

# Projet Application WEB Orientée Services

Réalisé par : Ruben ALBERTINI , Samar RHILANE

Encadré par : M. Mouloud MENCEUR

## 1. Objectif

Le projet consiste à créer deux micro-services. Le micro-services :

**CompteBancaire** se caractérise par {"IBAN" : "FR7630004000031234567890143", "type de compte" : "Compte courant", "interet" : 0.0, "frais de tenue de compte" : "gratuit" }

**OperationBancaire** se caractérise par {"id operation" : 12345, "type operation" : "VIREMENT", "IBAN source" : "FR7630004000031234567890143", "IBAN destination" : "USD", "montant" : 1000.0, "date" : "2021-12-30"}

Les deux microservices communiquent par l'envoi d'une requête HTTP

Le client utilisé est POSTMAN où nous pouvons réaliser des requêtes HTTP (POST, GET, PUT, DELETE)

**Key words** : JAVA, SpringBoot, Docker, Junit, H2, REST API, Kubernetes

## Installation et Utilisation

```
$ git clone --single-branch --branch ruben  
https://github.com/samarrhilane/MicroService1_CompteBancaire
```

```
$ git clone --single-branch --branch ruben  
https://github.com/samarrhilane/MicroService2_OperationBancaire
```

```
mvn clean install package
```

## Set up the Spring Boot Application

```
java -jar target/ CompteBancaire-0.0.1-SNAPSHOT.jar
```

```
java -jar target/ OperationBancaire -0.0.1-SNAPSHOT.jar
```

## Set up Docker

```
docker build -t CompteBancaire-0.0.1-SNAPSHOT.jar
```

```
docker build -t OperationBancaire -0.0.1-SNAPSHOT.jar
```

## Set up Kubernetes

```
CompteBancaire>minikube start --driver=docker
```

```
Terminé ! kubectl est maintenant configuré pour utiliser "minikube" cluster et espace de noms "default" par défaut.
```

```
CompteBancaire>kubectl cluster-info
```

```
Kubernetes control plane is running at https://127.0.0.1:57577  
CoreDNS is running at https://127.0.0.1:57577/api/v1/namespaces/kube-system/services/kube-dns:dns/proxy
```

```
CompteBancaire>kubectl run comptebancaire --image=comptebancaire:latest \ --port=8080 --image-p
```

```
ull-policy Never  
pod/comptebancaire created
```

```
\CompteBancaire>kubectl create deployment comptebancaire --image=k8s.gcr.io/echoserver:
```

```
deployment.apps/comptebancaire created
```

```
\CompteBancaire>kubectl expose deployment comptebancaire --type=NodePort --port=8080
```

```
service/comptebancaire exposed
```

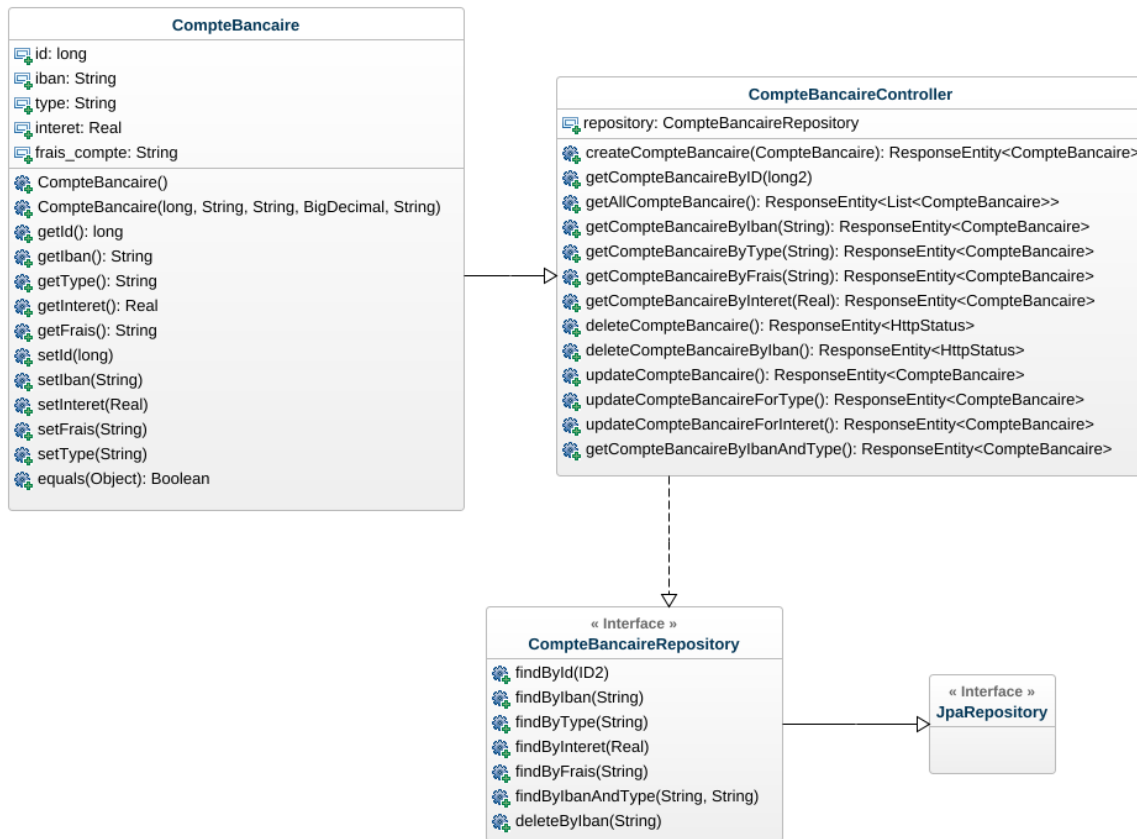
```
CompteBancaire>kubectl get services comptebancaire
```

