**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.

Each microservice will be a specific independent Spring RESTful Webservice maven project having it's own pom.xml. The only difference is that, instead of having both account and loan as a single application, it is split into two different applications. These webservices will be a simple service without any backend connectivity.

**Step-by-Step: Create Account and Loan Microservices**

**For Account Microservice:**

**1. Setup Folder Structure**

1. Go to D: drive.
2. Create a folder with your employee ID (e.g., D:\1234567)
3. Inside that, create a folder named microservices  
   → Final Path: D:\1234567\microservices

**2. Generate Account Microservice**

1. Open <https://start.spring.io>
2. Fill in:
   * Group: com.cognizant
   * Artifact: account
3. Add Dependencies:
   * Developer Tools → Spring Boot DevTools
   * Web → Spring Web
4. Click Generate → A zip file will be downloaded**.**

**3. Setup Account Microservice**

1. Extract the ZIP file.
2. Move the extracted account folder into D:\1234567\microservices
3. Open Command Prompt in D:\1234567\microservices\account

**4. Import into Eclipse**

1. Open **Eclipse IDE**
2. Go to: File → Import → Maven → Existing Maven Projects
3. Browse and select: D:\1234567\microservices\account
4. Click Finish.

**5. Implement Account Controller**

**Create a new Java class:**  
src/main/java/com/cognizant/account/controller/AccountController.java

package com.cognizant.account.controller;

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

import java.util.Map;

@RestController

@RequestMapping("/accounts")

public class AccountController {

@GetMapping("/{number}")

public Map<String, Object> getAccountDetails(@PathVariable String number) {

return Map.of(

"number", number,

"type", "savings",

"balance", 234343

);

