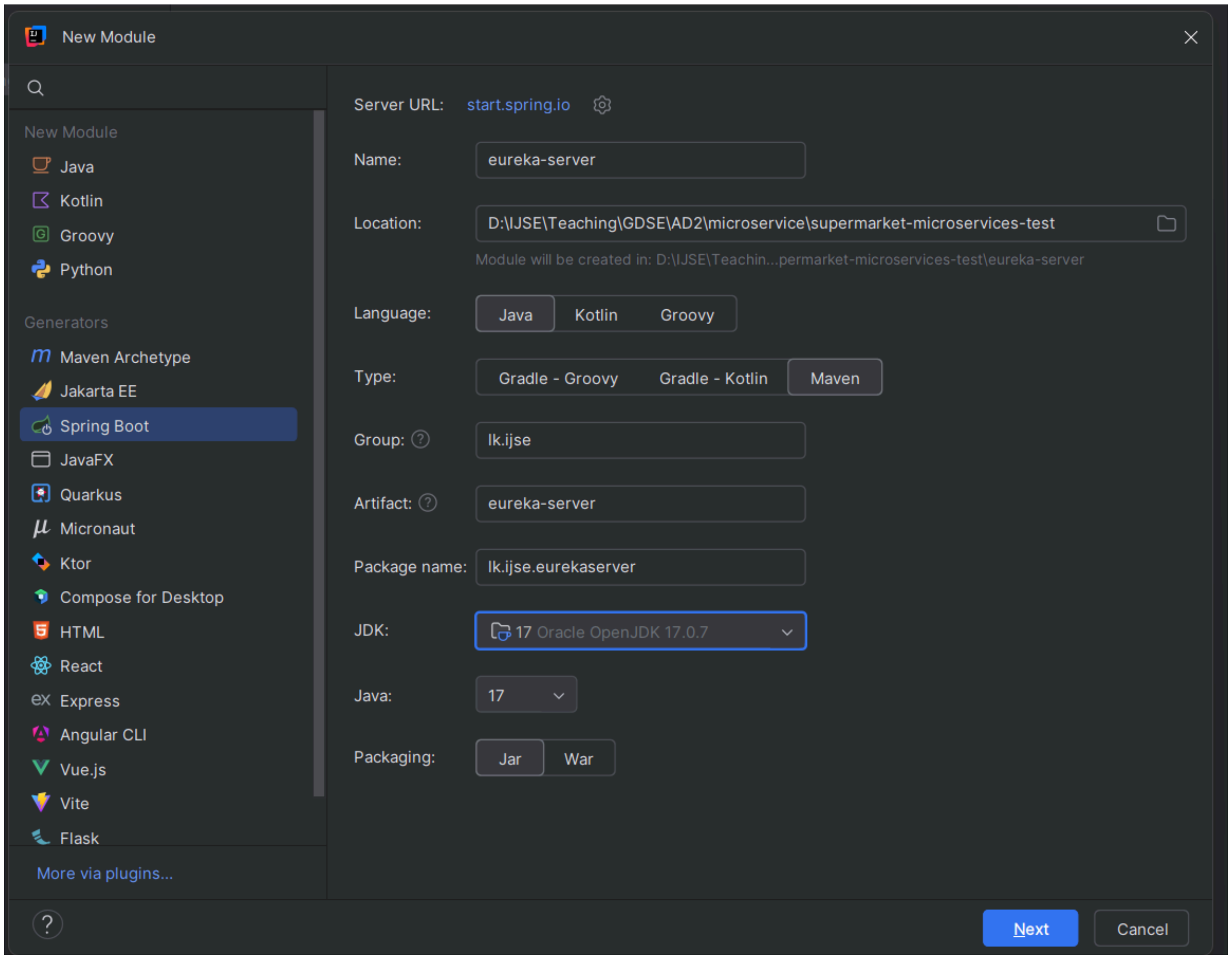
Microservices

supermarket-microservices

1. Eureka Discovery Server (Java Spring Boot) Maven :8761
2. Product Service (Java Spring Boot) Maven :8081
3. Order Service (Java Spring Boot) Gradle :8082
4. Inventory Service (Node Express.js) :3000
5. Customer Service (Python Flask) :5000
6. API Gateway (Spring Boot) Maven :8080

Eureka Discovery Server (Java Spring Boot) Maven

eureka-server



Server URL: start.spring.io

Name: eureka-server

Location: D:\IJSE\Teaching\GDSE\AD2\microservice\supermarket-microservices-test

Module will be created in: D:\IJSE\Teachin...permarket-microservices-test\eureka-server

Language: [✓ Java] Kotlin Groovy

Type: Gradle - Groovy Gradle - Kotlin [✓ Maven]