NAME	TYPE	CLUSTER-IP	EXTERNAL-IP	PORT(S)	AGE
comptebancaire	NodePort	10.100.217.129	<none>	8080:30279/TCP	4m

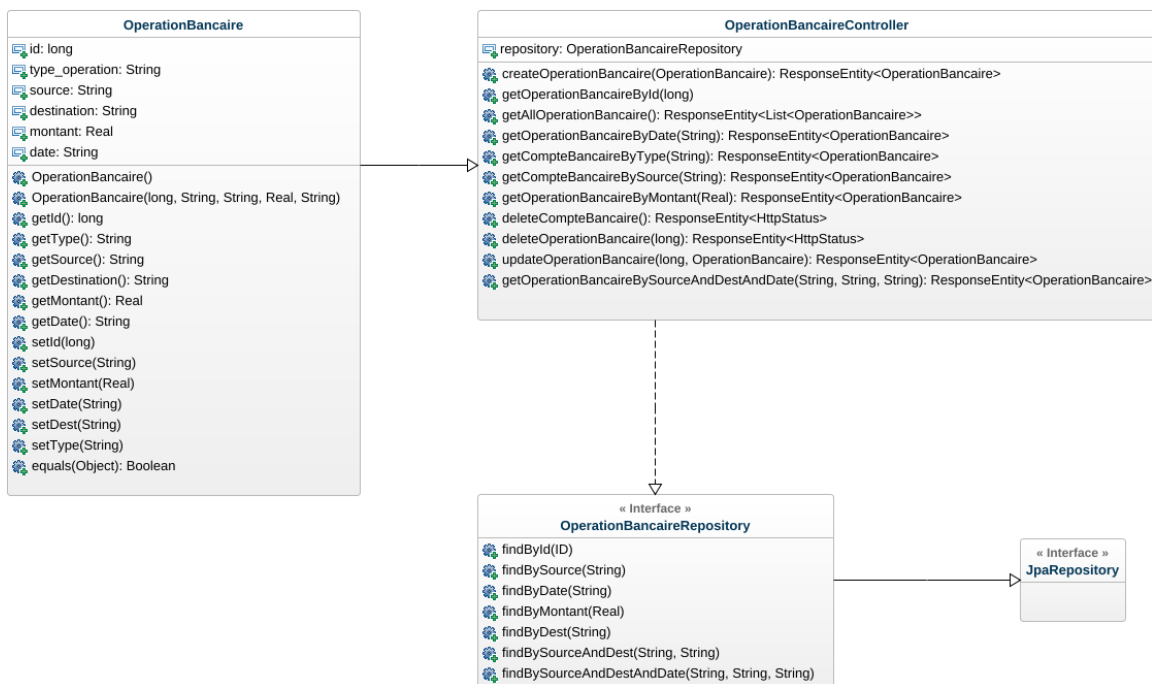
```
\CompteBancaire>minikube service comptebancaire
```

NAMESPACE	NAME	TARGET PORT	URL
default	comptebancaire	8080	http://192.168.49.2:30279

## 2. CompteBancaire



## 3. OperationBancaire



## 4. REST API

### GET

Récupérer tous les comptes bancaires

```
"http://localhost:8080/COMPTEBANCAIRE"
```

Récupérer une opération bancaire

```
"http://localhost:8088/operation-bancaire"
```

Récupérer le compte bancaire en spécifiant l'IBAN

```
"http://localhost:8080/COMPTEBANCAIRE/iban/FR7630004000031234567891011"
```

Récupérer le compte bancaire en spécifiant le type du compte

```
"http://localhost:8080/COMPTEBANCAIRE/type/Compte_courant"
```

Récupérer le compte bancaire en spécifiant une valeur d'intérêt du compte

```
"http://localhost:8080/COMPTEBANCAIRE/interet/10"
```

Récupérer une opération bancaire en spécifiant le montant

```
"http://localhost:8088/operation-bancaire/montant/1000"
```

Récupérer une opération bancaire en spécifiant une date

```
"http://localhost:8088/operation-bancaire/date/2022-06-21"
```

