Banking-Services-Platform Documentation

The Microservice was developed using Java language and Spring boot framework. Also used the databases Mysql and Mongo db and Rabbit MQ for email notifications. The platform contains 3 microservices, one discovery service and one api-gateway services which are

1. **Eureka-server-service**
2. **Api-gateway-service**
3. **Account-management-service**
4. **Transaction-processing –service**
5. **Customer-support-service**

The services communicate together and they have one entry point which is the api-gateway service. In terms of the databases, three services are connected to the databases

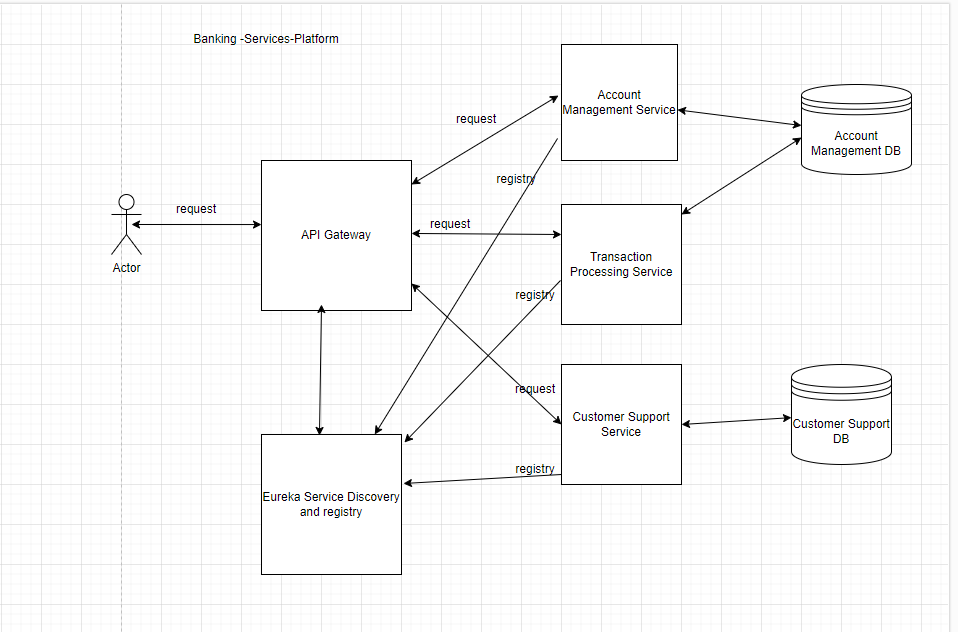
1) Account-management service – Mysql

2) Transaction-processing-service Mysql

3) Customer-support- Mongo db

Both the Account-management-service and Transaction\_processing connect to one database account\_management\_db and Customer-support connect to customersupportdb.

Below is the diagram to show the setup of the banking-services-platform.



