BANK ACCOUNT PROJECT

Document technique

|  |  |
| --- | --- |
| **Rédacteur** | **Maine ben adesmad** |
| **Date de création** | **Dimanche 11 Octobre 2020** |
| **Version** | **001** |

SOMMAIRE

[1. User stories du Kata 3](#_Toc53336170)

[2. Environnement technique 4](#_Toc53336171)

[3. Conception et architecture 5](#_Toc53336172)

[3.1 Modèle de données UML 5](#_Toc53336173)

[3.2 Architecture 6](#_Toc53336174)

# User stories du Kata

* Création d'une API : Application bancaire
* Création d'un service pour retirer/déposer une somme d'argent d'un compte bancaire
* Création d'un service pour créer un virement d'un compte bancaire à un autre
* Pour chaque virement entres deux comptes, création d'un historique pour chacun
* Création d'un service pour afficher l'historique des transactions d'un compte bancaire

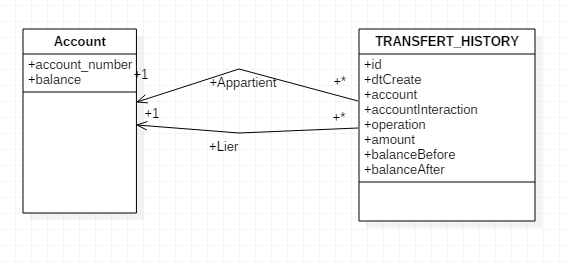
# Environnement technique

L'api est développée avec le langage de programmation JAVA la version 8.

|  |  |
| --- | --- |
| Technologie | Utilité |
| Maven | Faciliter et automatiser les différentes manipulations d'un projet Java |
| Spring-boot | * Framework léger, il m’a permis de réduire la complexité de configuration du projet spring via les starters. * Il m’a offert la possibilité de créer des API Rest et de les déployés rapidement sans autres composants. * Facilité de la réalisation des tests fonctionnels. |
| JPA | Afin de construire les modèles de notre application, avec des annotations pour la définition des entités et leurs relations, et pour éviter les spécificités des ORM |
| JpaRepository de spring-data | Utiliser ces méthodes prédéfinies : écrire beaucoup moins de code |
| Spring Rest | Pour la création des web services |
| Hibernate-validator | Pour la validation des paramètres des requêtes |
| Swagger | Visualiser et tester les ressources : avec seulement des dépendances simple et une annotation qui active l’ajout des dépendances de swagger et une annotation on peut profiter d’une interface web qui offre la possibilité |
| Junit | Pour le lancement des tests |
| Base de données H2 | simple et facile à exploité |

# Conception et architecture

## Modèle de données UML



TRANSFER\_HISTORY : présente l’historique des virements entre un compte et un autre.

* account : le compte propriétaire de TRANSFERT\_HISTORY
* accountInteraction : le deuxième compte qui fait partie du virement
* operation : retrait ou dépôt
* amount : montant du virement
* balanceBefore : solde avant le virement
* balanceAfter : solde après virement

Il existe deux relations unidirectionnelles de TRANSFER\_HISTORY vers ACCOUNT :

1. Un TRANSFER\_HISTORY est la propriété d’un et un seul ACCOUNT
2. Un TRANSFER\_HISTORY est relié à un et un seul ACCOUNT (pour savoir l’autre partie du virement)

## Architecture

Client web

swagger

REST Controller

Base de données H2

Entity

Repository

Service