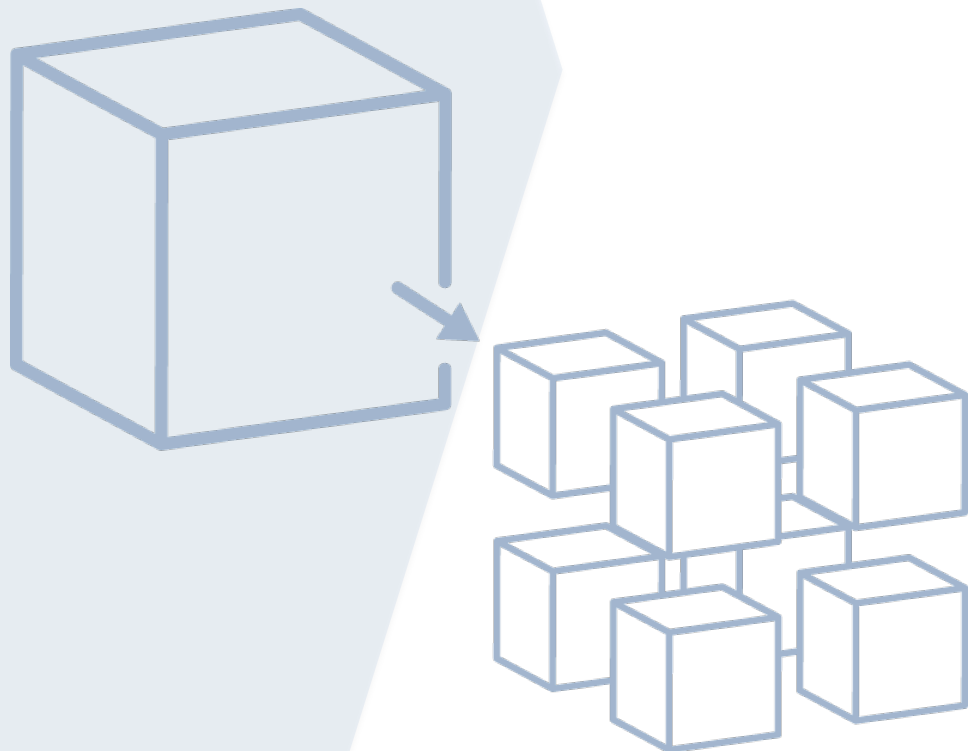


PROJET ARCHITECTURE MICROSERVICES

Quoc Huy VO – Linh Linh THAI



A/ Installation et compilation :

1. Compiler :

Dans git hub, vous avez 2 répertoires représentant 2 microservices à cloner:

<https://github.com/VOQuocHuy/CompteBancaire-microservice>

<https://github.com/VOQuocHuy/Transaction-microservice>

Ensuite, vous ouvrez le command (ou terminal si vous utilisez Mac OS) puis tapez :

```
cd chemincourantduprojet/  
mvn clean install
```

2. Exécuter :

Vous construisez 2 images Docker à partir du Dockerfile dans 2 microservices :

```
docker build -f Dockerfile -t CompteBancaire-microservice
```

```
docker build -f Dockerfile -t Transaction-microservice
```

Vous pouvez vérifier les images Docker en indiquant la commande suivante :

```
docker images
```

Enfin, vous lancez chaque container :

```
docker run -d -p 8000 :8000 CompteBancaire-microservice
```

```
docker run -d -p 8001 :8001 Transaction -microservice
```

