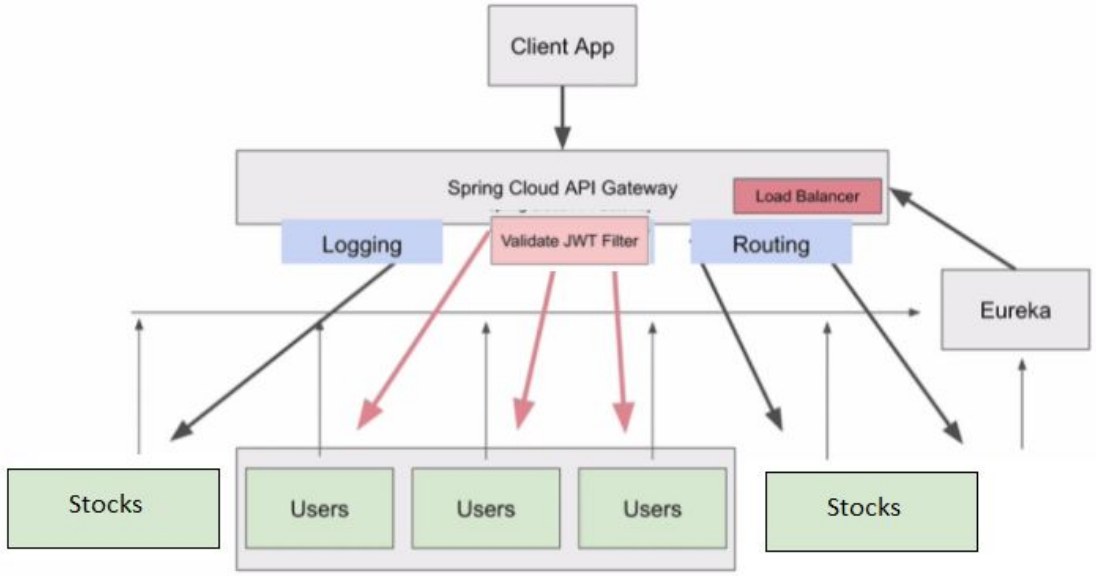
Pay Day Trade App Document

Architectural Diagram



Microservices architecture
Multi-layered (Controller, Service, Repository, Entity, Security .. etc)

MicroServices:

- Spring Cloud API Gateway
- Eureka Discovery Server
- Users
- Stocks

Functionalities

- Sign Up
- Sign In
- List Stocks
- Deposit Cash
- Place an Order

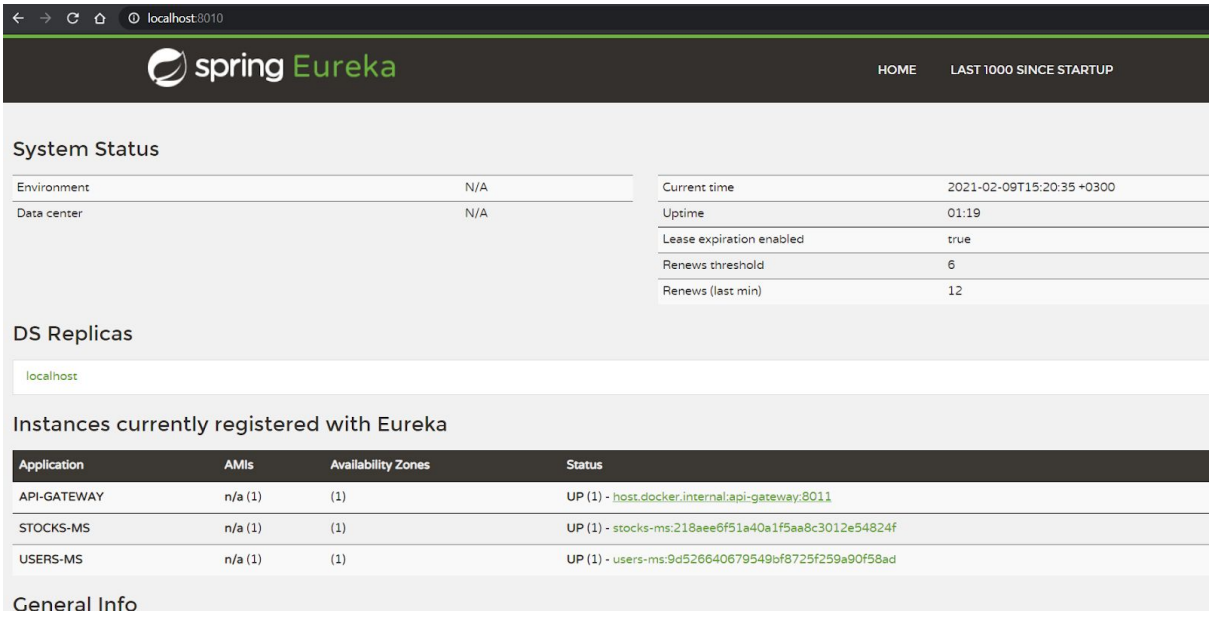
Technologies

- Java 11
- Spring Boot v2.4.2
- Maven
- H2 in-memory database
- Spring Data JPA-Hibernate
- Spring Cloud API Gateway & Load Balancer
- Netflix Eureka Discovery
- Microservices Communication with Feign Client
- Circuit Breaker implementation with Netflix Hystrix
- Spring Security (Authentication, filtering, csrf prevention, password encryption)
- Json Web Token
- Model Mapper for DTO classes
- Lombok for getters & setters
- Logging (Slf4j)
- Spring Boot Actuator for health check metrics
- DevTools (Auto starting services)
- Dockerfile
- Running applications for different environments (Development, Production)