}

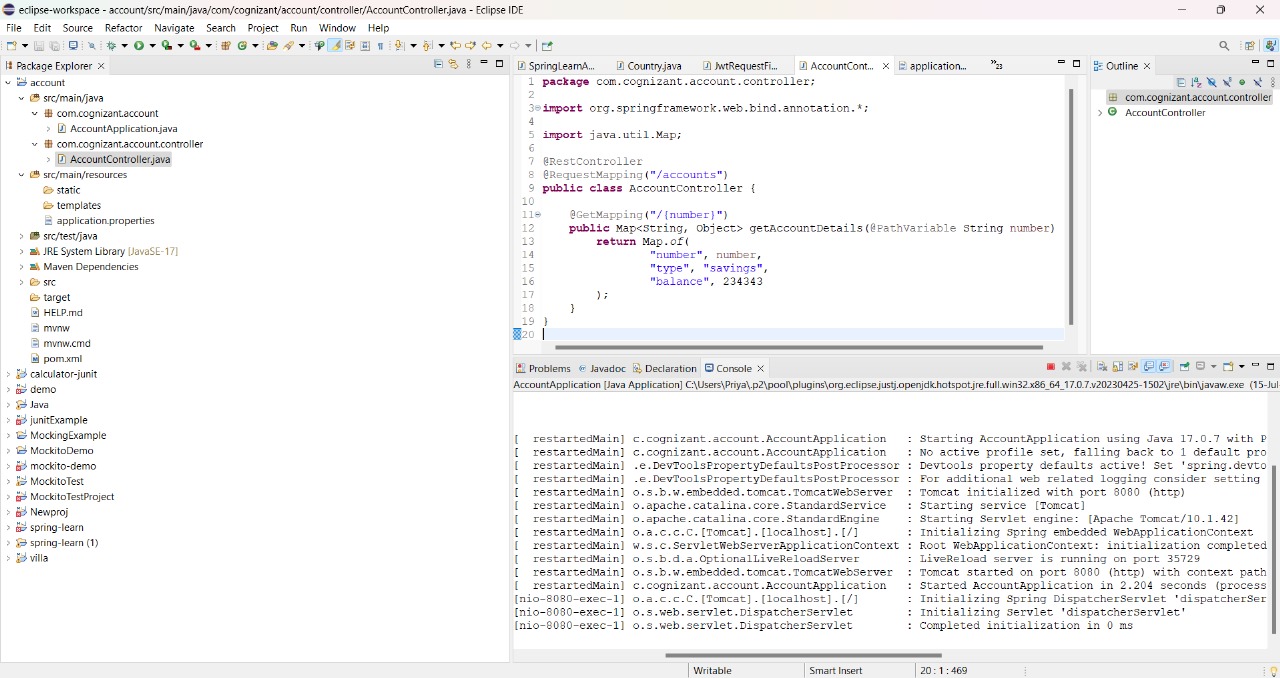
}

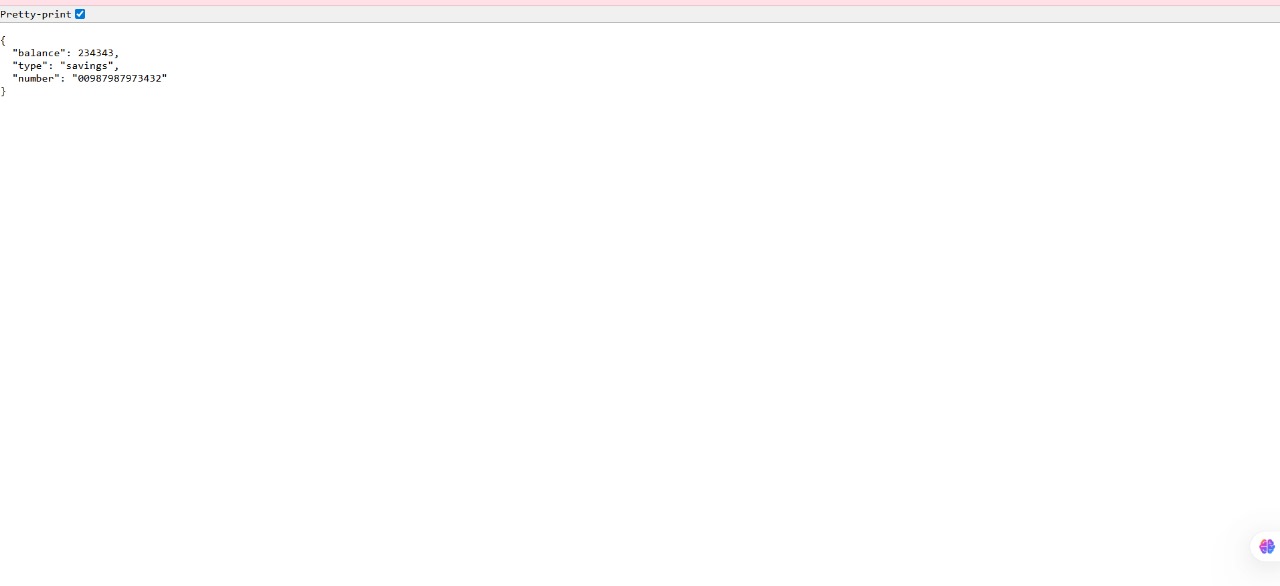
**6. Run Account Microservice**

1. Right-click on the main class: AccountApplication.java
2. Run as → Java Application
3. Open browser and go to:

<http://localhost:8080/accounts/00987987973432>

**OUTPUT:**





**For Loan Microservice:**

1**. Generate Loan Microservice**

1. Open <https://start.spring.io>
2. Fill in:
   * Group: com.cognizant
   * Artifact: loan
3. Add Dependencies:
   * Developer Tools → Spring Boot DevTools
   * Web → Spring Web
4. Click Generate and extract it to:

D:\1234567\microservices\loan

**8. Build and Import Loan Microservice**

1. Open Command Prompt in loan folder:
2. Import it into Eclipse (just like you did for account).

**9. Configure Port for Loan Microservice**

1. Go to: src/main/resources/application.properties
2. Add:

server.port=8081

**10. Create Loan Controller**

**Create a new Java class:**src/main/java/com/cognizant/loan/controller/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