Group: lk.ijse

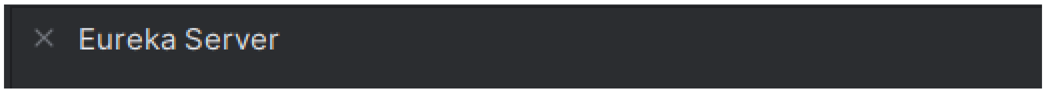
Artifact: eureka-server

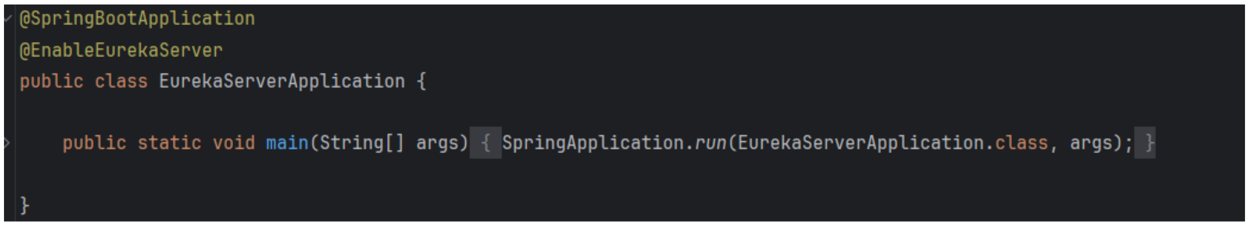
Package name: lk.ijse.eurekaserver

JDK: 17 Oracle OpenJDK 17.0.7

Java: 17

Packaging: [✓ Jar] War

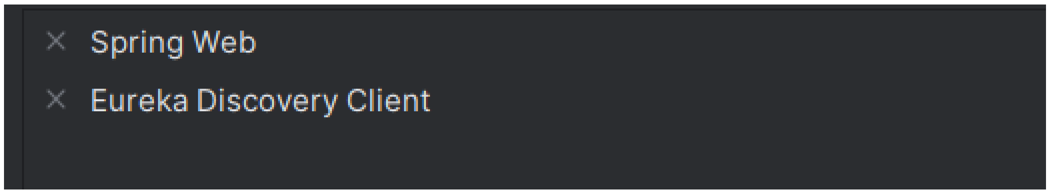

× Eureka Server


```java
@SpringBootApplication
@EnableEurekaServer
public class EurekaServerApplication {

    public static void main(String[] args) { SpringApplication.run(EurekaServerApplication.class, args); }

}
```

application.yml
```yaml
server:
 port: 8761

spring:
 application:
   name: eureka-server

eureka:
 client:
#    Server should not register itself as a service
#    central service registry,
#    Eureka server eka client kenenk vidiyata novima
   register-with-eureka: false
#    This prevents fetching its own service list
#    Eureka Server ekata thamanage client list eka ganna beri wenna
   fetch-registry: false
```

Spring Boot Service Applications (Java) Maven / Gradle

order-service
product-service


× Spring Web
× Eureka Discovery Client

Main class
```java
@EnableDiscoveryClient
```

application.yml
```yaml
server:
 port: 8082 # service running port (unique)
 servlet:
   context-path: /order-service # service context path (unique)


spring:
 application:
   name: order-service # service name (unique)
```

```
eureka:
 client:
   serviceUrl:
     defaultZone: http://localhost:8761/eureka/
```

Express.js Service Applications (Node) NPM

inventory-service

```
npm init -y
npm install express eureka-js-client
```

app.js

```
const express = require('express');
const {Eureka} = require('eureka-js-client');

const app = express();
const port = 3000;  // The port the service is running on

const router = express.Router()
// Inventory route
router.get('/inventory', (req, res) => {
   res.json({
       items: ['Milk', 'Eggs', 'Bread'],
       message: 'Welcome to the Inventory Service',
   });
});

app.use('/inventory-service', router)

// Eureka Client Configuration
const eurekaClient = new Eureka({
   instance: {
       instanceId: "inventory-service",
       app: "INVENTORY-SERVICE",
       hostName: "localhost",
       ipAddr: "127.0.0.1",
       port: {
           $: port,              "@enabled": true,
       },
       vipAddress: "inventory-service",
       dataCenterInfo: {
           "@class": "com.netflix.appinfo.InstanceInfo$DefaultDataCenterInfo",
           name: "MyOwn",
       },
   },
   eureka: {
       host: "localhost",
       port: 8761,
       servicePath: "/eureka/apps/",
      // or
      servicePath: "/eureka/",
       // or
       // without any servicePath: "/eureka/apps/",
   },
});

// Start the Express Server
app.listen(port, () => {
   console.log(`✅ Inventory service running at http://localhost:${port}`);

   // Register with Eureka
   eurekaClient.start((error) => {
       if (error) {
           console.error('❌ Failed to register with Eureka:', error);
       } else {
           console.log('✅ Successfully registered with Eureka.');
       }
   });
});
```

```
Node app.js
```

