

# Configuration Manual

MSc Research Project  
Cloud Computing

Soumya Mohanan  
Student ID: x23104767

School of Computing  
National College of Ireland

Supervisor: Yasantha Samarawickrama

National College of Ireland  
Project Submission Sheet  
School of Computing



<b>Student Name:</b>	Soumya Mohanan
<b>Student ID:</b>	x23104767
<b>Programme:</b>	Cloud Computing
<b>Year:</b>	2023-2024
<b>Module:</b>	MSc Research Project
<b>Supervisor:</b>	Yasantha Samarawickrama
<b>Submission Due Date:</b>	12/08/2024
<b>Project Title:</b>	Configuration Manual
<b>Word Count:</b>	793
<b>Page Count:</b>	8

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

**ALL** internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

<b>Signature:</b>	Soumya Mohanan
<b>Date:</b>	12th August 2024

**PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST:**

Attach a completed copy of this sheet to each project (including multiple copies).	<input type="checkbox"/>
<b>Attach a Moodle submission receipt of the online project submission</b> , to each project (including multiple copies).	<input type="checkbox"/>
<b>You must ensure that you retain a HARD COPY of the project</b> , both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	<input type="checkbox"/>

Assignments that are submitted to the Programme Coordinator office must be placed into the assignment box located outside the office.

<b>Office Use Only</b>	
Signature:	
Date:	
Penalty Applied (if applicable):	

# Configuration Manual

Soumya Mohanan  
x23104767

## 1 Create AWS EC2 instances for the Microservices

1. Sign in to the AWS Management Console and open the Amazon EC2 console at <https://console.aws.amazon.com/ec2> and click on launch instance.

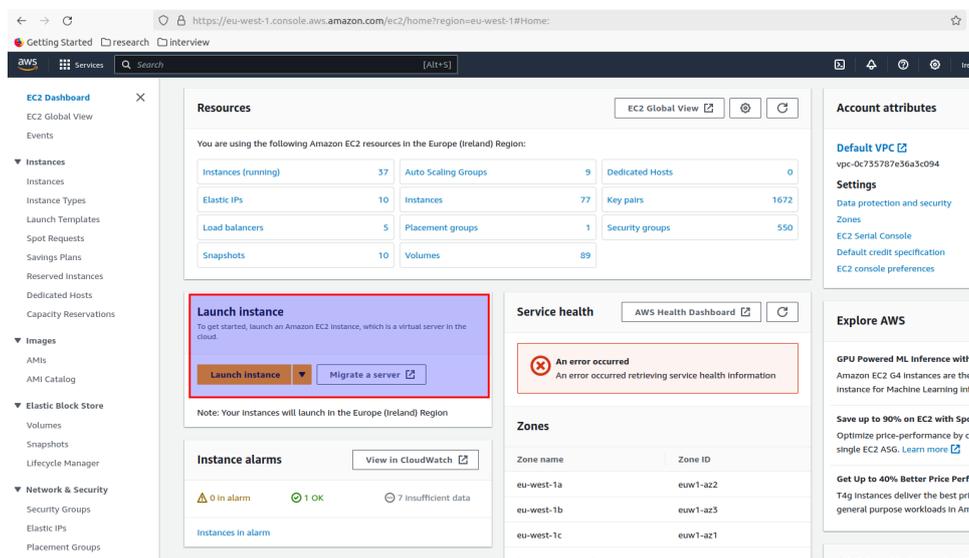


Figure 1: AWS EC2 Create Instance Page

2. Choose an Amazon Machine Image (AMI) and Instance Type: Amazon Linux 2 AMI and t2.micro instance type eligible for the AWS free tier.

- **Network Settings:**

- Select default VPC.
- Subnet preference set to "No preference."
- Auto-assign public IP enabled.
- New security group created with the following rules:
  - \* Allow SSH traffic from anywhere (0.0.0.0/0).
  - \* Allow HTTPS traffic from the internet (0.0.0.0/0).
  - \* Allow HTTP traffic from the internet (0.0.0.0/0).

3. Create a new key pair with the following options selected:

Key pair type : RSA

Private key format : .pem

Store the private key in secured place for establishing secure connection to the EC2 instances in later procedures.