To start the system below are the requirements in stegs

1. Install and setup Rabbit MQ to use defaults below url and credentials as below

host=127.0.0.1

port=5672

username=guest

password=guest

1. Setup Mysql database to run on below properties and create database *account\_management\_db*

MySql Properties

url=jdbc:mysql://localhost:3306

username=root

password=

1. Setup Mongo database to run on below properties and create a one document as ticket.

Mongo db Properties

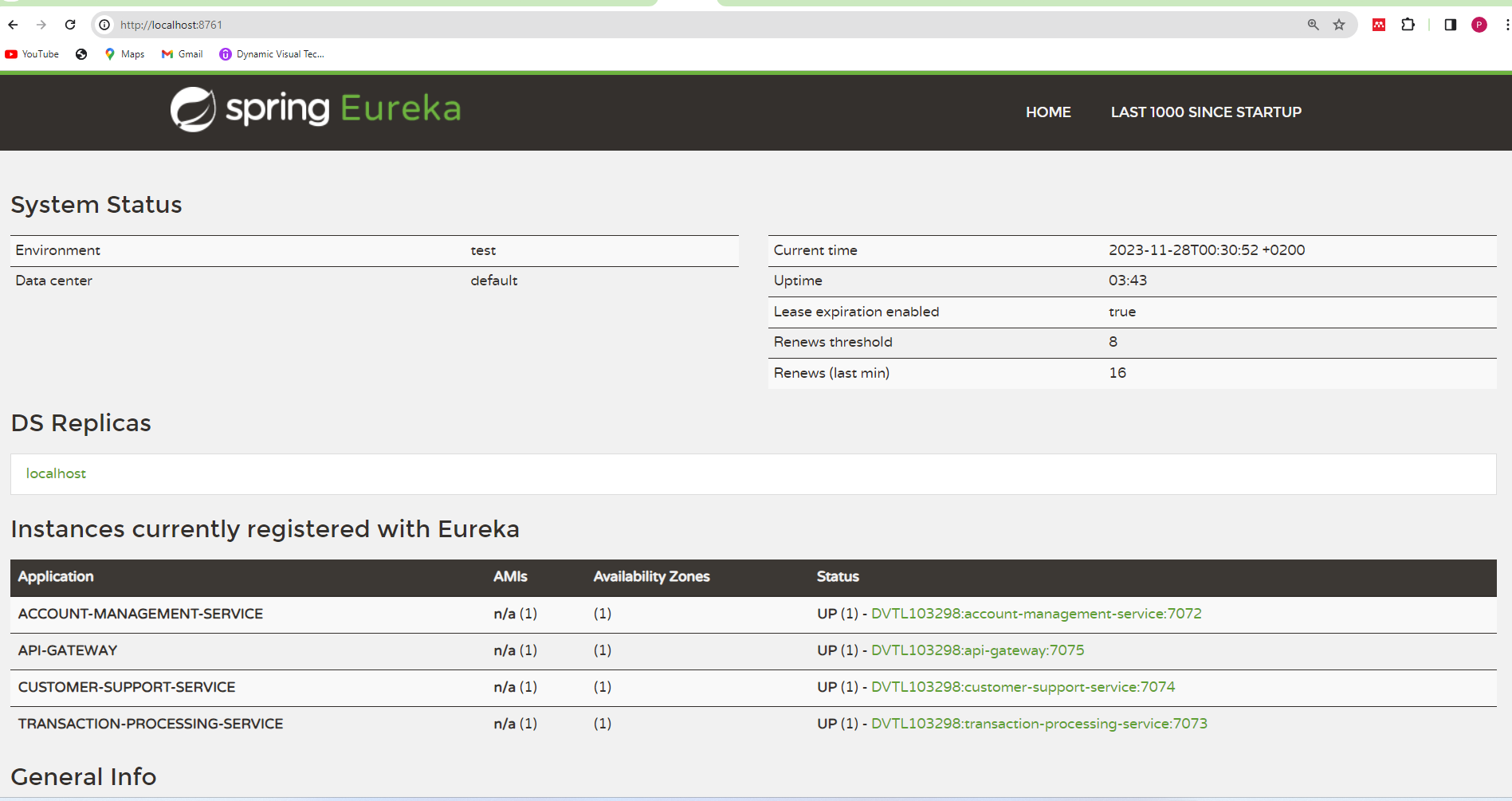
host=localhost

port=27017

1. Clone the application using the link <https://github.com/pridemore/banking-services-platform.git>
2. Download all the Maven dependences required and build the application
3. Start the Eureka services first by running the main class **EurekaServerApplication**
4. Start the API gateway by running the main class **ApiGatewayApplication**
5. Start the Account management service by running main class **AccountManagementApplication**
6. Start the Transction Processing service by running the **TransactionProcessingApplication**.
7. Finally start the Customer support service by running the main class **CustomerSupportApplication**.

Check if all services are registered on the Eureka using the url <http://localhost:8761/>

Below interface should be visible if all started well.



