|  |
| --- |
| Emsi Centre |
| Rapport Projet Micro-Service en Jee |
| Gestion de facture |

|  |
| --- |
| Youssef Filali Khattabi  19/02/2021 |

Table des matières

[Table des matières 2](#_Toc60219796)

[Introduction 3](#_Toc60219797)

[Enoncé 4](#_Toc60219798)

[Conception 5](#_Toc60219799)

[1.Architecture des Micro-Services 5](#_Toc60219800)

[2.Diagramme de classe de l’application 5](#_Toc60219801)

[Code 6](#_Toc60219802)

[Screenshots d’exécution 7](#_Toc60219803)

[Conclusion 9](#_Toc60219804)

# Table des matières

[Figure 1 : Architecture des Micro-Services 5](#_Toc60219745)

[Figure 2 : Diagramme de classe de l'application 5](#_Toc60219746)

[Figure 3 : Clients enregistrés dans eureka 7](#_Toc60219747)

[Figure 4 : Affichage des clients 7](#_Toc60219748)

[Figure 5 : Affichage des informations sur le client dont id est 1 7](#_Toc60219749)

[Figure 6 : Affichage des produits 7](#_Toc60219750)

[Figure 7 : Affichage des informations du produit dont l’id est 1 7](#_Toc60219751)

[Figure 8 : Affichage de la facture numéro 1 8](#_Toc60219752)

[Figure 9 : Affichage des factures du client 1 8](#_Toc60219753)

[Figure 10 : Login et récupération de tokens 8](#_Toc60219754)

[Figure 11 : Utilisation de l'access token pour récupérer des données 8](#_Toc60219755)

[Figure 12 : Expiration de l'access token 8](#_Toc60219756)

[Figure 13 : Récupération de l'access token à l'aide du refresh token 8](#_Toc60219757)

# Introduction

Dans le cadre de l’Architecture des Applications Micro Services de la formation EMSI (Ecole Marocaine des Sciences de l’Ingénieur) de 5eme année un projet est à réaliser pour mettre à pratique les connaissances acquises et découvrir le fonctionnement d’une application moderne.

Une application Micro-Services consiste à découper une grande application en petit bout de codes, des programmes métier indépendant l’un à l’autre développés, testés et déployés séparément et qui peuvent être écrit en langage différent. Ces Micro-Service sont gérés par d’autre Micro-Services fonctionnels de gestion d’application.

L’idée consiste à réaliser un programme de gestion de factures qui sera organisé comme suit :

* Un Micro-Service Customer Service
* Un Micro-Service Inventory Service
* Un Micro-Service Billing Service
* Un Micro-Service Gateway de gestion de traffic
* Un Micro-Service Eureka Discovery Service comme annuaire d’application

- Un Micro-Service d'authentification basée sur Spring Security et Json Web Token

Ainsi qu’une partie dont le but est de sécuriser les parties Front et Backend d’une application avec Keycloak.

Ce travail se présentera comme suit : d’abord un énoncé ensuite une partie conception, une partie code, une partie screen-shots d’exécution puis une partie conclusion.

Ce document fait office du rapport de projet de l’étudiant Filali Khattabi Youssef

# Enoncé

## Architecture micro-services

1. Créer le micro service Customer-service

* Créer l’entité Customer
* Créer l’interface CustomerRepository basée sur Spring Data
* Déployer l’API Restful du micro-service en utilisant Spring Data Rest
* Tester le Micro service

2. Créer le micro service Inventory-service

* Créer l’entité Product
* Créer l’interface ProductRepository basée sur Spring Data
* Déployer l’API Restful du micro-service en utilisant Spring Data Rest
* Tester le Micro service

3. Créer la Gateway service en utilisant Spring Cloud Gateway

4. Créer l’annuaire Discovery Service basé sur NetFlix Eureka Server

6. Créer le service Billing Service

7. Créer un service d’authentification Statless basé sur Spring Security et Json Web Token. Ce service devrait permettre de :

* Gérer les utilisateurs et les rôles de l’application
* Authentifier un utilisateur en lui délivrant un access Token et un refresh Token de type JWT
* Gérer les autorisation d’accès
* Renouveler l’access Token à l’aide du refresh Token