**Create key pair** [X]

**Key pair name**  
Key pairs allow you to connect to your instance securely.  
  
The name can include up to 255 ASCII characters. It can't include leading or trailing spaces.

**Key pair type**

**RSA**  
RSA encrypted private and public key pair

**ED25519**  
ED25519 encrypted private and public key pair

**Private key file format**

**.pem**  
For use with OpenSSH

**.ppk**  
For use with PuTTY

**⚠** When prompted, store the private key in a secure and accessible location on your computer. **You will need it later to connect to your instance.** [Learn more](#)

Cancel **Create key pair**

Figure 2: Create New key pair

4. After these settings are performed click on "Launch Instance".

## 2 Change the network ACL inbound and outbound rules

Change the network ACL rules for ec2 instance to allow traffic from other microservice ports and external traffic by selecting the appropriate Network ACL. Add rules by editing "Inbound Rules" and "Outbound Rules" tabs as shown in the image for port 8080, 8083 and 8761.

Network ACL: [acl-070138242b2fd7d79](#) Edit network ACL association

**Inbound rules (8)**

Rule number	Type	Protocol	Port range	Source	Allow/Deny
1	All traffic	All	All	0.0.0.0/0	Allow
2	Custom TCP	TCP (6)	0	0.0.0.0/0	Allow
3	HTTP* (8080)	TCP (6)	8080	0.0.0.0/0	Allow
4	HTTP* (8080)	TCP (6)	8080	0.0.0.0/0	Allow
5	Custom TCP	TCP (6)	8761	0.0.0.0/0	Allow
6	Custom TCP	TCP (6)	8083	0.0.0.0/0	Allow
100	All traffic	All	All	0.0.0.0/0	Allow
*	All traffic	All	All	0.0.0.0/0	Deny

**Outbound rules (5)**

Rule number	Type	Protocol	Port range	Destination	Allow/Deny
1	Custom TCP	TCP (6)	8761	0.0.0.0/0	Allow
2	Custom TCP	TCP (6)	8083	0.0.0.0/0	Allow
3	HTTP* (8080)	TCP (6)	8080	0.0.0.0/0	Allow
100	All traffic	All	All	0.0.0.0/0	Allow
*	All traffic	All	All	0.0.0.0/0	Deny

Figure 3: Inbound and outbound rules required

## 3 Create JAR File and Transfer to the EC2 instance for the eureka server

Copy the application code from the code artifacts to the EC2 instance using the below steps:

1. Create a jar file for the application using the command:

```
mvn clean package
```

2. Change the permissions of the .pem file downloaded earlier when the new key pair was created:

```
chmod 400 /path/to/your-key.pem
```

3. Securely transfer the JAR file to the EC2 Instance using SCP:

```
scp -i /path/to/your-key.pem /path/to/local/file.jar ec2-user@EC2-Public-IP>:/remote/directory
```

