1. **Creating Microservices for account and loan**

In this hands on exercises, we will create two microservices for a bank. One

microservice for handing accounts and one for handling loans.

**AccountController.java:**

package com.cognizant.account.controller;

import org.springframework.web.bind.annotation.\*;

*@RestController*

*@RequestMapping*("/accounts")

public class AccountController

{

*@GetMapping*("/{number}")

public Account getAccount(*@PathVariable* String number)

{

return new Account(number, "savings", 234343);

}

static class Account

{

private String number;

private String type;

private double balance;

public Account(String number, String type, double balance)

{

this.number = number;

this.type = type;

this.balance = balance;

}

public String getNumber()

{

return number;

}

public String getType()

{

return type;

}

public double getBalance()

{

return balance;

}

}

}

**AccountApplication.java:**

package com.cognizant.account;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

*@SpringBootApplication*

public class AccountApplication

{

public static void main(String[] args)

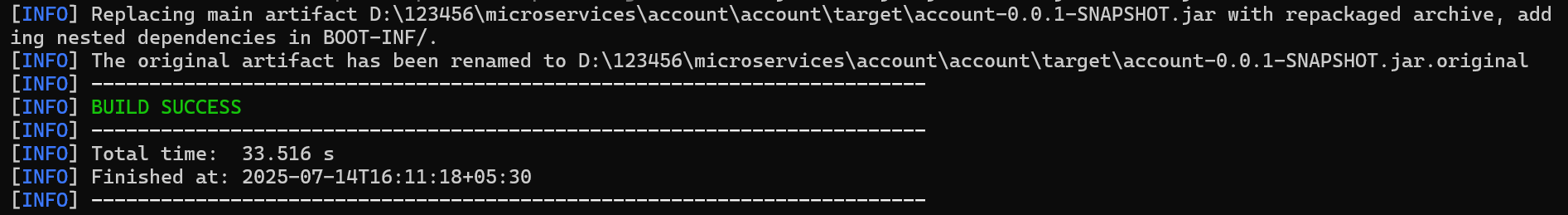
{

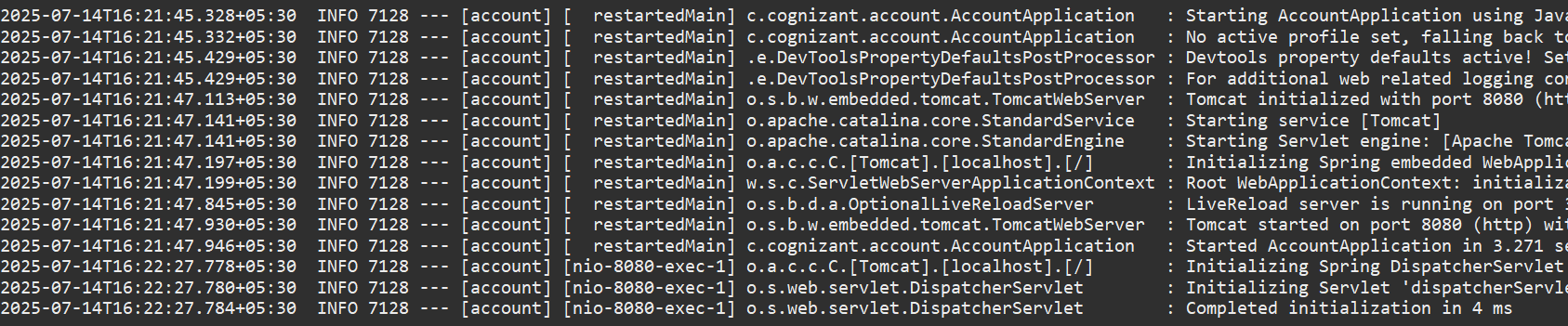
SpringApplication.*run*(AccountApplication.class, args);

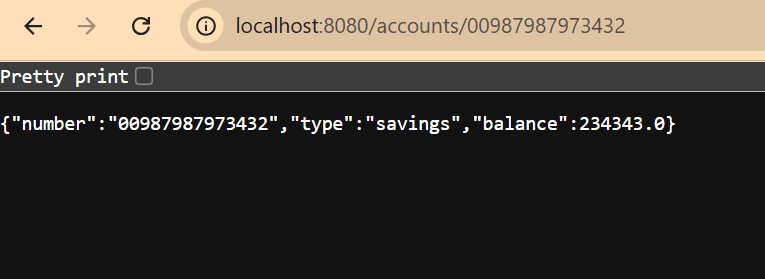
}

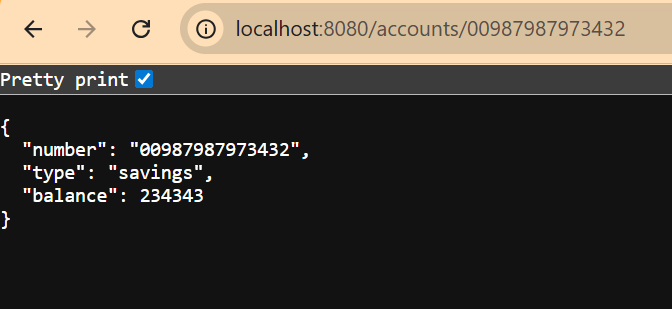
}

**Result:**









**Loan Microservice**

**application.properties:**

spring.application.name=loan

server.port=8085

**LoanController.java:**

package com.cognizant.loan.controller;

import org.springframework.web.bind.annotation.\*;

import java.util.Map;

*@RestController*

*@RequestMapping*("/loans")

public class LoanController

{

*@GetMapping*("/{number}")

public Map<String, Object> getLoanDetails(*@PathVariable* String number)