API's

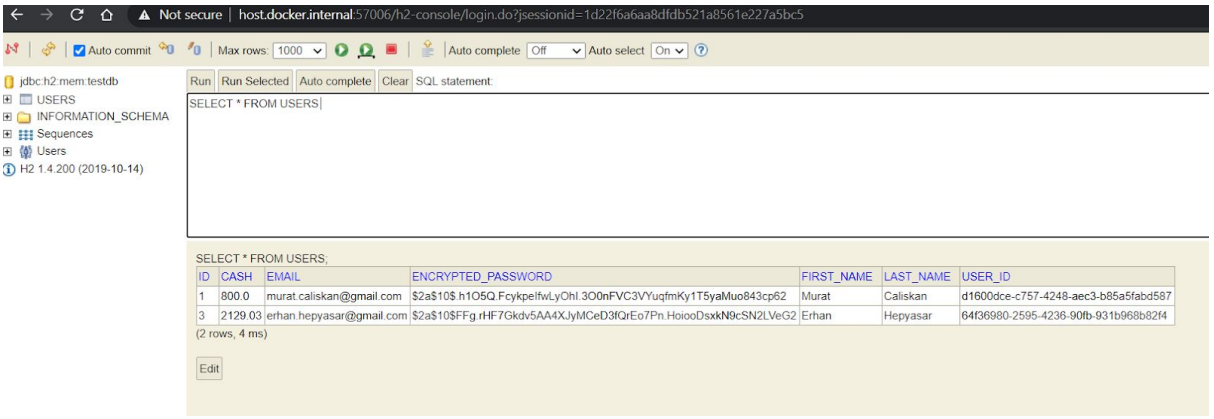
Eukera Discovery Server

http://localhost:8010/

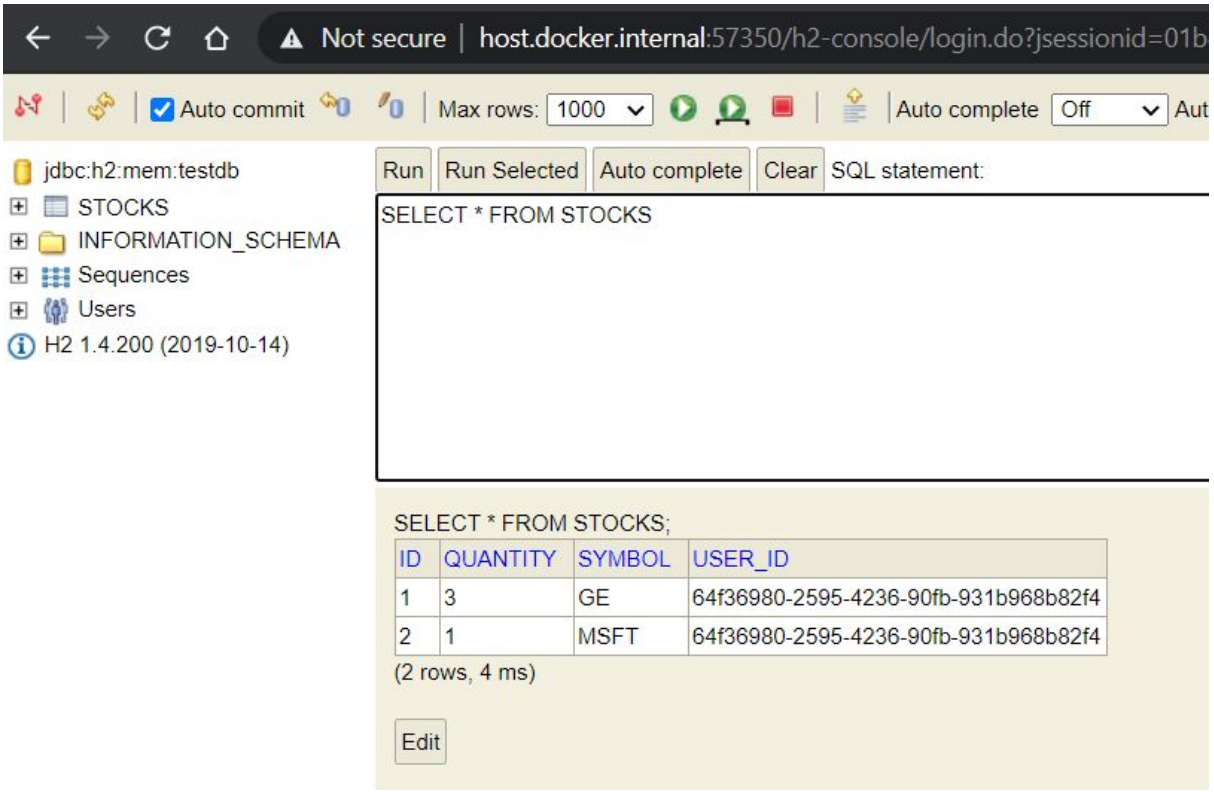


H2-console:

Eureka Dashboard > API Gateway > /users-ms/h2-console
http://host.docker.internal:8011/users-ms/h2-console



http://host.docker.internal:8011/stocks-ms/h2-console



Api gateway & load balancer

Port: 8011

Status Check

GET
http://localhost:8011/users-ms/users/status

Response Body:

Users MS is working on port: 57006

Sign Up

POST
http://localhost:8011/users-ms/users

Request Body:

```
{
    "firstName": "Erhan",
    "lastName": "Hepyasar",
    "email": "erhan.hepyasar@gmail.com",
    "password": "12345678",
    "cash": 800.00
}
```

Response Body:

```
{
    "firstName": "Erhan",
    "lastName": "Hepyasar",
    "email": "erhan.hepyasar@gmail.com",
    "cash": 800.0
}
```

Sign In API

POST
http://localhost:8011/users-ms/users/login

Response Headers:

userId:

64f36980-2595-4236-90fb-931b968b82f4

token (Json Web Token):

eyJhbGciOiJIUzUxMiJ9.eyJzdWIiOiI2NGYzNjk4MC0yNTk1LTQyMzYtOTBmYi05MzFiOTY4YjgyZjQiLCJleHAiOjE2MTI4ODE3MDh9.TQkDDHYLNZHknFB1zBAvDemTZ-WJjp2Fa1zF0emG4feGKWRWIskYCC4e6datlD2QJ4dJzjY33HYsK7R-p2OjHg