## 4 Start the Eureka Server First

1. Connect to the EC2 Instance created earlier using SSH:

```
ssh -i /path/to/your-key.pem ec2-user@<EC2-Public-IP>
```

2. Install OpenJDK

```
sudo amazon-linux-extras install java-openjdk11 -y
```

3. Install Maven

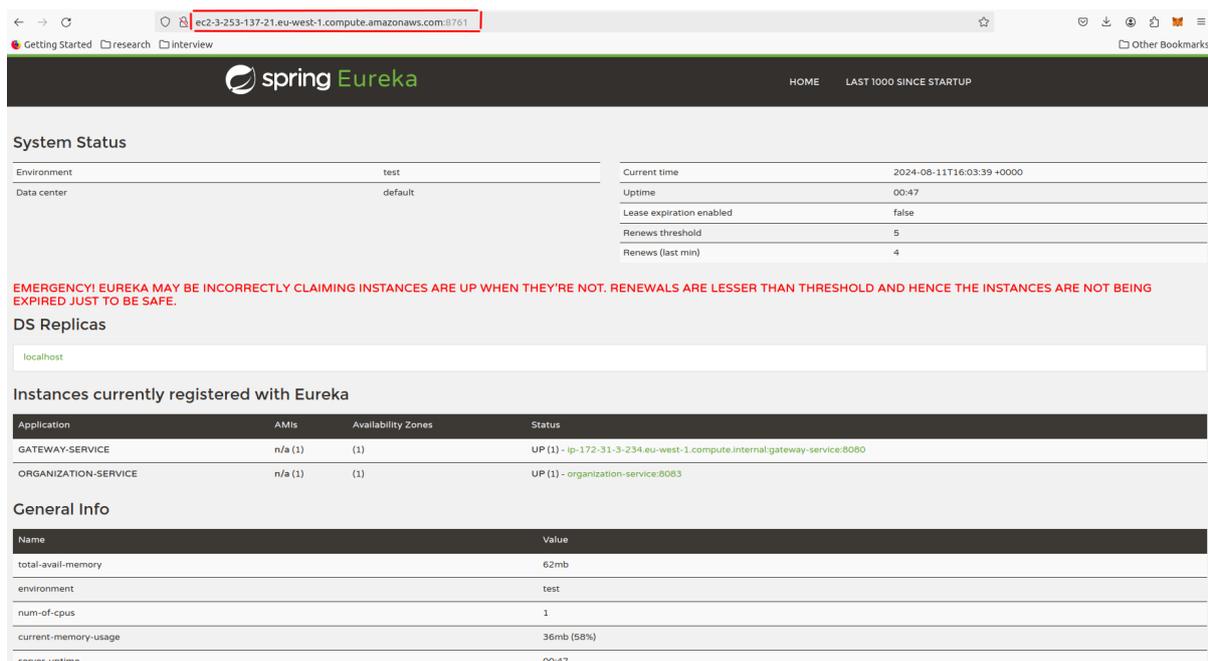
```
sudo yum install maven -y
```

4. Navigate to the folder where the jar file was copied.

5. Start the Spring Boot Eureka server on the EC2 Instance using the below command:

```
java -jar /remote/directory/file.jar
```

6. The eureka server will start on port 8761



The screenshot shows the Spring Eureka Dashboard in a browser window. The URL is `ec2-3-253-137-21.eu-west-1.compute.amazonaws.com:8761`. The dashboard includes a navigation bar with 'HOME' and 'LAST 1000 SINCE STARTUP'. The main content is divided into several sections:

- System Status:** A table showing environment details.

Environment	test	Current time	2024-08-11T16:03:39 +0000
Data center	default	Uptime	00:47
		Lease expiration enabled	false
		Renews threshold	5
		Renews (last min)	4
- EMERGENCY!** A red warning message: "EMERGENCY! EUREKA MAY BE INCORRECTLY CLAIMING INSTANCES ARE UP WHEN THEY'RE NOT. RENEWALS ARE LESSER THAN THRESHOLD AND HENCE THE INSTANCES ARE NOT BEING EXPIRED JUST TO BE SAFE."
- DS Replicas:** A table showing one replica on localhost.
- Instances currently registered with Eureka:** A table listing registered services.

Application	AMIs	Availability Zones	Status
GATEWAY-SERVICE	n/a (1)	(1)	UP (1) - ip-172-31-3-234.eu-west-1.compute.internal:gateway-service:8080
ORGANIZATION-SERVICE	n/a (1)	(1)	UP (1) - organization-service:8083
- General Info:** A table showing system metrics.

Name	Value
total-avail-memory	62mb
environment	test
num-of-cpus	1
current-memory-usage	36mb (58%)
server-up-time	00:47

Figure 4: Eureka Dashboard

## 5 Run the Zuul Gateway and Organisation Service

Follow the same steps as in Step 1 to Step 4 for starting the Eureka service with the respective application code and start the service.

- The Zuul Gateway will start on port 8080, accessible at:

```
http://<ec2_address>:8080
```

- The Organisation service will start on port 8083, accessible at:

```
http://<ec2_address>:8083
```

Instances currently registered with Eureka			
Application	AMIs	Availability Zones	Status
GATEWAY-SERVICE	n/a (1)	(1)	UP (1) - ip-172-31-3-234.eu-west-1.compute.internal:gateway-service:8080
ORGANIZATION-SERVICE	n/a (1)	(1)	UP (1) - organization-service:8083

General Info	
Name	Value
total-avail-memory	62mb
environment	test
num-of-ec2s	1

Figure 5: Services Registered on Eureka Dashboard

## 6 Load Testing using the Jmeter tool

### Step 1: Install JMeter

1. Download JMeter from the official Apache JMeter website.
2. Extract the downloaded archive.
3. Launch JMeter by running the `jmeter` script in the `bin` directory.