{

return Map.*of*(

"number", number,

"type", "car",

"loan", 400000,

"emi", 3258,

"tenure", 18

);

}

}

**LoanApplication.java:**

package com.cognizant.loan;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

*@SpringBootApplication*

public class LoanApplication

{

public static void main(String[] args)

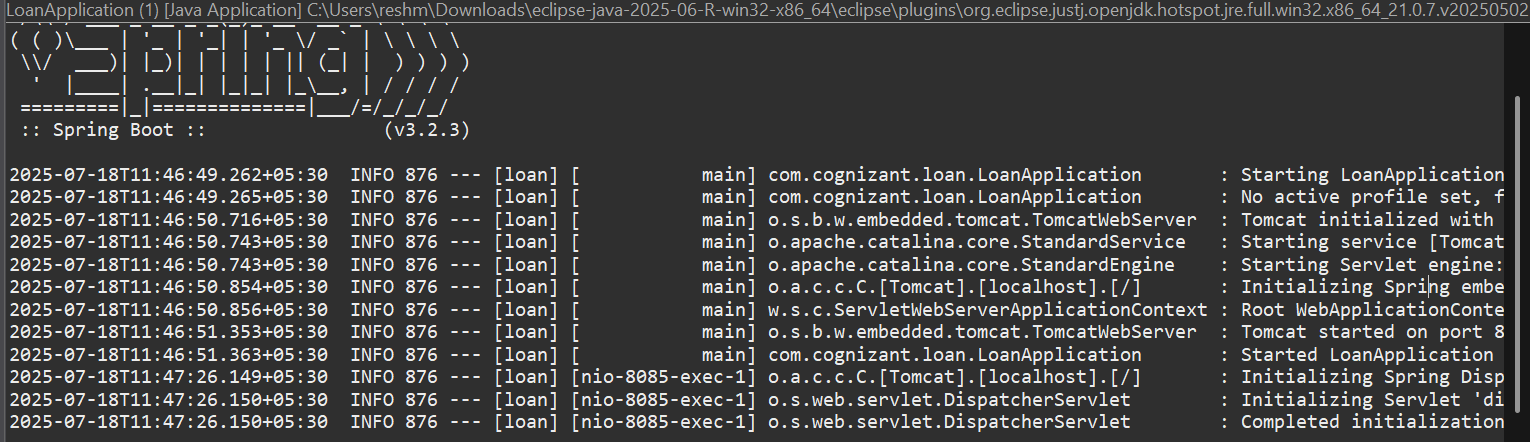
{

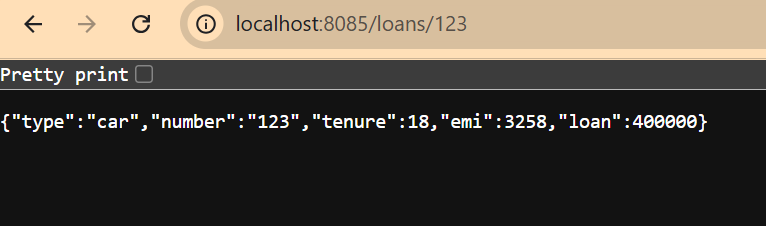
SpringApplication.*run*(LoanApplication.class, args);

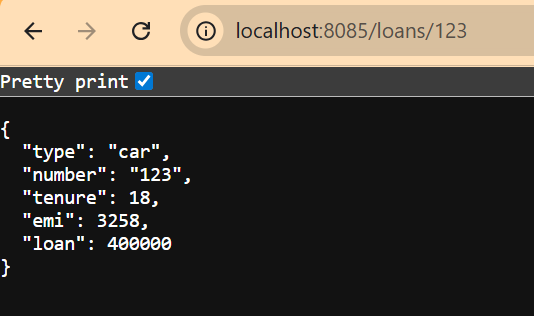
}

}

**Result:**







1. **Create Eureka Discovery Server and register microservices**

**EurekaDiscoveryServerApplication.java:**

package com.cognizant.eureka\_discovery\_server;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import org.springframework.cloud.netflix.eureka.server.EnableEurekaServer;

*@SpringBootApplication*

*@EnableEurekaServer*

public class EurekaDiscoveryServerApplication

{

public static void main(String[] args)

{

SpringApplication.*run*(EurekaDiscoveryServerApplication.class, args);

}

}

**pom.xml:**

<project xmlns="http://maven.apache.org/POM/4.0.0"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 https://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>com.cognizant</groupId>

<artifactId>eureka-discovery-server</artifactId>

<version>0.0.1-SNAPSHOT</version>

<name>Eureka Discovery Server</name>

<description>Spring Boot Eureka Discovery Server</description>

<packaging>jar</packaging>

<parent>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-parent</artifactId>

<version>3.1.5</version>

<relativePath/>

</parent>

<properties>

<java.version>21</java.version>

</properties>

<dependencyManagement>

<dependencies>

<dependency>

<groupId>org.springframework.cloud</groupId>

<artifactId>spring-cloud-dependencies</artifactId>

<version>2022.0.4</version>

<type>pom</type>

<scope>import</scope>

</dependency>

</dependencies>

</dependencyManagement>

<dependencies>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-web</artifactId>

</dependency>

<dependency>

<groupId>org.springframework.cloud</groupId>

<artifactId>spring-cloud-starter-netflix-eureka-server</artifactId>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-actuator</artifactId>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-test</artifactId>

<scope>test</scope>

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-maven-plugin</artifactId>

<version>3.1.5</version>

</plugin>

</plugins>

</build>

</project>

**application.properties:**

server.port=8761

spring.application.name=eureka-discovery-server

eureka.client.register-with-eureka=false

eureka.client.fetch-registry=false

**Result:**