## Keycloak

1.Prise en main

* Télécharger Keycloak 12.0.1, démarrer Keycloak
* Créer le Realm “my-eco-realm", créer un nouveau client sécuriser
* Créer des utilisateurs, créer des routes, affecter les rôles aux utilisateurs
* Tester l’authentification des utilisateurs en utilisant un client Rest comme ARC
  + Authentification avec le mot de passe
  + Authentification avec le refresh-token
* Personnaliser le paramétrage des timeout des tokens

2.Récupérer et tester le code source des deux applications à sécuriser à partir du repository Github (<https://github.com/mohamedYoussfi>)

A) “products-app-kc": Application Frontend basée sur thymeleaf qui permet de gérer des produits et des fournisseurs

B) “supplieurs-service-kc": Application Backend Micro-service permettant de gérer les fournisseurs

3. Sécurisé l’applicationFrontend en mode public client

4. Sécuriser le micro-service en mode barer only

5. Liez le frontend avec le backend en utilisant KeycloakRestController pour afficher la liste des fournisseurs de la partie Frontend

6. Personaliser la sécurité,de la patie frontend en ajoutant les autres fonctionnalités fournies par Keycloak

* Auto-inscription des utilisateurs
* Politique des mots de passe
* Double authentification OTP
* Etc …