### Step 2: Create a Test Plan

1. Start JMeter by running the `jmeter` script in the `bin` directory.
2. Add a Test Plan

### Step 3: Add Thread Group

1. Thread Group: Right-click on the Test Plan, then choose Add Threads (Users).
2. Configure Thread Group:
  - **Number of Threads (Users):** Set the number of virtual users you want to simulate. 5000 in this case.
  - **Ramp-Up Period:** The time JMeter should take to start all the users.
  - **Loop Count:** Set how many times to execute the test.

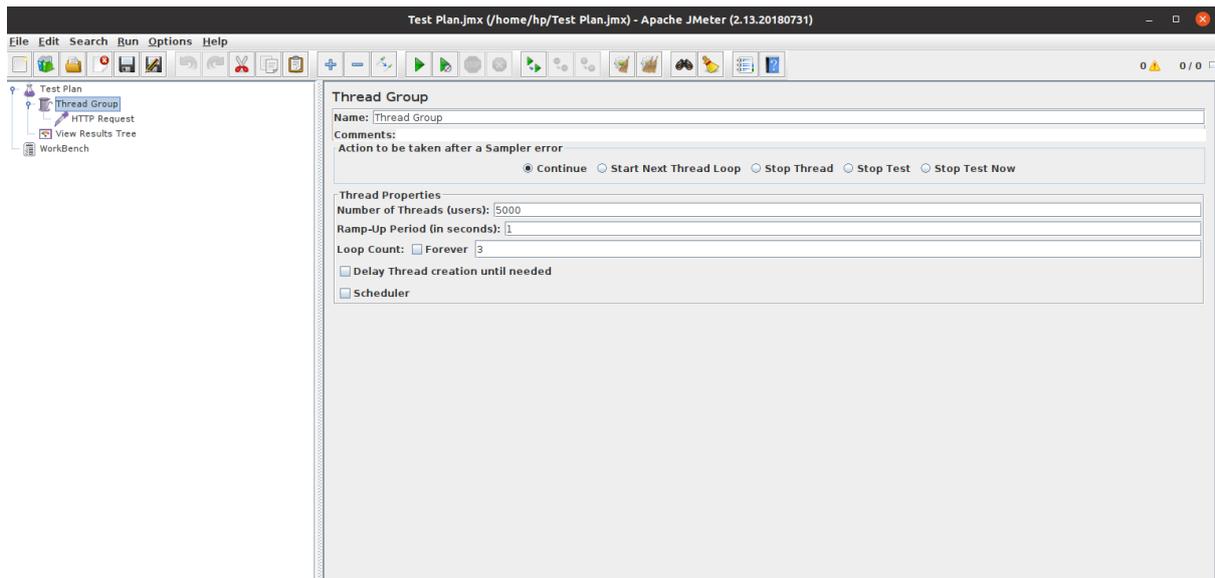


Figure 6: Thread Group settings

## Step 4: Add HTTP Request Sampler

1. HTTP Request: Right-click on the Thread Group, then choose Add > Sampler > HTTP Request.
2. Configure HTTP Request:
  - **Server Name or IP:** Enter the domain name or IP address of the server.
  - **Path:** Specify the API endpoint here it is /api/organisations/1.
  - **Method:** Choose the HTTP method - GET

