Set up cloud infrastructure – Components

- ✓ Config server
- ✓ Auth Server
- ✓ Api Gateway
- ✓ Discovery Service
- ✓ Cloud Monitoring- Spring Boot Admin
- ✓ Hystrix Dashboard

Services to implement

- ✓ Account Service
- ✓ Bank Information Service

API endpoints

- ✓ **/accounts/** Get list of all accounts **GET** /accounts/
- ✓ **/account/{id}** Create an account with specified ID **POST** /account/1
- ✓ **/account/{id}** Get specific account information **GET** /account/1
- ✓ **/bank/{id}** Get bank information for specified account id GET /bank/1

Tasks

- ✓ Implement service to manage accounts
- ✓ Implement service to get bank data of account
- ✓ for requested account id fetch account information from account service
- ✓ make sure that authentication checked automatically – Implementation in progress
- ✓ make sure that endpoint is resolved dynamically
- ✓ return account id as last digits of IBAN – Need more clarification
- ✓ take "DE89370400440532013087" as a base value
- ✓ replace last N digits of IBAN wit account id – Need more clarification
- ✓ Make sure that both services are properly registered in discovery and monitoring services

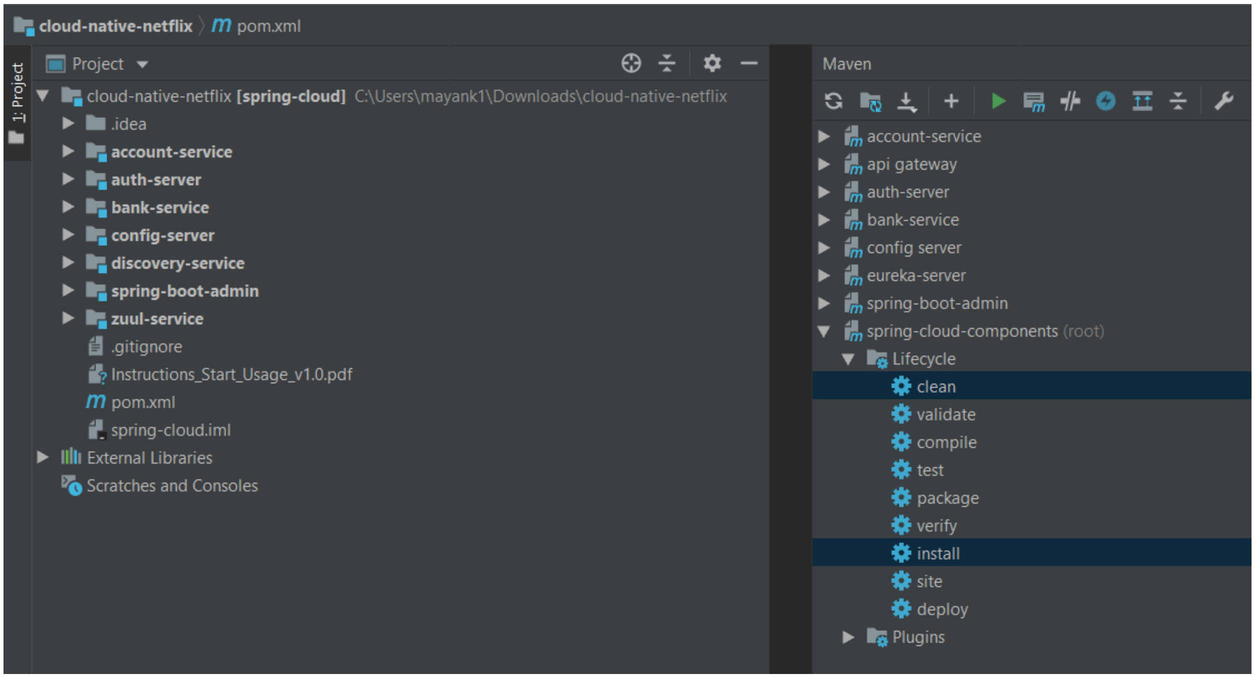
General Instructions

- ✓ Git Repository https://github.com/71mayank/cloud-native-netflix
- ✓ Clone Code git clone https://github.com/71mayank/cloud-native-netflix
- ✓ Import as Maven Project browse pom.xml from cloud-native-netflix folder
- ✓ Build Project (Parent Pom)– **mvn clean install** inside cloud-native-netflix

