, grâce à un beau design garni d'animations.

IV- Fonctionnement général



Pour se connecter l'utilisateur devra remplir le formulaire habituel (Login / Password). L'utilisateur pour se déconnecter en cliquant sur le bouton « Logout » qui n'apparaitra que lorsqu'il est authentifié.

Cette vue ci-dessous est la vue principale. On y trouve toutes les ressources avec les informations les plus importantes, Hostname, ID, état, …

On y retrouve aussi des boutons associés à chacune pour les actions réalisables.

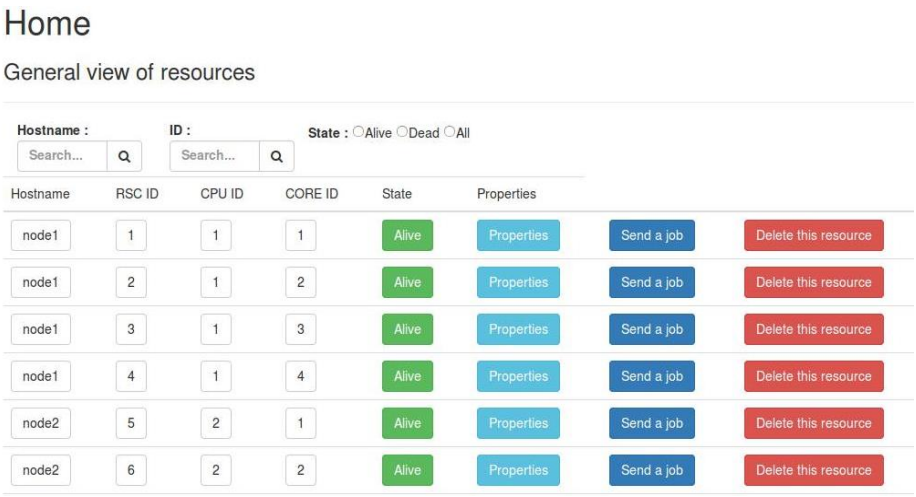


Figure 2- Vue principale

En cliquant sur le bouton « Properties », vous serez redirigé sur une autre page, sur laquelle vous retrouverez beaucoup plus d'informations sur la ressource sélectionnée.

Figure 3 - Détail d'une ressource

En cliquant sur le bouton « Send a job » et si vous êtes enregistré en tant que « docker », vous serez redirigé vers un formulaire à remplir correctement avant de soumettre la tâche. (Encadré en rouge les champs non renseignés et obligatoires)

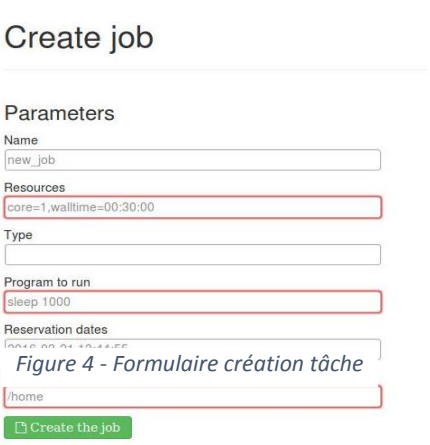


Figure 4 - Formulaire création tâche

Vous pouvez retrouver l'état de toutes les tâches. Mais aussi avoir d'amples informations en cliquant sur le bouton « View Details », qui vous amènera sur une page de la même apparence que celle du détail d'une ressource.

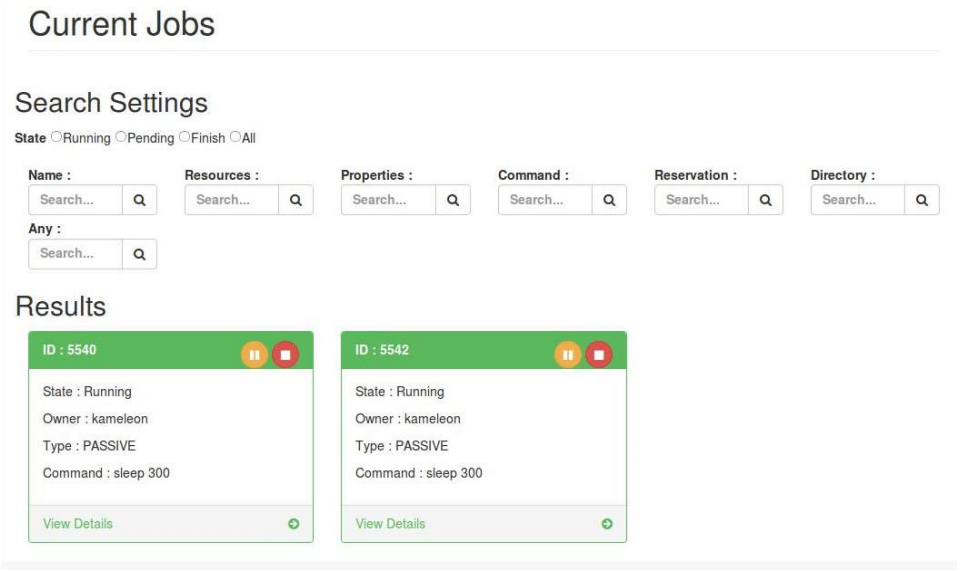


Figure 5 - Etat des tâches

L'utilisateur enregistré en tant qu' « oar » pourra ajouter de nouvelles ressources grâce à ce formulaire. Mais aussi, en supprimer en cliquant sur le bouton « Delete this resource » de la vue principale.

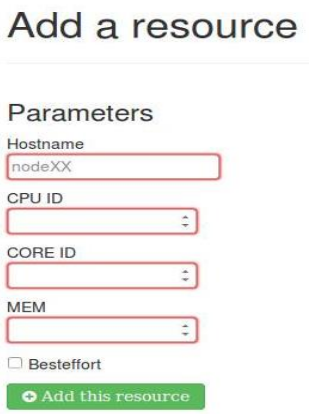


Figure 6 - Formulaire ajout ressource

V- Conclusion

Pour conclure, nous pouvons dire que nous avons eu quelques difficultés pour nous lancer entièrement dans le projet.
Dû au fait que nous ne connaissions pas du tout l'informatique du Web, nous avons dû apprendre les langages JavaScript et HTML. Nous avons appris l'utilisation du Framework AngularJS, qui a été implémenté pour faciliter et rendre dynamique la gestion des vues d'une application Web.
Nous avons dû, aussi, nous familiariser avec l'environnement oar-docker, chose plutôt complexe.

D'autres requêtes possibles avec l'API d'oar-docker pourront parfaire notre projet en affichant d'autres informations utiles, qui ne sont pas lisibles directement depuis l'API.
Comme, par exemple, connaître le pourcentage de ressources en train de réaliser une tâche. Ceci nous permettrait de voir rapidement si le système est bien exploité ou si des ressources sont sous-exploitées.