Compte Rendu Projet
Langage Web 2 - XML

Sommaire

---

1. Introduction

Le code source du projet se trouve à l'adresse suivante :

https://github.com/Ushyr7/rss22SB1/tree/master

Le répertoire src contient le code source du service REST et le répertoire client contient le code source du client React JS.

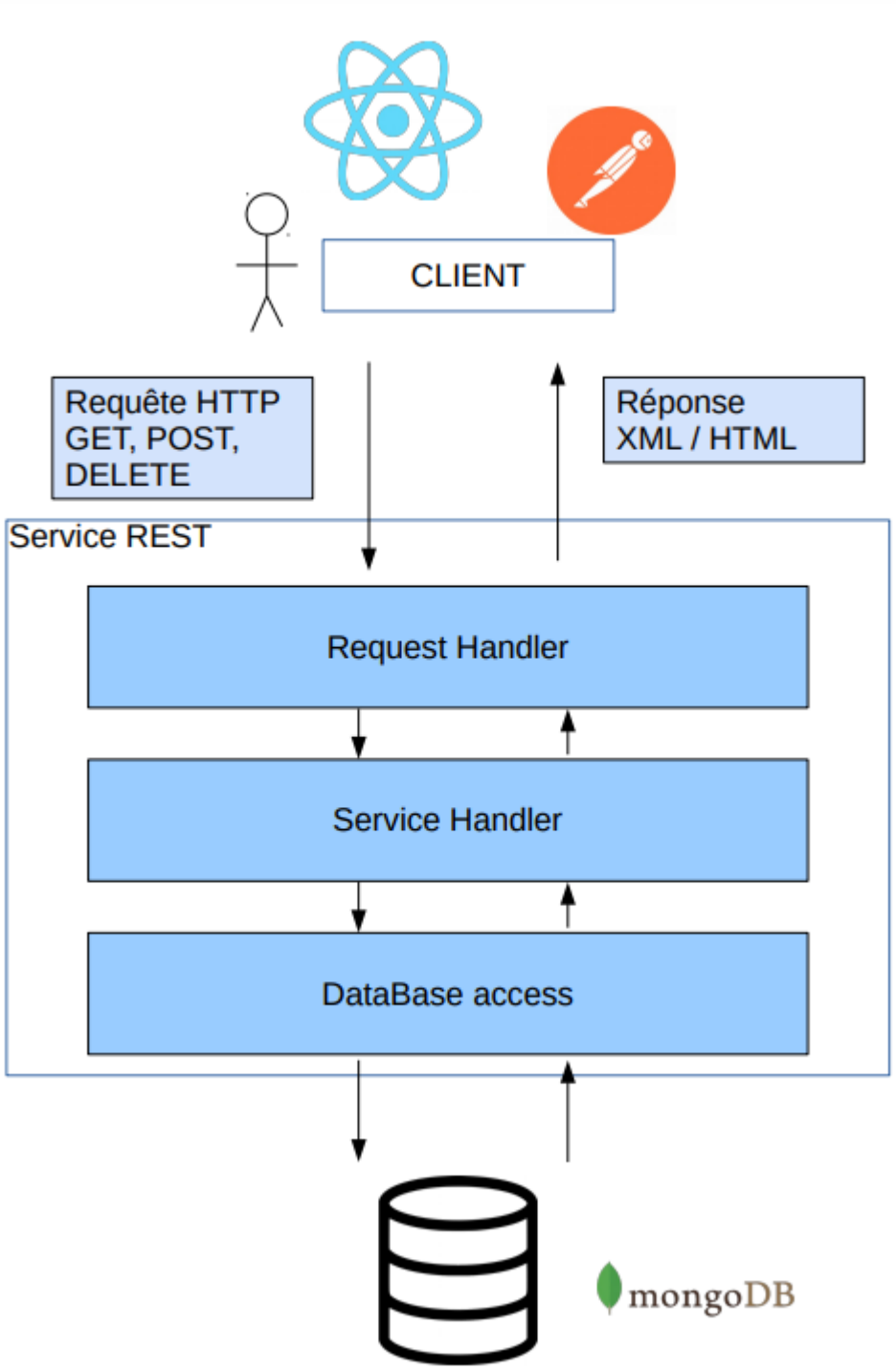
Enfin, un répertoire ressources est mis à disposition, il contient un fichier JSON contenant un test complet du service REST avec PostMan, 2 exemples de fichiers XML conformes à la spécifications RSS22, ainsi qu'une copie de ce compte rendu.