7.Faire de même pour developper et sécuriser la partie Frontend avec Angular

# Conception

## 1.Architecture des Micro-Services

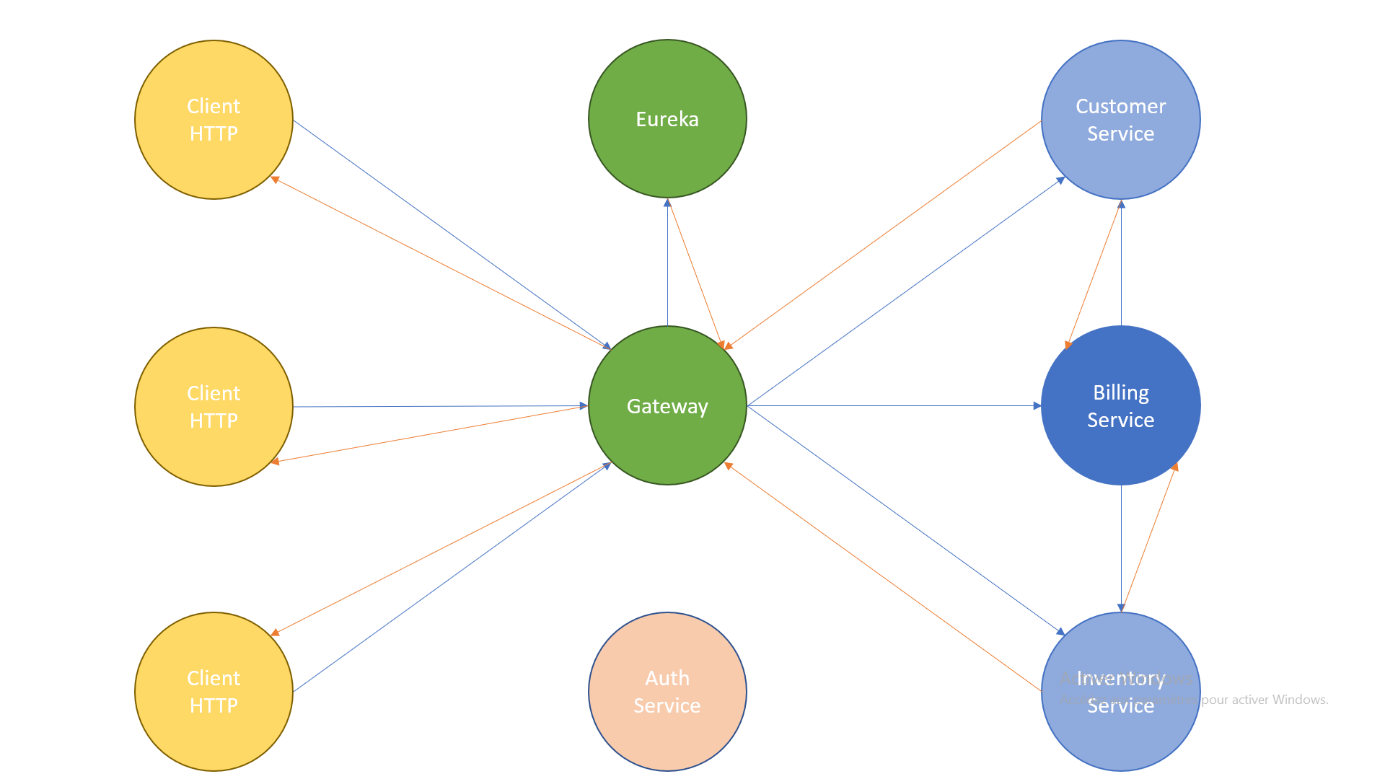


Figure 1 : Architecture des Micro-Services

## 2.Diagramme de classe de l’application

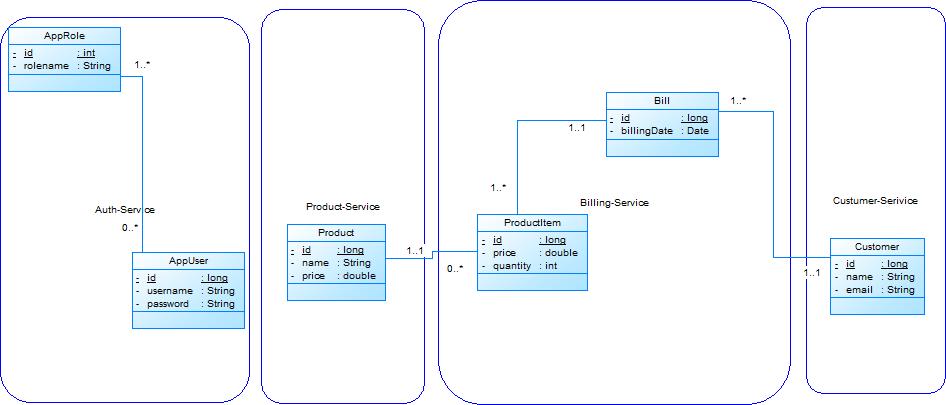


Figure 2 : Diagramme de classe de l'application

# Code

|  |  |  |
| --- | --- | --- |
| **Micro-Service** | **Description** | **Lien GitHub** |
| **Customer-Service** | Gère les clients | https://github.com/GAliNor/costumer-service |
| **Inventory-Service** | Gère les produits | https://github.com/GAliNor/Inventory-Service |
| **Gateway-Service** | Gère le Traffic, consulte l’annuaire et renvoi la requête au bon micro-service | https://github.com/GAliNor/gateway-service |
| **Eureka-Service** | Annuaire, recapitule les informations sur les micro-services | https://github.com/GAliNor/Eureka-Server |
| **Billing-Service** | Gère factures, communique avec Customer-Service et Inventory-Service | https://github.com/GAliNor/Billing-Service |
| **Auth-Service** | Gere authentification à l’aide de la technologie JWT | https://github.com/GAliNor/Auth-Serv |

|  |  |  |  |
| --- | --- | --- | --- |
| **Keycloak** | Sécuriser les parties Front et Backend d’une application avec Keycloak | | https://github.com/GAliNor/Angular-Keycloak-Microservices |
| Screenshots d’exécution   Figure 3 : Clients enregistrés dans eureka | | | | |
| Figure 4 : Affichage des clients | | Figure 5 : Affichage des informations sur le client dont id est 1 | | |
| Figure 6 : Affichage des produits | | Figure 7 : Affichage des informations du produit dont l’id est 1 | | |
| Figure 8 : Affichage de la facture numéro 1 | | Figure 9 : Affichage des factures du client 1 | | |
| Figure 10 : Login et récupération de tokens | | Figure 11 : Utilisation de l'access token pour récupérer des données | | |
| Figure 12 : Expiration de l'access token | | Figure 13 : Récupération de l'access token à l'aide du refresh token | | |
| Figure 14 : Formulaire d’authentification | | Figure 15 : Affichage produits | |
| Figure 16 : Affichage Fournisseurs | | | |

# Conclusion

Ce projet a été l’occasion d’appréhender les principes de base d’une architecture micro-services et de les mettre en pratique dans différentes situations de conception.