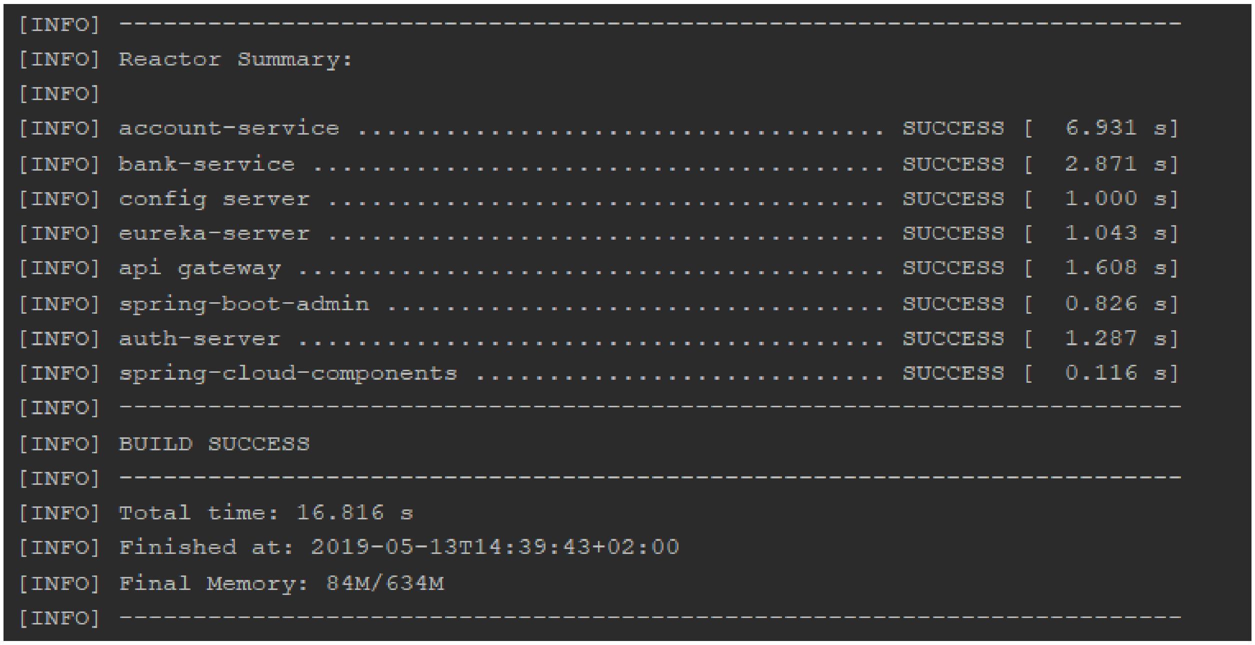
Project Dependencies

- Spring Boot, Spring cloud Eureka Server, Spring Cloud Config Server, Spring Cloud Zuul, Spring boot admin, Swagger, Lombok, Spring data jpa and H2 Database.

IntelliJ Import



Maven Clean Install on root – Spring cloud components



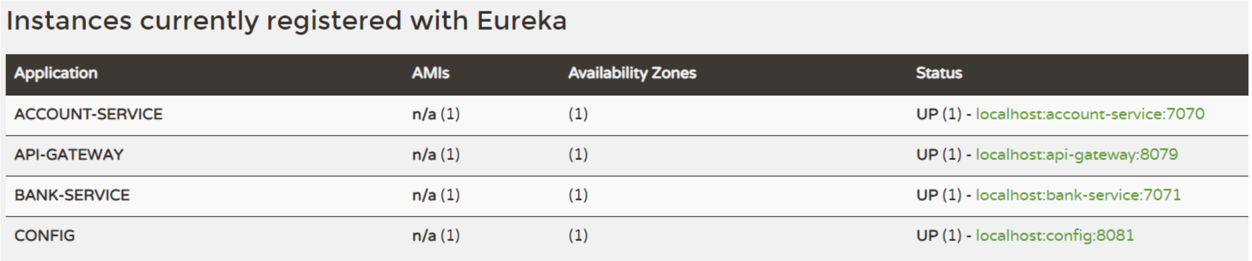
```
[INFO] ------------------------------------------------------------------------
[INFO] Reactor Summary:
[INFO]
[INFO] account-service .................................... SUCCESS [  6.931 s]
[INFO] bank-service ....................................... SUCCESS [  2.871 s]
[INFO] config server ...................................... SUCCESS [  1.000 s]
[INFO] eureka-server ...................................... SUCCESS [  1.043 s]
[INFO] api gateway ........................................ SUCCESS [  1.608 s]
[INFO] spring-boot-admin .................................. SUCCESS [  0.826 s]
[INFO] auth-server ........................................ SUCCESS [  1.287 s]
[INFO] spring-cloud-components ............................ SUCCESS [  0.116 s]
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time: 16.816 s
[INFO] Finished at: 2019-05-13T14:39:43+02:00
[INFO] Final Memory: 84M/634M
[INFO] ------------------------------------------------------------------------
```