);

}

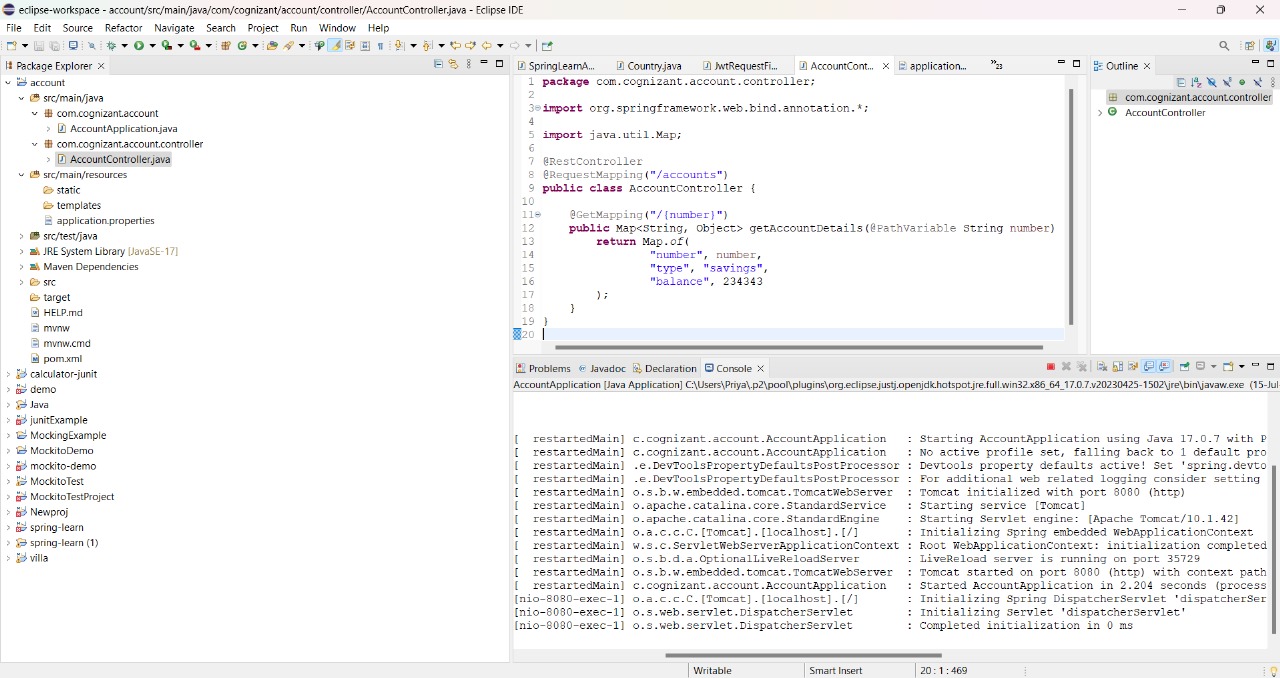
}

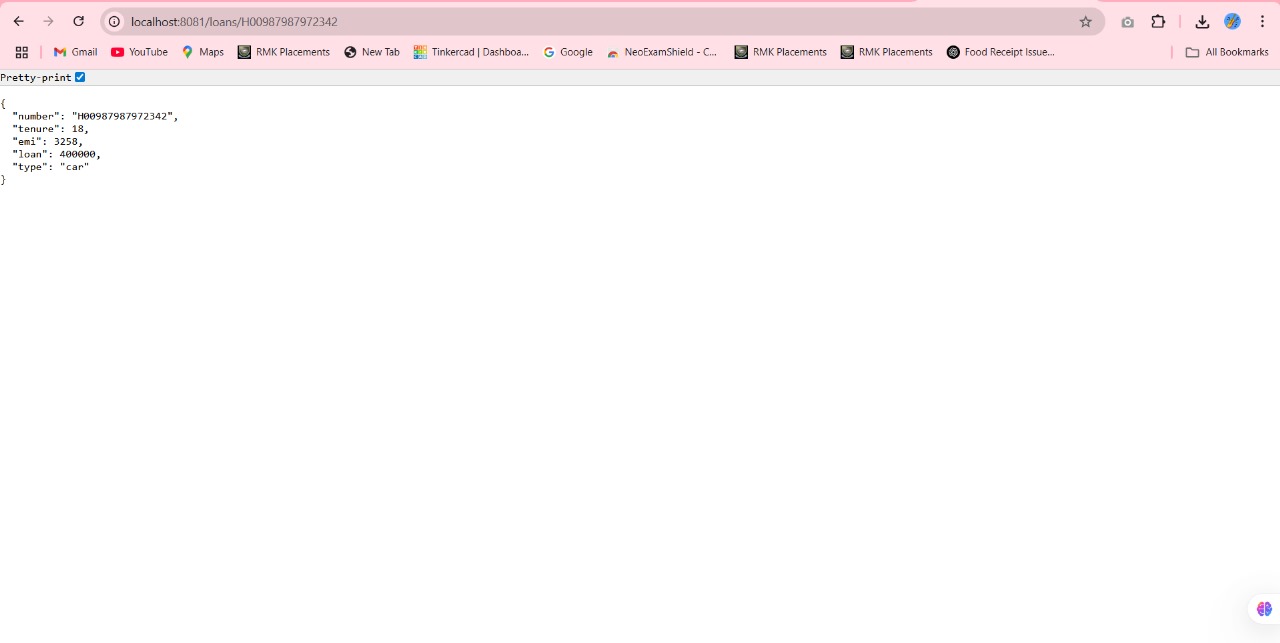
**11. Run Loan Microservice**

1. Right-click on the main class: LoanApplication.java
2. Run as → Java Application
3. Open browser and go to:

<http://localhost:8081/loans/H00987987972342>

**OUTPUT:**





**2.Create Eureka Discovery Server and register microservices**   
  
Eureka Discovery Server holds a registry of all the services that are available for immediate consumption. Anybody whom wants to consume a RESTful Web Service can come to the discovery server and find out what is available and ready for consumption. Eureka Discovery Server is part of spring cloud module.

**Step 1: Generate Eureka Discovery Server**

Go to: <https://start.spring.io>

* **Group:** com.cognizant
* **Artifact:** eureka-discovery-server
* **Module:** Select under “Spring Cloud Discovery”: **Eureka Server**

**Step 2: Import into Eclipse**

* Open Eclipse
* Go to File → Import → Existing Maven Projects
* Select the eureka-discovery-server folder
* Finish the import

**Step 3: Enable Eureka Server**

**EurekaDiscoveryServerApplication.java:**

@SpringBootApplication

@EnableEurekaServer

public class EurekaDiscoveryServerApplication {

public static void main(String[] args) {

SpringApplication.run(EurekaDiscoveryServerApplication.class, args);