Get Users

GET
http://localhost:8011/users-ms/users

Response Body:


```
[
   {
      "firstName": "Murat",
      "lastName": "Caliskan",
      "email": "murat.caliskan@gmail.com",
      "cash": 800.0
   },
   {
      "firstName": "Erhan",
      "lastName": "Hepyasar",
      "email": "erhan.hepyasar@gmail.com",
      "cash": 800.0
   }
]
```


Get User By Email

GET
http://localhost:8011/users-ms/users/email/{email}

Response Body:


```
{
   "firstName": "Murat",
   "lastName": "Caliskan",
   "email": "murat.caliskan@gmail.com",
   "cash": 650.8
}
```


Get User Stocks By UserId

GET
http://localhost:8011/users-ms/users/64f36980-2595-4236-90fb-931b968b82f4/stocks

Response Body:


```
[
   {
```

```
        "symbol": "KO",
        "quantity": 2
    },
    {
        "symbol": "GE",
        "quantity": 3
    }
]
```


Get User Stock by UserId and Symbol

GET
http://localhost:8011/users-ms/users/{userId}/stocks/{symbol}

Response Body:


```
{
    "symbol": "GE",
    "quantity": 3
}
```


Deposit Cash

POST
http://localhost:8011/users-ms/users/{userId}/cash/{amount}

Response Body:

```
{
    "firstName": "Erhan",
    "lastName": "Hepyasar",
    "email": "erhan.hepyasar@gmail.com",
    "cash": 906.33
}
```


Stocks Status

GET
http://localhost:8011/stocks-ms/stocks/status

Response Body:
Stocks MS is working on port: 56994

Get Stocks

GET

http://localhost:8011/stocks-ms/stocks

Response Body:


```
[
   {
      "symbol": "GE",
      "name": "General Electric Company",
      "price": 11.61,
      "currency": "USD",
      "stockExchange": "NYSE"
   },
   {
      "symbol": "HD",
      "name": "The Home Depot, Inc.",
      "price": 280.03,
      "currency": "USD",
      "stockExchange": "NYSE"
   },
   {
      "symbol": "MSFT",
      "name": "Microsoft Corporation",
      "price": 242.47,
      "currency": "USD",
      "stockExchange": "NasdaqGS"
   },
   {
...
```


Get Stock By Symbol

GET
http://localhost:8011/stocks-ms/stocks/MSFT

Response Body:


```
{
   "symbol": "MSFT",
   "name": "Microsoft Corporation",
   "price": 242.47,
   "currency": "USD",
   "stockExchange": "NasdaqGS"
}
```


Buy Stock With User Id, Symbol and Price

POST
http://localhost:8011/users-ms/users/{userid}/stocks/buy

Request Body:

```
{
    "symbol": "MSFT",
    "price": 250,
    "quantity": 1
}
```

Response Body:

```
{
    "symbol": "MSFT",
    "price": 242.47,
    "quantity": 1
}
```

Sell Stock With User Id, Symbol and Price

POST
http://localhost:8011/users-ms/users/{userId}/stocks/sell

Request Body:

```
{
    "symbol": "MSFT",
    "price": 250,
    "quantity": 1
}
```

Response Body:

```
{
    "symbol": "MSFT",
    "price": 250.12,
    "quantity": 1
}
```

Spring Boot Actuator for Metrics

GET
http://localhost:8011/actuator
Response Body:


```
{
    "_links": {
        "self": {
            "href": "http://localhost:8011/actuator",
            "templated": false
        },
        "beans": {
            "href": "http://localhost:8011/actuator/beans",
            "templated": false
        },
        "health": {
            "href": "http://localhost:8011/actuator/health",
            "templated": false
        },
        "health-path": {
            "href": "http://localhost:8011/actuator/health/{*path}",
            "templated": true
        }
    }
}
```


GET
http://localhost:8011/actuator/health

Response Body:


```
{
    "status": "UP"
}
```


GET
http://localhost:8011/actuator/beans

GET
http://localhost:8011/users-ms/actuator/mappings

Running applications for different environments

Development: project-path> mvn spring-boot:run

Production: project-path> mvn spring-boot:run
-Dspring-boot.run.arguments=--spring.profiles.active=production