### DELETE

Supprimer un compte bancaire en spécifiant l'identifiant

```
"http://localhost:8080/COMPTEBANCAIRE/id/1"
```

Supprimer un compte bancaire en spécifiant l'IBAN

```
"http://localhost:8080/COMPTEBANCAIRE/iban/FR7630004000031234567891011"
```

## Exemple de GET :

Afin d'afficher tous les comptes bancaires

GET

http://localhost:8080/COMPTEBANCAIRE

ParamsAuthorizationHeaders (7)BodyPre-request ScriptTestsSettings

Query Params

	KEY	VALUE	DESCRIPTION
	Key	Value	Description

BodyCookiesHeaders (5)Test Results

Status: 200 OK

PrettyRawPreviewVisualizeJSON

```
1  {
2    "id": 1,
3    "iban": "FR7630004000031234567890143",
4    "type": "Compte courant",
5    "interet": 0.00,
6    "frais": "gratuit"
7  },
8  {
9    "id": 2,
10   "iban": "FR7630004000031234567891011",
11   "type": "Compte courant",
12   "interet": 1000.00,
13   "frais": "gratuit"
14 }
15
16
```

## Exemple de POST :

Afin de rajouter un nouveau compte bancaire

http://localhost:8080/COMPTEBANCAIRE/

POST

http://localhost:8080/COMPTEBANCAIRE/

ParamsAuthorizationHeaders (9)BodyPre-request ScriptTestsSettings

none

form-data

x-www-form-urlencoded

raw

binary

GraphQL

JSON

1 {
2 "id": 3,
3 "iban": "FR763000400003123456789991",
4 "type": "Compte courant",
5 "interet": 3000.00,
6 "frais": "gratuit"
7 }

BodyCookiesHeaders (5)Test Results

Status: 200 OK

PrettyRawPreviewVisualizeJSON

```
1  {
2    "id": 3,
3    "iban": "FR763000400003123456789991",
4    "type": "Compte courant",
5    "interet": 3000.00,
6    "frais": "gratuit"
7  }
```

## Exemple de PUT :

Afin de modifier une opération bancaire prenant l'identifiant 3 (modification du montant)

Rest / <http://localhost:8088/operation-bancaire/>

PUT <http://localhost:8088/operation-bancaire/id/3>

Params Authorization Headers (9) **Body** Pre-request Script Tests Settings

☐ none ☐ form-data ☐ x-www-form-urlencoded ☒ raw ☐ binary ☐ GraphQL **JSON** [v](#)

```
1 {
2   "type": "VIREMENT",
3   "source": "FR7630004000031234567898888",
4   "dest": "FR76100110002012345678909999",
5   "montant": 1111,
6   "date": "2022-06-23",
7   "id": 3
8 }
```

Body Cookies Headers (5) Test Results [v](#) Status: 200 OK

Pretty Raw Preview Visualize **JSON** [v](#) [↻](#)

```
1 {
2   "type": "VIREMENT",
3   "source": "FR7630004000031234567898888",
4   "dest": "FR76100110002012345678909999",
5   "montant": 1111,
6   "date": "2022-06-23",
7   "id": 3
8 }
```

## Exemple GET : operation DELETE : operation

## Exemple

Rest / <http://localhost:8088/operation-bancaire/>

GET <http://localhost:8088/operation-bancaire/>

Params Authorization Headers (7) **Body** Pre-request Script

☒ none ☐ form-data ☐ x-www-form-urlencoded ☐ raw ☐ binary

Body Cookies Headers (5) Test Results

Pretty Raw Preview Visualize **JSON** [v](#) [↻](#)

```
1 {
2   "type": "VIREMENT",
3   "source": "FR7630004000031234567890143",
4   "dest": "FR7610011000201234567890188",
5   "montant": 1000.0,
6   "date": "2021-12-30",
7   "id": 1
8 }
9
10 {
11   "type": "VIREMENT",
12   "source": "FR7610011000201234567890188",
13   "dest": "FR7630004000031234567890143",
14   "montant": 2000.0,
15   "date": "2022-06-23",
16   "id": 2
17 }
18 }
```

DELETE <http://localhost:8088/operation-bancaire/> [Send](#) [v](#)

Params [v](#) [...](#)

	KEY	VALUE	DE	...	Bulk Edit
	Key	Value			Description

Body [v](#) [v](#) 204 No Content 13 ms 112 B [Save Response](#) [v](#)

Pretty [v](#) Text [v](#) [↻](#)

1

Rest / <http://localhost:8088/operation-bancaire/>

GET <http://localhost:8088/operation-bancaire/id/1>

Params Authorization Headers (7) **Body** Pre-request Script Tests Settings

	KEY	VALUE	DESCR
	Key	Value	Description

Body Cookies Headers (4) Test Results [v](#) 404 Not Found

Pretty Raw Preview Visualize **Text** [v](#) [↻](#)

1