}

}

**Step 4: Add Configuration to application.properties**

server.port=8761

eureka.client.register-with-eureka=false

eureka.client.fetch-registry=false

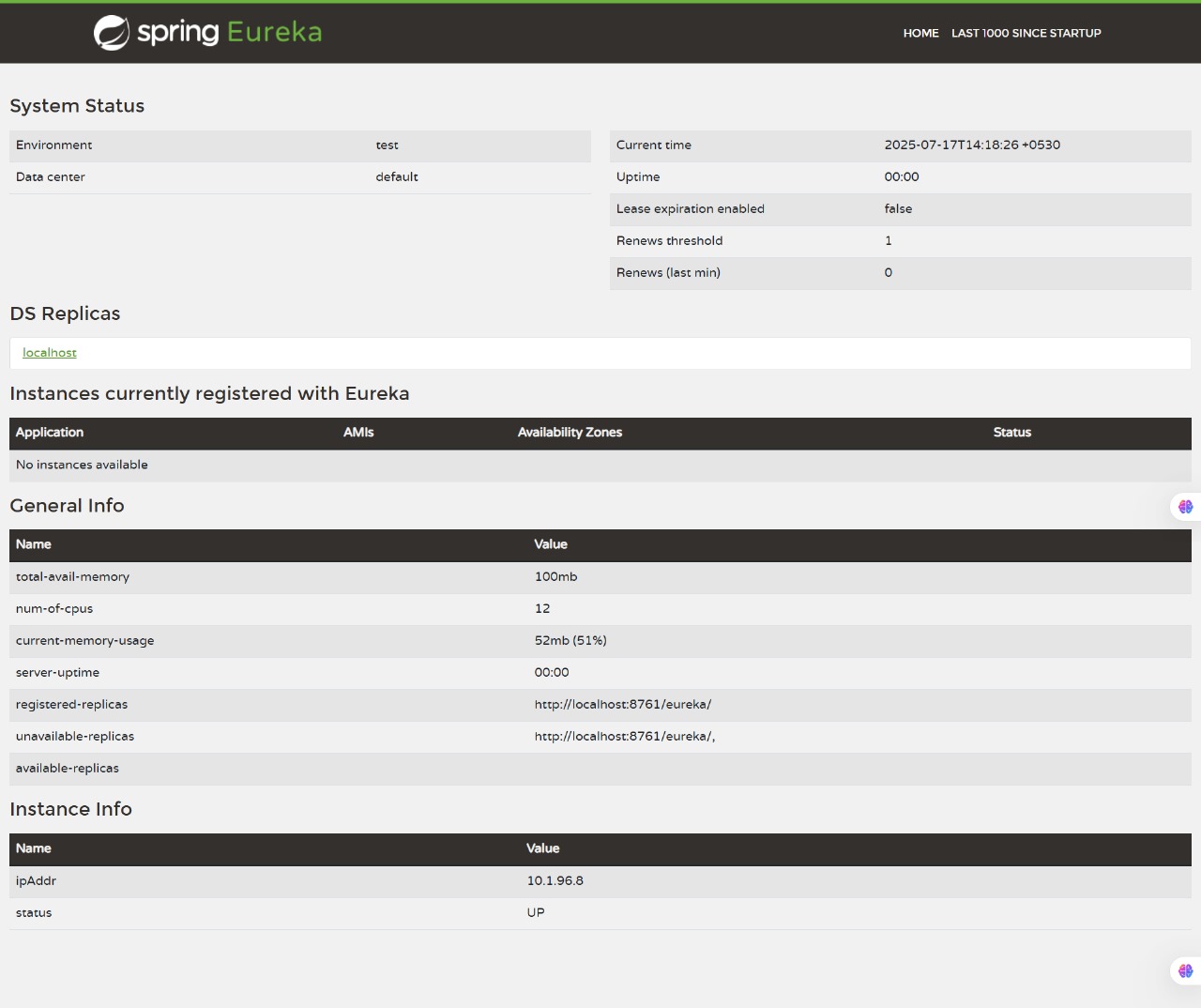
logging.level.com.netflix.eureka=OFF

logging.level.com.netflix.discovery=OFF

**Step 5: Run Eureka Server**

* Right-click the main class → Run As → Java Application
* Open browser → visit: http://localhost:8761

“**Instances currently registered with Eureka**” → Empty for now.



**Register Account Service with Eureka**

**Step 1: Generate Account Service**

Go to: <https://start.spring.io>

* **Group:** com.cognizant
* **Artifact:** account
* **Dependencies:**
  + Spring Boot DevTools
  + Eureka Discovery Client
  + Spring Web

Download and unzip.