Below are the APIs in exposed using the gateway base url as below:

Gateway base url <http://localhost:7075>

**NB:** All requests requires a Bearer token as **S61Whhzh2X**

1. Create UserDetails API </account-management-service/api/v1/userDetail/create>

Http Method: **POST**

Sample request:

{

  "address": "4536 Mazowe",

  "dob": "1978-01-11",

  "gender": "Female",

  "name": "Rutendo",

  "nationalId": "09-46759N09",

  "phoneNumber": "++23467847845",

  "surname": "Terera",

  "email":"ruvimbo@gmail.com"

}

Sample response:

{

    "statusCode": 200,

    "message": "Success",

    "result": {

        "userDetailId": 4,

        "name": "Rutendo",

        "surname": "Terera",

        "nationalId": "09-46759N09",

        "gender": "Female",

        "dob": "1978-01-11",

        "address": "4536 Mazowe",

        "phoneNumber": "++23467847845",

        "email": "ruvimbo@gmail.com",

        "accounts": **null**,

        "status": "ACTIVE",

        "dateCreated": "2023-11-27 22:52",

        "lastUpdated": "2023-11-27 22:52"

    }

}

1. Linking user Details to accounts API </account-management-service/api/v1/account/create>

Http Method: **POST**

Sample request:

{

  "accountNumber": "46475899688",

  "accountType": "SAVINGS",

  "nationalId": "09-46759N09"

}

Sample response:

{

    "statusCode": 200,

    "message": "Success",

    "result": {

        "id": 11,

        "accountNumber": "46475899688",

        "accountType": "SAVINGS",

        "accountBalance": 0.0,

        "status": "ACTIVE",

        "dateCreated": "2023-11-27 23:01",

        "lastUpdated": "2023-11-27 23:01"

    }

}

3)Get userdetails by phoneNumber </account-management-service/api/v1/userDetail/getUserDetail/++23467847845>

Http Method: **GET**

Sample response:

{

    "statusCode": 200,

    "message": "Success",

    "result": {

        "userDetailId": 4,

        "name": "Rutendo",

        "surname": "Terera",

        "nationalId": "09-46759N09",

        "gender": "Female",

        "dob": "1978-01-11",

        "address": "4536 Mazowe",

        "phoneNumber": "++23467847845",

        "email": "ruvimbo@gmail.com",

        "accounts": [

            {

                "id": 11,

                "accountNumber": "46475899688",

                "accountType": "SAVINGS",

                "accountBalance": 0.0,

                "status": "ACTIVE",

                "dateCreated": "2023-11-27 23:01",

                "lastUpdated": "2023-11-27 23:01"

            }

        ],

        "status": "ACTIVE",

        "dateCreated": "2023-11-27 22:52",

        "lastUpdated": "2023-11-27 22:52"

    }

}

4)Get user by account number </account-management-service/api/v1/account/getUserDetailsByAccount/46475899688>

Http Method: **GET**

Sample response:

{

    "statusCode": 200,

    "message": "Success",

    "result": {

        "name": "Rutendo",

        "surname": "Terera",

        "nationalId": "09-46759N09",

        "gender": "Female",

        "dob": "1978-01-11",

        "address": "4536 Mazowe",

        "phoneNumber": "++23467847845",

        "email": "ruvimbo@gmail.com"

    }

}

5)Set user balance </account-management-service/api/v1/account/update>

Http Method: **PUT**

Sample request:

{

    "accountNumber":"46475899688",

    "accountBalance":45000

}

Sample response:

{

    "statusCode": 200,

    "message": "Success",

    "result": {

        "accountNumber": "46475899688",

        "accountBalance": 45000.0,

        "email": "ruvimbo@gmail.com"

    }

}

1. Get Balance </transaction-processing-service/api/v1/transaction/balance/46475899688>

Http Method: **GET**

Sample response:

{

    "statusCode": 200,

    "message": "Success",

    "result": 45000.0

}

1. Deposit funds </transaction-processing-service/api/v1/transaction/deposit>

Http Method: **POST**

Sample request:

{

    "accountNumber":"46475899688",

    "amount":500.0

}

Sample response:

{

{

    "statusCode": 200,

    "message": "Success",

    "result": {

        "id": 68,

        "transactionReference": "TRX9f19afbd11ec4dc2941b72ea716f75ea",

        "accountNumber": "000053635",

        "amount": 500.0,

        "balance": 46700.0,

        "transactionType": "DEPOSIT",

        "status": "ADDED\_TO\_QUEUE",

        "dateCreated": "2023-11-27 22:03",

        "lastUpdated": "2023-11-27 22:03"

    }

}

1. Withdraw funds </transaction-processing-service/api/v1/transaction/withdraw>

Http Method: **POST**

Sample request:

{

    "accountNumber":"46475899688",

    "amount":450.0

}

Sample response:

{

    "statusCode": 200,

    "message": "Success",

    "result": {

        "id": 69,

        "transactionReference": "TRXca9524df2044461c94a420c873cca6ab",

        "accountNumber": "46475899688",

        "amount": 450.0,

        "balance": 44550.0,

        "transactionType": "WITHDRAWAL",

        "status": "ADDED\_TO\_QUEUE",

        "dateCreated": "2023-11-27 23:17",

        "lastUpdated": "2023-11-27 23:17"

    }

}

1. Statement </transaction-processing-service/api/v1/transaction/statement>

Http Method: **GET**

Sample request:

{

    "accountNumber":"46475899688",

    "startDate":"2023-11-26",

    "endDate":"2023-11-28"

}

Sample response:

{

    "statusCode": 200,

    "message": "Success",

    "result": [

        {

            "id": 69,

            "transactionReference": "TRXca9524df2044461c94a420c873cca6ab",

            "accountNumber": "46475899688",

            "amount": 450.0,

            "balance": 44550.0,

            "transactionType": "WITHDRAWAL",

            "status": "COMPLETED",

            "dateCreated": "2023-11-27 23:17",

            "lastUpdated": "2023-11-27 23:17"

        },

        {

            "id": 70,

            "transactionReference": "TRX4cf5e0555c7f444496b66ce62187b500",

            "accountNumber": "46475899688",

            "amount": 500.0,

            "balance": 45050.0,

            "transactionType": "DEPOSIT",

            "status": "COMPLETED",

            "dateCreated": "2023-11-27 23:21",

            "lastUpdated": "2023-11-27 23:21"

        }

    ]

}

10)Create ticket </customer-support-service/api/v1/customerSupport/createTicket>

Http Method: **POST**

Sample request:

{

    "accountNumber":"46475899688",

    "queryDescription":"This is the ticket for account 46475899688 of Rutendo"

}

Sample response:

{

    "statusCode": 200,

    "message": "Success",

    "result": {

        "id": "656509a3e04d5258f7abab05",

        "ticketReference": "QURYa28b1d6a1e804642947f8df27588b63e",

        "accountNumber": "46475899688",

        "queryDescription": "This is the ticket for account 46475899688 of Rutendo",

        "ticketStatus": "OPEN",

        "dateCreated": "2023-11-27 23:26",

        "lastUpdated": "2023-11-27 23:26"

    }

}

11)Update userDetail </account-management-service/api/v1/userDetail/update/4>

Http Method: **PUT**

Sample request:

{

"email": "rutendoterera@gmail.com"

}

Sample response:

{

    "statusCode": 200,

    "message": "Success",

    "result": {

        "userDetailId": 4,

        "name": "Rutendo",

        "surname": "Terera",

        "nationalId": "09-46759N09",

        "gender": "Female",

        "dob": "1978-01-11",

        "address": "4536 Mazowe",

        "phoneNumber": "++23467847845",

        "email": "rutendoterera@gmail.com",

        "accounts": [

            {

                "id": 11,

                "accountNumber": "46475899688",

                "accountType": "SAVINGS",

                "accountBalance": 45050.0,

                "status": "ACTIVE",

                "dateCreated": "2023-11-27 23:01",

                "lastUpdated": "2023-11-27 23:21"

            }

        ],

        "status": "ACTIVE",

        "dateCreated": "2023-11-27 22:52",

        "lastUpdated": "2023-11-27 23:31"

    }

}

12) Update Ticket </customer-support-service/api/v1/customerSupport/updateTicket/656509a3e04d5258f7abab05>

Http Method: **PUT**

**Sample request:**

{

    "queryDescription":"The decription was updated. The last withdraw for this user was ovecharged"

}

**Sample response:**

{

    "statusCode": 200,

    "message": "Success",

    "result": {

        "id": "65632135ed723b3995fbbd0c",

        "ticketReference": "QURY6b488e6969d44ab3b73b6395e8fa7ed8",

        "accountNumber": "0002347467",

        "queryDescription": "The decription was updated. The last withdraw for this user was ovecharged",

        "ticketStatus": **null**,

        "dateCreated": "2023-11-26 12:43",

        "lastUpdated": "2023-11-27 19:56"

    }

}