Order to Start Spring Boot Applications

1. Eureka Server -**za.co.discovery.EurekaServerRestApplication**
2. Spring cloud config server -**za.co.config.ConfigServerRestApplication**
3. Zuul Proxy -**za.co.zuul.ZuulProxyRestApplication**
4. Account Service- **za.co.account.AccountRestApplication**
5. Bank Service- **za.co.bank.BanktRestApplication**
6. Spring boot admin Server - **za.co.admin.SpringBootAdminApplication**
7. Authorization Server - **za.co.auth.AuthServerRestApplication**

Observations

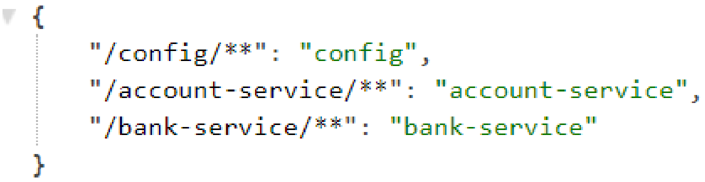
Eureka Dashboard (http://localhost:8020/)



Instances currently registered with Eureka

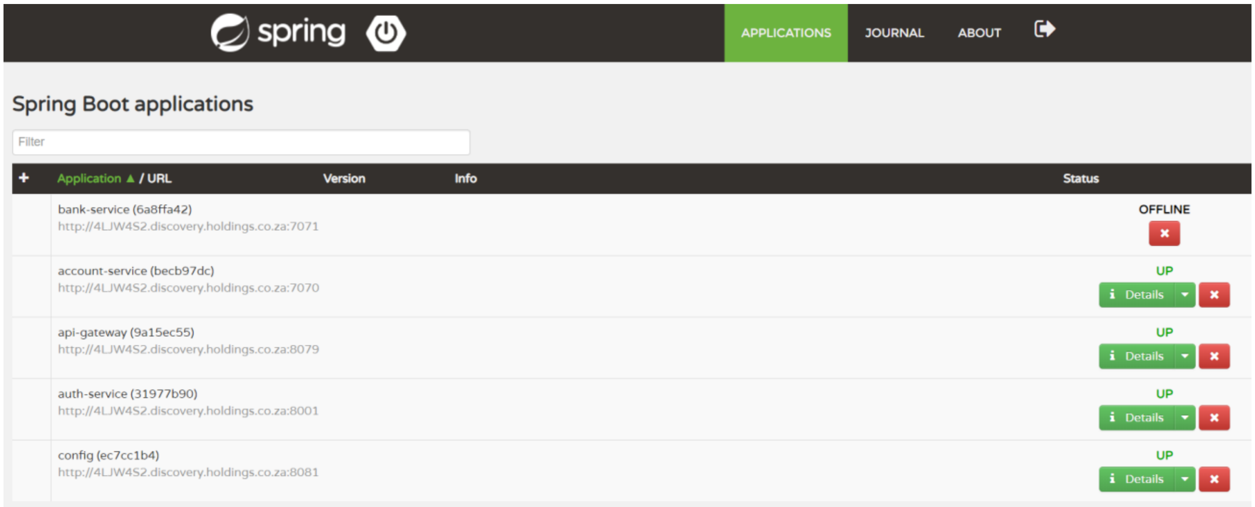
Application	AMIs	Availability Zones	Status
ACCOUNT-SERVICE	n/a (1)	(1)	UP (1) - localhost:account-service:7070
API-GATEWAY	n/a (1)	(1)	UP (1) - localhost:api-gateway:8079
BANK-SERVICE	n/a (1)	(1)	UP (1) - localhost:bank-service:7071
CONFIG	n/a (1)	(1)	UP (1) - localhost:config:8081

Zuul Routes (http://localhost:8079/routes)



```
{
    "/config/**": "config",
    "/account-service/**": "account-service",
    "/bank-service/**": "bank-service"
}
```

Spring boot admin – Monitoring (http://localhost:8093/) username – admin password – password