**Step 2: Enable Discovery Client**

**AccountApplication.java (or main class):**

@SpringBootApplication

@EnableDiscoveryClient

public class AccountApplication {

public static void main(String[] args) {

SpringApplication.run(AccountApplication.class, args);

}

}

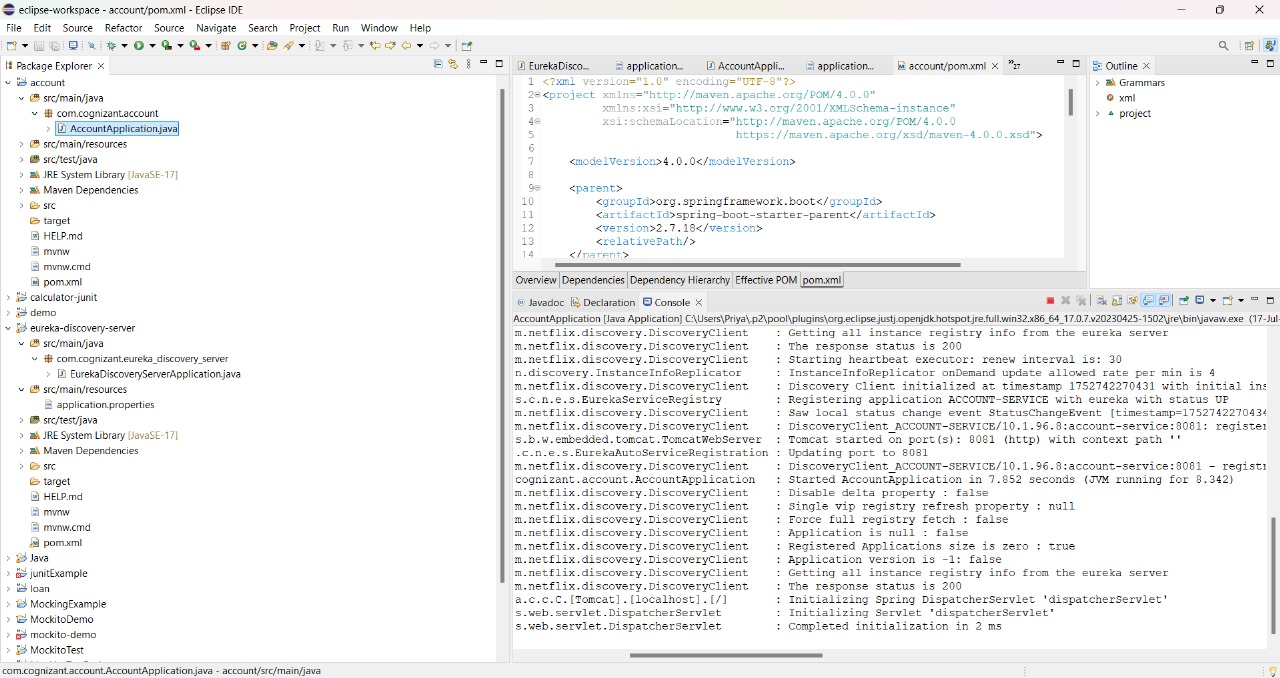
**Step 3: Add to application.properties**