Le service REST déployé se trouve à l'URL :

https://rss22-ridel.cleverapps.io/

2. Architecture du projet

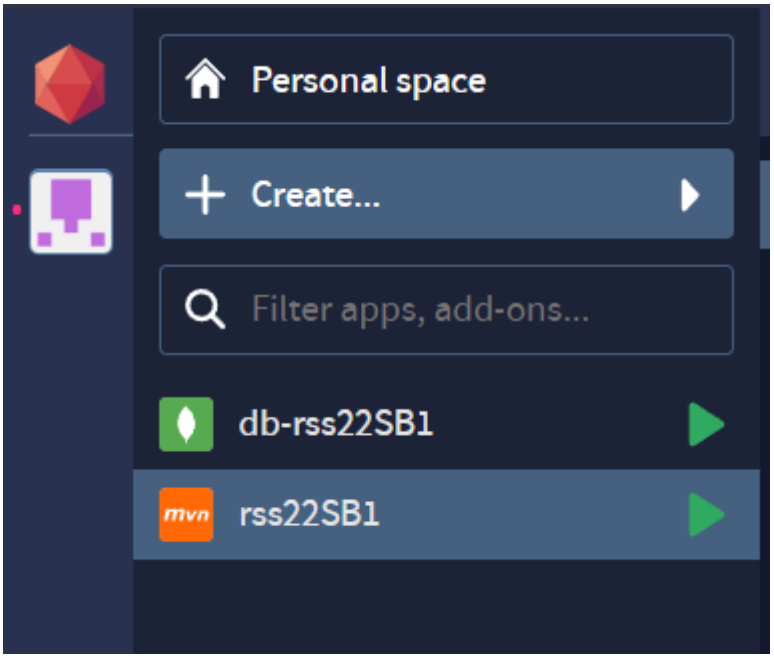
Voici un Diagramme d'architecture du projet :



- Request Handler : cet élément permet la gestion des requêtes http, il est constitué des classes dans le package controllers :
    - DeleteController : le controller qui permet la suppression d'un article dans la base de donnée, "mappé" à l'URL: /rss22/delete/{guid}
    - IndexController : fournit les pages d'accueils et d'aide mappé aux URL : / et /help
    - PostController : le controller qui permet l'insertion d'un flux rss22 dans la base de données, mappé à l'URL: /rss22/insert
    - RSSController : le controller qui permet d'obtenir la liste des articles disponibles avec le verbe HTTP GET, au format XML et HTML, ainsi que l'obtention d'un article complet, encore une fois au format XML ou HTML. mappé aux URL: /rss22/resume/xml /rss22/resume/html /rss22/resume/xml/{guid} /rss22/resume/html/{guid}
- Service Handler: cet élément contient les services et modèles dans le package rss22 :
    - FeedService : une interface représentant les opérations qu'on peut effectuer sur un Feed dans la base de données
    - FeedServiceImpl : implémentation de l'interface FeedService
    - TransformXML: une classe permettant de transformer un flux rss22 en XML en HTML à partir du fichier xslt dans src/main/ressources/xslt/rss22.tp4.xslt
    - XMlValidator: une classe permettant la validation d'un flux xml selon les contraintes des flux rss22 définit dans src/main/ressources/xsd/rss22.xsd
    - Enfin, tout les élements dans le package main.java.model qui sont des classes java permettant de représenter les différents éléments d'un flux rss22.

- DataBase Access: cet élément permet l'accès à la base de données MongoDB, il est composé de l'interface FluxRepository.java ainsi que du fichier application.properties qui contient les informations de connexion à la base de données

MongoDB, MySQL et PostgreSQL sont 3 systèmes de base de données disponibles sur Clever-cloud sous la forme d'add on, ce qui facilite le déploiement. Ce projet était pour moi une opportunité de découvrir MongoDB, à travers un conteneur Docker et l'aide de mongo-express, c'est la raison principale qui justifie mon choix de MongoDB pour ce projet.
Sur Clever-cloud j'ai donc le service REST déployé ainsi qu'une base de données mongoDB accessible :



3. Utilisation du service REST en local

Une version pour l'utilisation locale du projet est disponible sur le dépôt git :

https://github.com/Ushyr7/rss22SB1/tree/local

Pour démarrer le service, il faut d'abord obtenir le conteneur Docker pour MongoDB et mongo-express, pour ça, un fichier .yaml est disponible à la racine du projet. Les ports par défauts sont 8081 et 27017, mais ils sont modifiables dans le fichier docker-compose.yaml

Il suffit de se placer à la racine avec un terminal et d'y lancer la commande :

docker compose -f docker-compose.yaml up -d

On peut ensuite accéder à mongo-express à l'adresse : http://localhost:8081/

Enfin, on peut lancer le service REST avec la commande

mvn spring-boot:run

3. Utilisation du client en local

Une version pour l'utilisation locale du client est, là aussi, disponible sur la branche local du dépôt git :

https://github.com/Ushyr7/rss22SB1/tree/local

Le client dans cette branche est, par défaut, configuré pour se lancer sur le port 5000 et faire appel au service sur l'adresse : http://localhost:8080/

Néanmoins, le client disponible dans la branche master utilise, par défaut, le service déployé (et se lance aussi sur le port 5000).

Pour modifier le port, on peut modifier le fichier .env dans le repertoire client/client-rss22.

Pour ce qui est de l'adresse du service, on peut modifier le champ "proxy" dans le fichier package.json, dans le même répertoire.

L'application cliente est développée en React JS, pour pouvoir l'utiliser il faut d'abord obtenir les dépendances avec **npm**.

Il suffit de se placer dans le répertoire client-rss22, et lancer la commande

npm install

Une fois les dépendances obtenues, la commande **npm start** permet de lancer le service localement.

On devrait obtenir la page suivante :