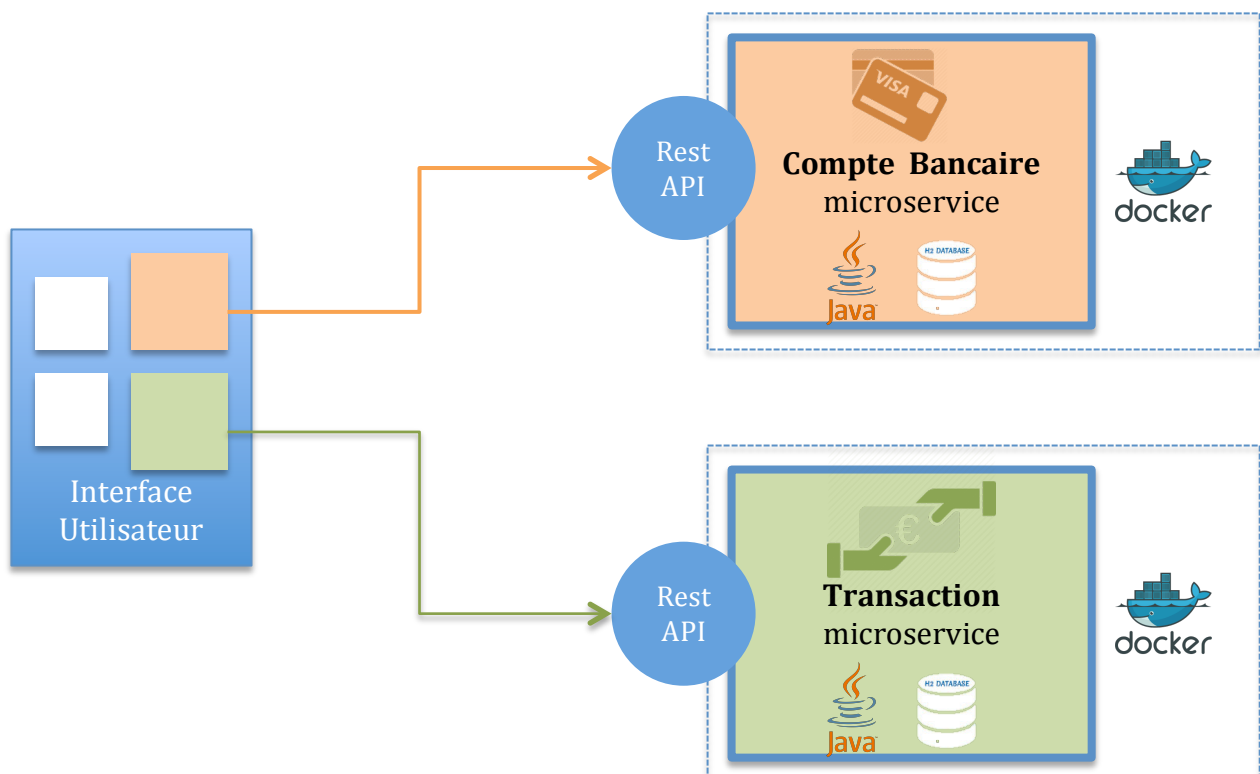
B/ Documentation technique

1. Description du logiciel :

Notre but est de créer une application java web permettant de réaliser les opérations bancaires. Autrement dit, cette application permettra aux utilisateurs de consulter, créer, lister ou modifier, supprimer les comptes bancaires.

Chaque compte bancaire doit fournir les informations tels que IBAN, type de compte, frais de tenue de compte,...En plus, les utilisateurs peuvent réaliser des virements bancaires (transférer de l'argent d'un compte à un autre) dans le logiciel.

2. Architecture Microservices:



Comme dans le schéma ci-dessus, on découpe cette application en 2 petits services ou bien Microservices (Compte Bancaire et Transaction) qui fonctionnent de manière autonomes : l'un expose une API REST que l'autre pourra consommer.

Chaque microservice a sa propre base de données, son propre serveur d'application. La plupart du temps ces Microservices sont chacun dans un container Docker.

3. Technologies utilisées :



Cette application est codé en java et utilise le framework SpringBoot.

« Spring Boot est un framework qui facilite le développement d'applications fondées sur Spring en offrant des outils permettant d'obtenir une application packagée en jar, totalement autonome. »

SGBD choisi pour ce projet est H2. Cette base propose des interfaces de programmations (APIs) SQL et JDBC.

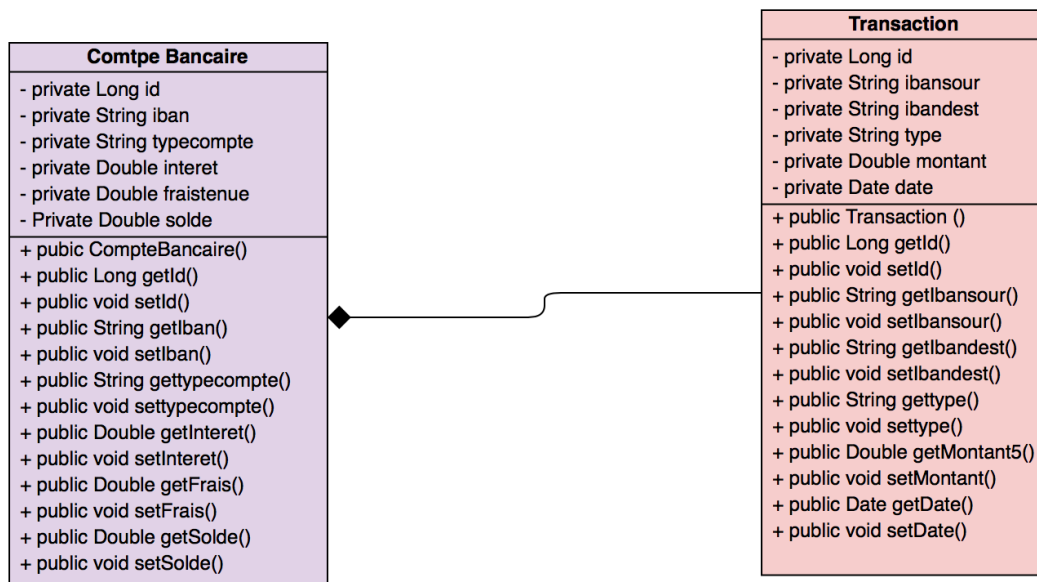
L'environnement de développement intégré (IDE) pour programmer en java est Eclipse. C'est un IDE facile, extensible et rapide pour coder. Pour ce projet, on doit installer dans Eclipse quelques plug-ins tels que Maven, Docker Tools,...

On utilise Postman pour lancer les API (Get, Delete, Post, Put).

Quelques tests ont été effectués avec JUnit dans Eclipse.




« JUnit est un framework open source pour le développement et l'exécution de tests unitaires automatisables.

4. Diagramme de classes :



Il y a 2 classes dans notre diagramme : « Compte Bancaire » et « Transaction ».
Chaque classe est un ensemble de fonctions et attributs et utilisé pour un microservice.

C/ Bilan du projet

 Les avantages	<ul style="list-style-type: none">▪ Comprendre le fonctionnement de Microservices▪ Connaître le framework StringBoot▪ Connaître Docker▪ Connaître le SGBD H2
 Les réussites	<ul style="list-style-type: none">▪ Premier programme avec IDE Eclipse▪ Coder en Java▪ Ecrire back-end pour une application Microservices▪ Savoir utiliser StringBoot▪ Savoir créer images Dockers
 Les difficultés	<ul style="list-style-type: none">▪ Ne pas maîtriser Java▪ Rester des problèmes non résolues▪ Construire Microservices (compliqué pour la première fois)▪ Apprendre plusieurs nouveaux logiciels en même temps