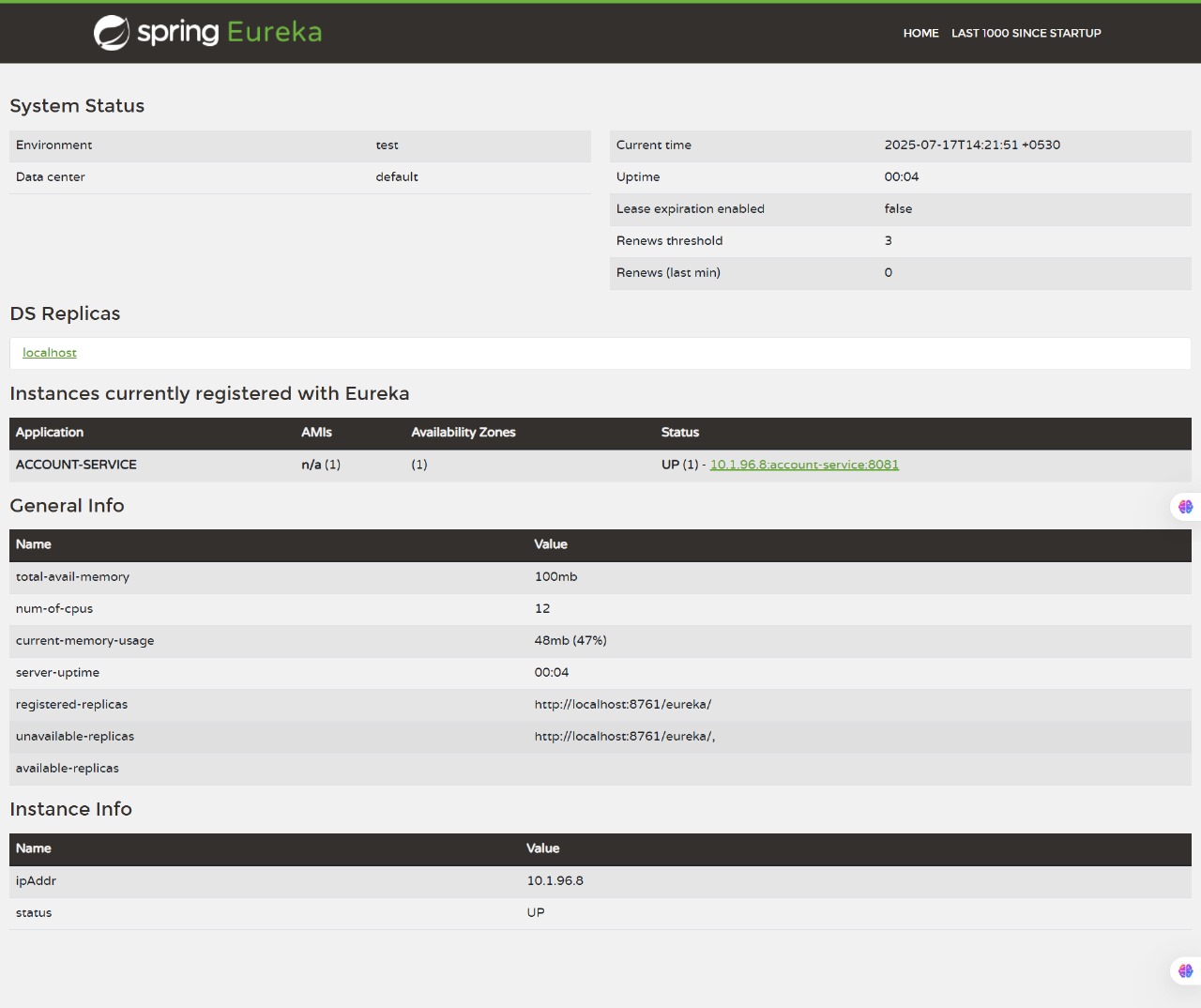
spring.application.name=account-service

eureka.client.service-url.defaultZone=http://localhost:8761/eureka

**Step 4: Run Services in Correct Order**

1. **Stop all services** using Eclipse Console (Terminate buttons).
2. **Start Eureka Discovery Server**.
   * Wait until it's fully started.
   * Open http://localhost:8761 → Confirm service list is **empty**.
3. **Start account service**.
   * Once started, refresh Eureka Dashboard.
   * You will see:





**Register Loan Service :**

1. Generate a new Spring Boot project from [start.spring.io](https://start.spring.io):
   * **Group:** com.cognizant
   * **Artifact:** loan
   * **Dependencies:**
     + Spring Boot DevTools
     + Eureka Discovery Client
     + Spring Web
2. Overwrite pom.xml using **Explore** → copy-paste
3. Build with Maven:
4. Import into Eclipse
5. Add @EnableDiscoveryClient to LoanApplication.java
6. application.properties:

spring.application.name=loan-service

eureka.client.service-url.defaultZone=http://localhost:8761/eureka

1. Run:
   * First eureka-discovery-server
   * Then loan application

Refresh Eureka UI. You will see both:

* account-service
* loan-service

**FINAL OUTPUT:**