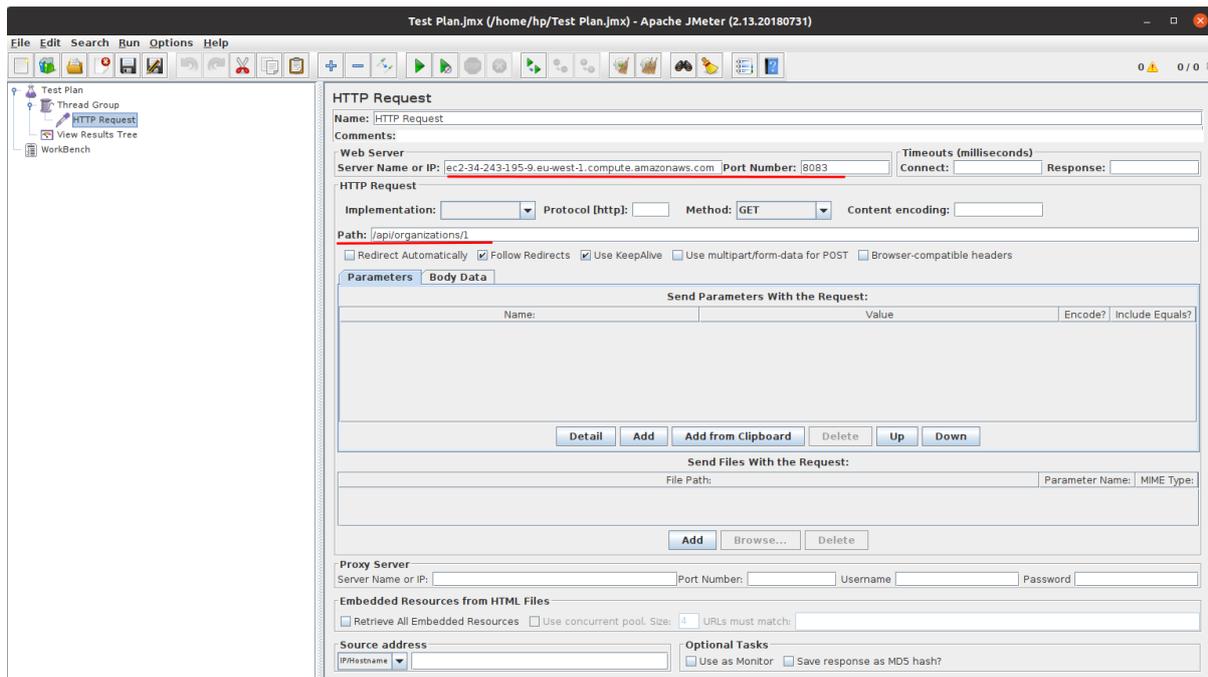


Figure 7: Request Sampler settings

## Step 5: Add Listeners

1. View Results: Right-click on the Thread Group, then choose **Add > Listener**.
2. Listeners:
  - **View Results Tree**: Provides detailed logs of each request.