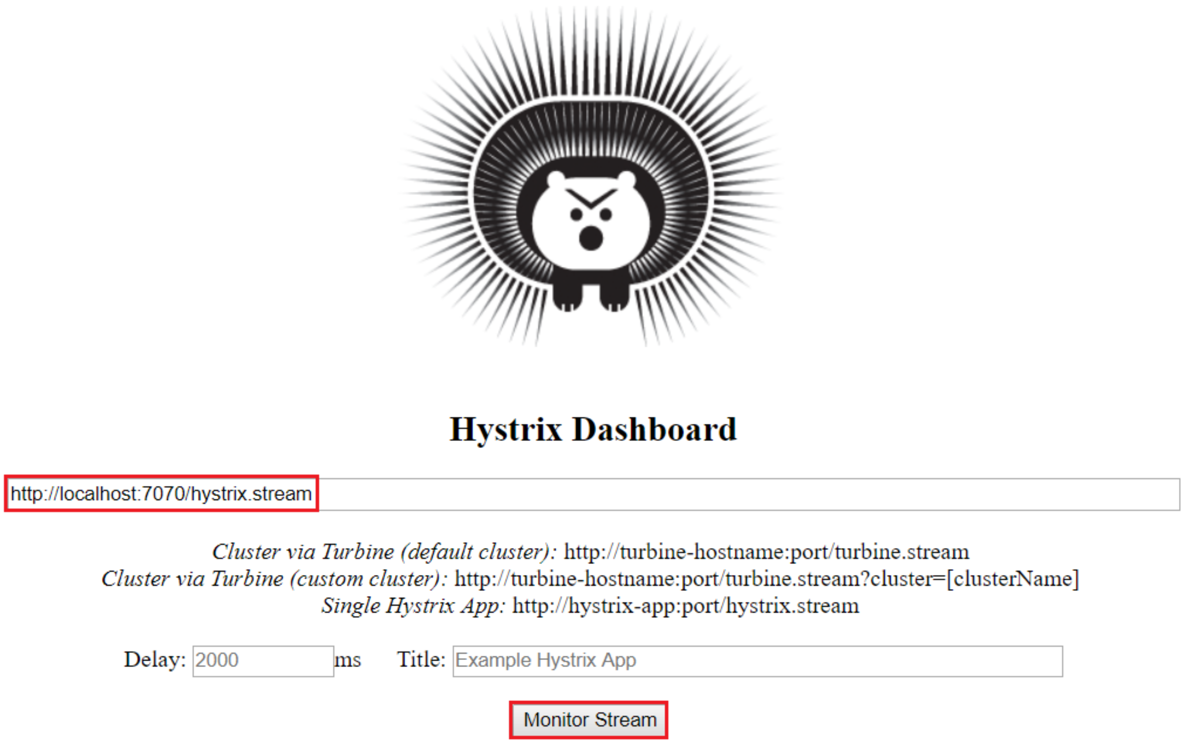


Spring Boot applications

Filter

+	Application ▲ / URL	Version	Info	Status
	bank-service (6a8ffa42) http://4LJW4S2.discovery.holdings.co.za:7071			OFFLINE ✖
	account-service (becb97dc) http://4LJW4S2.discovery.holdings.co.za:7070			UP Details ✖
	api-gateway (9a15ec55) http://4LJW4S2.discovery.holdings.co.za:8079			UP Details ✖
	auth-service (31977b90) http://4LJW4S2.discovery.holdings.co.za:8001			UP Details ✖
	config (ec7cc1b4) http://4LJW4S2.discovery.holdings.co.za:8081			UP Details ✖

Hystrix Dashboard – Account Service (http://localhost:7070/hystrix/)

