





E









E-banking-App


-  Project overview
-  Repository
- 

Issues

0
- 


Merge requests

0
-  Requirements
-  CI/CD
-  Security & Compliance
-  Operations
-  Packages & Registries
-  Analytics
-  Wiki
- 

Snippets
- 

Members

Abdellah AHBANE > E-banking-App > Wiki > Transaction Service

Last edited by  **Abdou Ahban** 11 months ago

Page history

New page

Transaction Service



All what you need to run this service is to

- CREATE database transactions .
- Add database password if you have one !
- Run RabbitMQ server
- Run the Registry service eureka & the account-service && Customer-service.

If you take a look at the main application :

```
@SpringBootApplication
@EnableFeignClients
public class TransactionServiceApplication implements CommandLineRunner {

    public static void main(String[] args) {
        SpringApplication.run(TransactionServiceApplication.class, args);
    }

    @Autowired
    VirementRepository virementRepository;

    @Autowired
    RechargeRepository rechargeRepository;

    @Override
    public void run(String... args) throws Exception {

        virementRepository.save ( new Virement ( null, new Date ( ), 10000d,12287689L, 1228768909L," Bec
        virementRepository.save ( new Virement ( null, new Date ( ), 30000d,12237689L, 1228761909L," Bec
        virementRepository.findAll ().forEach ( System.out::println );

        rechargeRepository.save ( new Recharge ( null, new Date ( ), 100d,12287689L, "IAM", " 0602468922
        rechargeRepository.save ( new Recharge ( null, new Date ( ), 30d,12237689L, "Orange", " 087639002
        rechargeRepository.findAll ().forEach ( System.out::println );

    }
}
```

This will simply create the table in DB and insert those data into it. But if u stop running it and re-run it again, it will insert them again, you will have duplicated data!

So you need, either

- to drop the database by DROP database transactions Or
- to edit the main application by this :

```
@SpringBootApplication
@EnableFeignClients
public class TransactionServiceApplication {

    public static void main(String[] args) {
        SpringApplication.run(TransactionServiceApplication.class, args);
    }

}
```

I recommend the first one, by dropping just the database especially if you have IntelliJ, it's easy and quick.

- PS. You can apply this to all my codes.

APIs of this Service :


NB: In this service: host = http://localhost:8083/api


The service has two type of Transactions: virement && recharge please check the package Entities to know more about each one. We've APIs for each type of account.

Virements :

GET :

- host/virements : To get the list of all virement. (ADMINs are the only ones allowed to call this)
- host/virements/{id} : To get a virement with the given id. (The ADMINs or the CLIENT who owns the accountNum, are the only ones allowed to call this)
- host/account/virements/{accountNum} : To get all virements of a customer with the given accountant. (CLIENT who owns the

 Clone repository

 Edit sidebar

AA Documentation for using Gateway !

- Account Service
- Agency Service
- Authentication Service
- Customer Service
- GlobalPrameters Service
- Officer Service
- Request Service
- Transaction Service

accountNum is the only one allowed to call this)

- `host/account/3LastVirements/{accountNum}` : To get `last 3` virements of a customer with the given accountNum. (**CLIENT who owns the accountNum is the only one allowed to call this**)
- `host/account/10LastVirements/{accountNum}` : To get `last 10` virements of a customer with the given accountNum. (**CLIENT who owns the accountNum is the only one allowed to call this**)
- `host/account/3LastMonthsVirements/{accountNum}` : To get all virements of a customer - in `last 3 months` - with the given accountNum. (**CLIENT who owns the accountNum is the only one allowed to call this**)
- `host/account/incomeOutcome/{accountNum}` : To get `Resume of` virements of a customer with the given accountNum. (**CLIENT who owns the accountNum is the only one allowed to call this**)

In this format :

```
// The first one ( month 6 in this case ) is the current month:
// for example if we're in 13/6/2020: this will return from 1 to 13 in month 6

{
  "income": null,
  "outcome": null,
  "month": 6,           // This means that there's no virement registered in this month.
  "accountNum": 1111,
  "balance": null
},
{
  "income": 0.0,
  "outcome": 300.0,
  "month": 5,
  "accountNum": 1111,
  "balance": 183052.0
},
{
  "income": 0.0,
  "outcome": 320.0,
  "month": 4,
  "accountNum": 1111,
  "balance": 183052.0
}
```

POST :

- `host/virements` : to create a transaction, the given data should be in this format: (**CLIENT who owns the accountNum in the Json sent below, is the only one allowed to call this**)

For example :

```
{
  "amount": 10000.0,
  "accountNum": 76528932992,
  "destinationAccountNum": 7652899972,
  "motif": " Because I WANT HAHA :p "
}
```

Recharges:

GET :

- `host/recharges` : To get the list of all recharges.
- `host/recharges/{id}` : To get a recharge with the given id.
- `host/account/recharges/{accountNum}` : To get all recharges of a customer with the given accountNum. (**CLIENT who owns the accountNum is the only one allowed to call this**)
- `host/account/3LastRecharges/{accountNum}` : To get `last 3` recharges of a customer with the given accountNum. (**CLIENT who owns the accountNum is the only one allowed to call this**)
- `host/account/10LastRecharges/{accountNum}` : To get `last 10` recharges of a customer with the given accountNum. (**CLIENT who owns the accountNum is the only one allowed to call this**)
- `host/account/3LastMonthsRecharges/{accountNum}` : To get all recharges of a customer - in `last 3 months` - with the given accountNum. (**CLIENT who owns the accountNum is the only one allowed to call this**)

POST :

- `host/recharges` : to create a transaction, the given data should be in this format: (**CLIENT who owns the accountNum in the Json sent below, is the only one allowed to call this**)

For example :

```
{
  "amount": 100.0,
  "accountNum": 76528932992,
```

⏪ Collapse sidebar

```
"operator": "IAM",
"numRecharge": " 0602468922 "
}
```