## Step 6: Run the Test

1. Click on the green Play button in the JMeter toolbar to begin the load test.

# 7 Custom Logging of the dynamic heartbeat interval

Custom logging can be checked on the log details of the ec2 instance on which the service is running.

```
Activities Terminal Jul 25 11:56 AM ec2-user@ip-172-31-18-17:/organizations-service
2024-07-25 10:56:01.344 DEBUG 318063 --- [FreshExecutor-0] c.n.d.util.DeserializerStringCache : Clearing global-level cache with size 1
2024-07-25 10:56:01.422 DEBUG 318063 --- [FreshExecutor-0] c.n.d.util.DeserializerStringCache : Clearing app-level serialization cache with size 8
2024-07-25 10:56:01.422 DEBUG 318063 --- [FreshExecutor-0] c.s.s.t.j.AbstractJerseyEurekaHttpClient : Jersey HTTP GET http://ec2-3-253-137-21.eu-west-1.compute.amazonaws.com:8761/eureka/apps/delta?; statusCode=200
2024-07-25 10:56:01.422 DEBUG 318063 --- [FreshExecutor-0] com.netflix.discovery.DiscoveryClient : Got delta update with apps hashcode UP_1
2024-07-25 10:56:01.422 DEBUG 318063 --- [FreshExecutor-0] com.netflix.discovery.DiscoveryClient : Added instance organization-service:8083 to the existing apps in region null
2024-07-25 10:56:01.422 DEBUG 318063 --- [FreshExecutor-0] com.netflix.discovery.DiscoveryClient : The total number of instances fetched by the delta processor : 1
2024-07-25 10:56:01.423 DEBUG 318063 --- [FreshExecutor-0] com.netflix.discovery.DiscoveryClient : The total number of all instances in the client now is 1
2024-07-25 10:56:01.423 DEBUG 318063 --- [FreshExecutor-0] com.netflix.discovery.DiscoveryClient : Completed cache refresh task for discovery. All Apps hash code is Local region apps hashcode: UP_1,
is fetching remote regions? false
2024-07-25 10:56:01.819 INFO 318063 --- [TIner-0] c.s.o.config.DynamicHeartbeatService : Sending heartbeat...
2024-07-25 10:56:01.820 INFO 318063 --- [TIner-0] c.s.o.config.CustomEurekaClient : Heartbeat sent to Eureka at Thu Jul 25 10:56:01 UTC 2024 with interval 30000 ms
2024-07-25 10:56:01.821 INFO 318063 --- [TIner-0] c.s.o.config.DynamicHeartbeatService : Scheduling next heartbeat with interval: 8494 ms
2024-07-25 10:56:01.843 INFO 318063 --- [TIner-0] c.s.o.config.DynamicHeartbeatService : Sending heartbeat...
2024-07-25 10:56:01.844 INFO 318063 --- [TIner-0] c.s.o.config.CustomEurekaClient : Heartbeat sent to Eureka at Thu Jul 25 10:56:10 UTC 2024 with interval 30000 ms
2024-07-25 10:56:01.845 INFO 318063 --- [TIner-0] c.s.o.config.DynamicHeartbeatService : Scheduling next heartbeat with interval: 8241 ms
2024-07-25 10:56:01.749 INFO 318063 --- [TIner-0] c.s.o.config.DynamicHeartbeatService : Sending heartbeat...
2024-07-25 10:56:01.750 INFO 318063 --- [TIner-0] c.s.o.config.CustomEurekaClient : Heartbeat sent to Eureka at Thu Jul 25 10:56:11 UTC 2024 with interval 30000 ms
2024-07-25 10:56:01.902 INFO 318063 --- [TIner-0] c.s.o.config.DynamicHeartbeatService : Scheduling next heartbeat with interval: 8208 ms
2024-07-25 10:56:01.903 INFO 318063 --- [TIner-0] c.s.o.config.DynamicHeartbeatService : Sending heartbeat...
2024-07-25 10:56:01.903 INFO 318063 --- [TIner-0] c.s.o.config.CustomEurekaClient : Heartbeat sent to Eureka at Thu Jul 25 10:56:17 UTC 2024 with interval 30000 ms
2024-07-25 10:56:01.973 INFO 318063 --- [TIner-0] c.s.o.config.DynamicHeartbeatService : Scheduling next heartbeat with interval: 8145 ms
2024-07-25 10:56:01.973 INFO 318063 --- [TIner-0] c.s.o.config.DynamicHeartbeatService : Sending heartbeat...
2024-07-25 10:56:01.973 INFO 318063 --- [TIner-0] c.s.o.config.CustomEurekaClient : Heartbeat sent to Eureka at Thu Jul 25 10:56:18 UTC 2024 with interval 30000 ms
2024-07-25 10:56:01.881 INFO 318063 --- [TIner-0] c.s.o.config.DynamicHeartbeatService : Scheduling next heartbeat with interval: 8145 ms
2024-07-25 10:56:01.881 INFO 318063 --- [TIner-0] c.s.o.config.DynamicHeartbeatService : Sending heartbeat...
2024-07-25 10:56:01.881 INFO 318063 --- [TIner-0] c.s.o.config.CustomEurekaClient : Heartbeat sent to Eureka at Thu Jul 25 10:56:20 UTC 2024 with interval 30000 ms
