

**Projet API**

Anthony PELLIZZERI

Professeur : Olivier Perrin

Académie Nancy-Metz

Université de Lorraine

Institut des Sciences du Digital

Management & Cognition

Master 2 MIAGE ACSI

Année 2020-2021

Table des matières

[I. Réalisation 3](#_Toc69065391)

[1. Outils utilisés 3](#_Toc69065392)

[a) Service discovery : Consul 3](#_Toc69065393)

[b) Metrics : Actuator 3](#_Toc69065394)

[c) Base de données : PostgreSQL 3](#_Toc69065395)

[d) Conteneurisation : Docker 3](#_Toc69065396)

[2. Découpage micro-services 3](#_Toc69065397)

[a) Service Utilisateur 3](#_Toc69065398)

[b) Service dégradation 4](#_Toc69065399)

[c) Service OAuth 5](#_Toc69065400)

[II. Amélioration 5](#_Toc69065401)

[1. Gateway 5](#_Toc69065402)

[2. Hystrix (circuit breaker) 5](#_Toc69065403)

[3. RabbitMQ 5](#_Toc69065404)

[4. Base de données 5](#_Toc69065405)

[III. Conclusion 5](#_Toc69065406)

# Réalisation

## Outils utilisés

### Service discovery : Consul

L’outil de service discovery Consul a été mis en place pour pouvoir récupérer toutes les instances de micro-services lancées. Cela permet de faciliter l’utilisation de service comme

feign qui communique avec Consul pour aller chercher une instance disponible et faire sa requête.

Ce service a été déployé à l’aide d’un docker, pas de configuration à faire sur le service mais de la configuration pour les services qu’on souhaite y enregistrer.

### Metrics : Actuator

Spring Boot Actuator est un sous projet (sub-project) du projet Spring Boot. Il est créé pour collecter, superviser les informations de l'application. Afin de superviser cette application, il faut accéder aux endpoint (points finaux) qui sont disponibles dans Spring Boot Actuator. Il est également possible d’en créer de nouveaux.

### Base de données : PostgreSQL

Pour ce qui est des bases de données, PostgreSQL a été utilisé, chaque micro-service possède sa base de données : utilisateur, cours, oauth. Ces bases de données sont peuplées au lancement du micro-service qui la gère. Elles sont lancées dans des Docker par le biais d’un docker-compose.

L’utilisation de PostgreSQL s’est faite sur la réflexion de plusieurs instances du micro-service, il faut que les instances du micro-service utilisateur partagent les mêmes informations, c’est donc pour ça qu’une base de données PostgreSQL a été utilisée à la place d’une base de données in-memory H2 par exemple.

### Conteneurisation : Docker

Docker a grandement été utilisé, que ce soit pour le lancement des services nécessaires comme Consul mais aussi pour le lancement des différents micro-services.

Dans les deux cas, il facilite le déploiement des applications car en un fichier il est possible de lancer tous les micro-services d’une application.

## Découpage micro-services

Le découpage en micro-service s’est fait sous découpage métier. D’abord un micro-service

Utilisateur qui permet la gestion des utilisateurs, un micro-service Degradation qui permet la gestion des dégradations et des actions sur celle-ci. Un micro-service OAuth a été mis en place également pour tout ce qui concerne la sécurité.

Chaque micro-service est protégé par OAuth avec l’utilisation de la technologie JWT, il est possible de savoir si un utilisateur est authentifié dans un micro-service ou non. Les micro-service métier comme utilisateur et dégradation ont également hateoas pour avoir des liens pour chaque requête.

### Service Utilisateur

Utilisateur-service est par définition le micro-service de gestion des utilisateurs. Il est connecté à la base de données PostgreSQL utilisateur. Il permet de créer, modifier, supprimer, récupérer un ou plusieurs utilisateurs. Un validator et des annotations de validation sont présentes sur les classes Input, pour que les données restent cohérentes en base de données et également éviter les erreurs de sauvegarde en base de données.

Il a également des routes permettant de référer un utilisateur dans le service oauth en tant que citizen. Pour se faire, un client feign a été mis en place pour communiquer directement avec le micro-service OAuth et ajouter l’utilisateur.