Flask Service Applications (Python) PIP

customer-service

```
pip install flask py_eureka_client
```

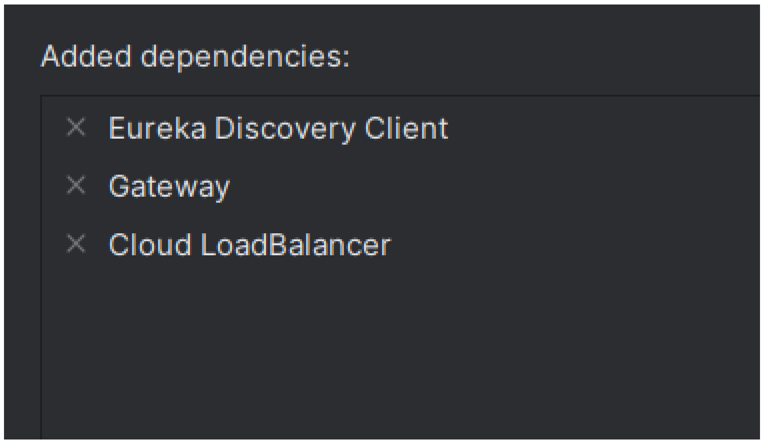
customer_service.py

```
from flask import Flask, jsonify
import py_eureka_client.eureka_client as eureka_client

app = Flask(__name__)

CONTEXT_PATH = "/customer-service"

# Eureka Configuration
EUREKA_SERVER = "http://localhost:8761/eureka/"
SERVICE_PORT = 5000  # Customer Service port

# Register the service with Eureka
eureka_client.init(
    eureka_server=EUREKA_SERVER,
    app_name="CUSTOMER-SERVICE",
    instance_port=SERVICE_PORT
)

# Sample Customer Data API
@app.route(f'{CONTEXT_PATH}/customers', methods=["GET"])
def get_customers():
    customers = [
        {"id": 1, "name": "John Doe", "email": "john@example.com"},
        {"id": 2, "name": "Jane Doe", "email": "jane@example.com"},
    ]
    return jsonify(customers)


if __name__ == "__main__":
    app.run(port=SERVICE_PORT)
```

```
python customer_service.py
```

API Gateway (Spring Boot) Maven

api-gateway



Added dependencies:

× Eureka Discovery Client

× Gateway

× Cloud LoadBalancer

Main class

```
@EnableDiscoveryClient
```

application.yml

```
spring:
 application:
   # The name of the application (API Gateway in this case).
   name: api-gateway

 cloud:
   loadbalancer:
     rule: random  # Choose random routing

   gateway:
     # Spring Cloud Gateway Configuration
     routes:
       # Define a route for the 'product-service'.
       - id: product-service
         # The URI where the product-service is registered in Eureka.
         # 'lb://' refers to load balancing using Spring Cloud Load Balancer to
discover the service.
         uri: lb://product-service
         predicates:
           # A predicate that matches requests to the path '/products/**'
```

```yaml
           # and forwards them to the product-service.
           - Path=/product-service/**

       # Define a route for the 'order-service'.
       - id: order-service
         # The URI where the order-service is registered in Eureka.
         uri: lb://order-service
         predicates:
           # This matches requests to the path '/orders/**'
           # and forwards them to the order-service.
           - Path=/order-service/**

       # Define a route for the 'inventory-service'.
       - id: inventory-service
         # The URI where the inventory-service is registered in Eureka.
         uri: lb://inventory-service
         predicates:
           # This matches requests to the path '/inventory/**'
           # and forwards them to the inventory-service.
           - Path=/inventory-service/**

       # Define a route for the 'customer-service'.
       - id: customer-service
         # The URI where the customer-service is registered in Eureka.
         uri: http://localhost:5000
#          uri: lb://customer-service
         predicates:
           # This matches requests to the path '/customer/**'
           # and forwards them to the customer-service.
           - Path=/customer-service/**

   # Eureka Client Configuration
   eureka:
     client:
       # The URL of the Eureka Server, where the services register themselves.
       serviceUrl:
         # Eureka Server URL where the API Gateway can register and discover services.
         defaultZone: http://localhost:8761/eureka/
```

Ex:-
http://localhost:8082/order-service/orders
http://localhost:8081/product-service/products
http://localhost:3000/inventory-service/inventory
http://localhost:5000/customer-service/customers

http://localhost:8080/order-service/orders
http://localhost:8080/product-service/products
http://localhost:8080/inventory-service/inventory
http://localhost:8080/customer-service/customers