2024-07-25 10:56:01.881 INFO 318063 --- [TIner-0] c.s.o.config.DynamicHeartbeatService : Scheduling next heartbeat with interval: 8143 ms
2024-07-25 10:56:01.881 INFO 318063 --- [TIner-0] c.s.o.config.DynamicHeartbeatService : Sending heartbeat...
2024-07-25 10:56:01.881 INFO 318063 --- [TIner-0] c.s.o.config.CustomEurekaClient : Heartbeat sent to Eureka at Thu Jul 25 10:56:26 UTC 2024 with interval 30000 ms
2024-07-25 10:56:01.881 INFO 318063 --- [TIner-0] c.s.o.config.DynamicHeartbeatService : Scheduling next heartbeat with interval: 8131 ms
2024-07-25 10:56:01.881 INFO 318063 --- [TIner-0] c.s.o.config.DynamicHeartbeatService : Sending heartbeat...
2024-07-25 10:56:01.881 INFO 318063 --- [TIner-0] c.s.o.config.CustomEurekaClient : Heartbeat sent to Eureka at Thu Jul 25 10:56:27 UTC 2024 with interval 30000 ms
2024-07-25 10:56:01.881 INFO 318063 --- [TIner-0] c.s.o.config.DynamicHeartbeatService : Scheduling next heartbeat with interval: 8131 ms
2024-07-25 10:56:01.881 INFO 318063 --- [TIner-0] c.s.o.config.DynamicHeartbeatService : Sending heartbeat...
2024-07-25 10:56:01.881 INFO 318063 --- [TIner-0] c.s.o.config.CustomEurekaClient : Heartbeat sent to Eureka at Thu Jul 25 10:56:28 UTC 2024 with interval 30000 ms
2024-07-25 10:56:01.881 INFO 318063 --- [TIner-0] c.s.o.config.DynamicHeartbeatService : Scheduling next heartbeat with interval: 8125 ms
2024-07-25 10:56:01.881 INFO 318063 --- [TIner-0] c.s.o.config.DynamicHeartbeatService : Sending heartbeat...
2024-07-25 10:56:01.881 INFO 318063 --- [TIner-0] c.s.o.config.CustomEurekaClient : Heartbeat sent to Eureka at Thu Jul 25 10:56:28 UTC 2024 with interval 30000 ms
2024-07-25 10:56:01.881 INFO 318063 --- [TConn-Cleaner2] c.n.d.shared.MonitoredConnectionManager : Closing connections idle longer than 30000 SECONDS
2024-07-25 10:56:01.881 INFO 318063 --- [TConn-Cleaner2] c.n.d.shared.MonitoredConnectionManager : Closing connections idle longer than 30000 SECONDS
2024-07-25 10:56:01.881 INFO 318063 --- [tbeatExecutor-0] c.n.d.shared.MonitoredConnectionManager : Getting free connection [{}->http://ec2-3-253-137-21.eu-west-1.compute.amazonaws.com:8761][null]
2024-07-25 10:56:01.881 INFO 318063 --- [tbeatExecutor-0] c.n.d.shared.MonitoredConnectionManager : Released connection is reusable
2024-07-25 10:56:01.881 INFO 318063 --- [tbeatExecutor-0] c.n.d.shared.MonitoredConnectionManager : Releasing connection [{}->http://ec2-3-253-137-21.eu-west-1.compute.amazonaws.com:8761][null]
2024-07-25 10:56:01.881 INFO 318063 --- [tbeatExecutor-0] c.n.d.shared.MonitoredConnectionManager : Pooling connection [{}->http://ec2-3-253-137-21.eu-west-1.compute.amazonaws.com:8761][null]; keep alive indefinitely
2024-07-25 10:56:01.881 INFO 318063 --- [tbeatExecutor-0] c.n.d.shared.MonitoredConnectionManager : Notifying no-one, there are no waiting threads
2024-07-25 10:56:01.881 INFO 318063 --- [tbeatExecutor-0] c.n.d.shared.MonitoredConnectionManager : Jersey HTTP PUT http://ec2-3-253-137-21.eu-west-1.compute.amazonaws.com:8761/eureka/apps/ORGANIZATION-SERVICE/organization-service:8083; statusCode=200
2024-07-25 10:56:01.881 INFO 318063 --- [tbeatExecutor-0] com.netflix.discovery.DiscoveryClient : DiscoveryClient_ORGANIZATION-SERVICE/organization-service:8083 - Heartbeat status: 200
2024-07-25 10:56:01.881 INFO 318063 --- [tbeatExecutor-0] c.n.d.shared.MonitoredConnectionManager : Get connection: [{}->http://ec2-3-253-137-21.eu-west-1.compute.amazonaws.com:8761, timeout = 5000
2024-07-25 10:56:01.881 INFO 318063 --- [tbeatExecutor-0] c.n.d.shared.MonitoredConnectionManager : [{}->http://ec2-3-253-137-21.eu-west-1.compute.amazonaws.com:8761] total kept alive: 2, total issued: 0, total allocated: 2 out of 200
2024-07-25 10:56:01.881 INFO 318063 --- [tbeatExecutor-0] c.n.d.shared.MonitoredConnectionManager : Getting free connection [{}->http://ec2-3-253-137-21.eu-west-1.compute.amazonaws.com:8761][null]
```

Figure 8: Dynamic heartbeat interval logs