Une autre route a également été créée afin de lister les dégradations d’un citizen. Pour se faire, un client feign a été mis en place pour communiquer directement avec le micro-service degradation et récupérer les degradation de cet utilisateur

### Service dégradation

Dégradation est le micro-service de gestion des dégradations. Il est connecté à la base de données PostgreSQL dégradation. Il permet de créer, modifier, supprimer, récupérer des dégradations. Un validator et des annotations de validation sont présentes sur les classes

Input, pour que les données restent cohérentes en base de données et également éviter les erreurs de sauvegarde en base de données.

Une dégradation est associée à plusieurs actions qui vont constituer le workflow de celle-ci.

Les traitements se font dans des services afin de dissocier l’aspect métier du reste du code.

Chaque méthode est décorée afin de gérer les droits d’accès.

La partie métier présent dans le processus métier (BPMN) est respecté.

Ainsi il est possible de créer une demande de dégradation par une personne non authentifié.

Lors d’une création de dégradation, deux actions vont directement être crées :

* Action pour le début de la dégradation et renseigner le responsable (par défaut « ROLE\_ANONYMOUS » s’il n’est pas connecté, sinon son. ID présent dans le JWT)
* Passage à l’état étude.

Une action peut être finalisé seulement par le responsable de cette action ou une personne aillant le rôle « ROLE\_CLERK » si l’action est à l’état « ETUDE »

A savoir qu’une action sera créée seulement si la dernière action est finalisée

Si la dernière action est à l’état « ETUDE », le « ROLE\_CLERK » pourra choisir :

* La rejeter, création d’une action à l’état « REJET »
* La valider, création d’une action à l’état « APPROUVEE »
* Demander plus de détails, création d’une action à l’état « ETUDE\_DETAILLEE »

Si la dernière action est à l’état « ETUDE\_DETAILLEE », le rôle « ROLE\_MANAGER » pourra choisir :

* Valider l’action, création d’une action à l’état « ETUDE »
* Refuser l’action, création d’une action à l’état « APPROUVEE »

Une dégradation peut être modifié par son créateur tant qu’elle est à l’état de « DEBUT »

A savoir également que si une dégradation est « INACTIVE », il ne sera plus possible de la modifier ainsi que ses actions.

NB : le rôle « ROLE\_ADMIN » a tous les droits

### Service OAuth

OAuth-service est le micro-service de gestion de la sécurité. Il est connecté à la base de données PostgreSQL OAuth. Il permet de créer un user pour qu’il puisse se connecter à l’API.

Des annotations de validation sont présentes sur la classe Input, pour que les données restent cohérentes en base de données et également éviter les erreurs de sauvegarde en base de données.

Plusieurs rôles ont été créés afin de respecter la partie métier, il s’agit des rôles ROLE\_ADMIN, ROLE\_CITIZEN, ROLE\_CLERK, ROLE\_MANAGER, ROLE\_ERP, ROLE\_EMPLOYE, ROLE\_PARTENER.

.

# Amélioration

## Gateway

Mettre en place un micro-service Gateway afin de faire du loadbalancing, cela permettra de généraliser les routes d’accès et faire de la haute disponibilité.

## Hystrix (circuit breaker)

Mettre en place des circuits breaker comme hystrix pour éviter des retours d’éventuels problèmes

## RabbitMQ

Utiliser un système de message afin d’éviter de perdre des informations si un des services est tombé.

## Base de données

L’amélioration possible pour les bases de données serait de clustériser chaque base de données utilisée par micro-service. Cela permet d’éviter la surcharge d’une seule base de données si le service relié est très demandé et également le single point of failure car si une base de données vient à tomber, les instances du service relié s’en trouvent impactées.

# Conclusion

J’ai trouvé ce projet très intéressant car mettre en place une architecture micro-service depuis le début jusqu’à la finalité avec un exemple concret est très enrichissant. Cela m’a permis de voir de nouvelles technologies.

J’ai eu un petit peu de difficulté à réaliser ce projet ce qui m’a permis d’augmenter mes connaissances et avoir une certaine fierté à sa finalisation.