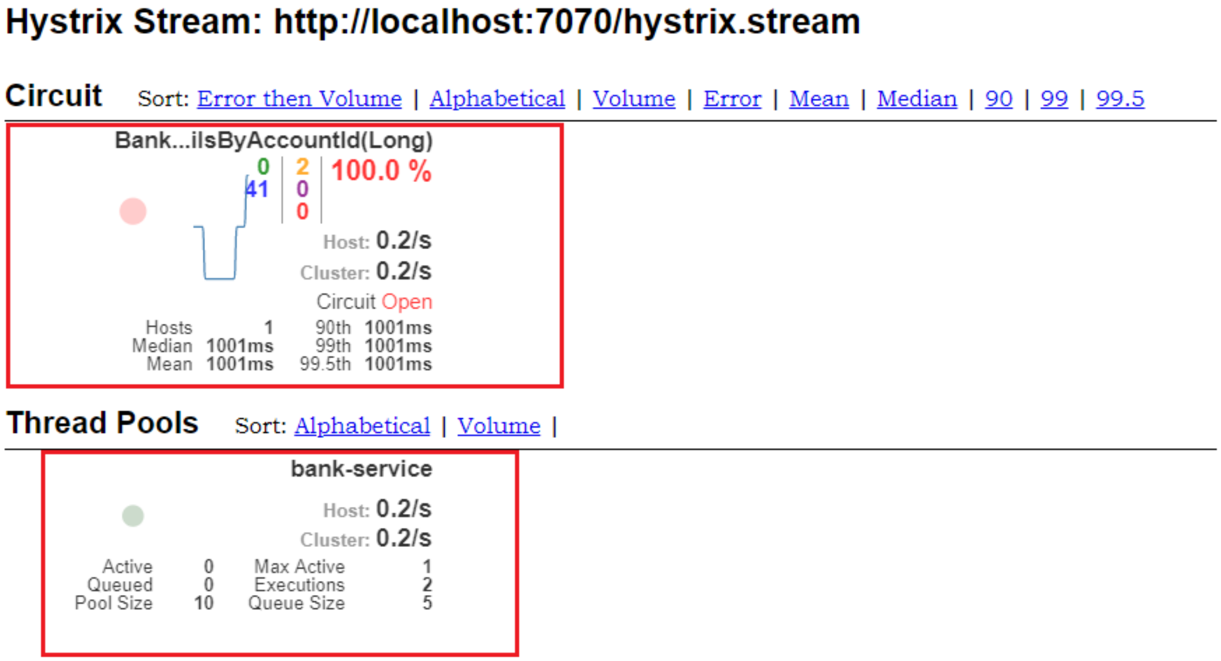


Hystrix Dashboard

http://localhost:7070/hystrix.stream

Cluster via Turbine (default cluster): http://turbine-hostname:port/turbine.stream
Cluster via Turbine (custom cluster): http://turbine-hostname:port/turbine.stream?cluster=[clusterName]
Single Hystrix App: http://hystrix-app:port/hystrix.stream

Delay: 2000 ms Title: Example Hystrix App



Hystrix Stream: http://localhost:7070/hystrix.stream

Circuit

Sort: Error then Volume | Alphabetical | Volume | Error | Mean | Median | 90 | 99 | 99.5

```
Bank...ilsByAccountId(Long)
  0 | 2 | 100.0 %
 41 | 0 |
    | 0 |
        Host: 0.2/s
     Cluster: 0.2/s
        Circuit Open
Hosts          1    90th   1001ms
Median   1001ms     99th   1001ms
Mean     1001ms     99.5th 1001ms
```

Thread Pools

Sort: Alphabetical | Volume |

```
                     bank-service
                     Host: 0.2/s
                  Cluster: 0.2/s
Active      0   Max Active     1
Queued      0   Executions     2
Pool Size  10   Queue Size     5
```

Eureka Instance
http://localhost:8020/

Zuul Proxy
http://localhost:8079/routes

Account Service via Zuul
http://localhost:8079/account-service/swagger-ui.html

Bank Service via Zuul
http://localhost:8079/bank-service/swagger-ui.html

Account Service Direct Link
http://localhost:7070/swagger-ui.html

Bank Service Direct Link
http://localhost:7071/swagger-ui.html

Auth Server to generate token

```
curl -i -X POST -H 'Accept: application/json'  -H 'Content-Type: application/json' --data
'{"username":"admin","password":"password"}' http://localhost:8001/generate-token
```

```
X-Content-Type-Options:  nosniff
X-XSS-Protection:  1; mode=block
Cache-Control: no-cache, no-store, max-age=0, must-revalidate
Pragma: no-cache
Expires: 0
X-Frame-Options:  DENY
Authorization: Bearer
eyJhbGciOiJIUzUxMiJ9.eyJzdWIiOiJhZG1pbiIsImV4cCI6MTU1ODYxMDUyM30.cBwSgI-
wpivn0i0L4gHj6FOUsE68w-QUoWmto2J8hPYlHXT_F0Kv2m6CxWGqxqaWoPT1uRS9akbPVDQYDQ705g
Content-Length: 0
Date: Mon, 13 May 2019 11:22:03 GMT
```

Client Accessing End Points with Authorization Header After getting a Valid Token (To be integrated)

```
curl -H 'Accept: application/json'  -H
"Authorization:eyJhbGciOiJIUzUxMiJ9.eyJzdWIiOiJhZG1pbiIsImV4cCI6MTU1ODYxMDUyM30.cBwSgI-
wpivn0i0L4gHj6FOUsE68w-
QUoWmto2J8hPYlHXT_F0Kv2m6CxWGqxqaWoPT1uRS9akbPVDQYDQ705g"
http://localhost:8001/users
```

Steps to check authentication automatically –

1. Service will first generate token using Auth Server – **generate-token**
2. End points will be invoked with **Authorization header** to get response.