**MICROSERVICE USING MAVEN**

Github - https://github.com/Prsingh9/Microservices.git

1) Installed the following on your system and set environment variable:

1. Java 21
2. Maven
3. Have Docker Desktop
4. Enable Kubernettes inside Docker Desktop
5. PostgreSQL
6. Kafka & Zookeeper

2) VS Code Extensions → Installed Java, Spring Boot, Docker, and Kubernetes tools  
  
3) Set up Spring Boot microservices

1. Create the user-service

mvn archetype:generate -DgroupId=com.ecommerce -DartifactId=user-service -DarchetypeArtifactId=maven-archetype-quickstart -DinteractiveMode=false

1. Create the Product Service

mvn archetype:generate -DgroupId=com.ecommerce -DartifactId=product-service -DarchetypeArtifactId=maven-archetype-quickstart -DinteractiveMode=false

1. Create the order-service

mvn archetype:generate -DgroupId=com.ecommerce -DartifactId=order-service -DarchetypeArtifactId=maven-archetype-quickstart -DinteractiveMode=false

1. Create the payment-service

mvn archetype:generate -DgroupId=com.ecommerce -DartifactId=payment-service -DarchetypeArtifactId=maven-archetype-quickstart -DinteractiveMode=false

**FILE STRUCTURE**

**Microservices/**

**├── docker-compose.yml**

**├── k8s**

**│ ├── order-service-deployment.yml**

**│ ├── payment-service-deployment.yml**

**│ ├── postgres-deployment.yml**

**│ ├── product-service-deployment.yml**

**│ └── user-service-deployment.yml**

**├── order-service**

**│ ├── Dockerfile**

**│ ├── pom.xml**

**│ ├── src**

**│ │ ├── main**

**│ │ │ └── java**

**│ │ │ └── com**

**│ │ │ └── ecommerce**

**│ │ │ └── orderservice**

**│ │ │ ├── OrderServiceApplication.java**

**│ │ │ ├── controller**

**│ │ │ │ └── OrderController.java**

**│ │ │ ├── entity**

**│ │ │ │ └── Order.java**

**│ │ │ ├── kafka**

**│ │ │ │ └── OrderProducer.java**

**│ │ │ ├── repository**

**│ │ │ │ └── OrderRepository.java**

**│ │ │ └── resources**

**│ │ │ └── application.properties**

**│ │ └── test**

**│ │ └── java**

**│ │ └── com**

**│ │ └── ecommerce**

**│ │ └── AppTest.java**

**├── payment-service**

**│ ├── Dockerfile**

**│ ├── pom.xml**

**│ ├── src**

**│ │ ├── main**

**│ │ │ ├── java**

**│ │ │ │ └── com**

**│ │ │ │ └── ecommerce**

**│ │ │ │ └── paymentservice**

**│ │ │ │ ├── PaymentServiceApplication.java**

**│ │ │ │ └── kafka**

**│ │ │ │ └── PaymentConsumer.java**

**│ │ │ └── resources**

**│ │ │ └── application.properties**

**│ │ └── test**

**│ │ └── java**

**│ │ └── com**

**│ │ └── ecommerce**

**│ │ └── AppTest.java**

**├── product-service**

**│ ├── Dockerfile**

**│ ├── pom.xml**

**│ ├── src**

**│ │ ├── main**

**│ │ │ ├── java**

**│ │ │ │ └── com**

**│ │ │ │ └── ecommerce**

**│ │ │ │ └── productservice**

**│ │ │ │ ├── ProductServiceApplication.java**

**│ │ │ │ ├── controller**

**│ │ │ │ │ └── ProductController.java**

**│ │ │ │ ├── entity**

**│ │ │ │ │ └── Product.java**

**│ │ │ │ └── repository**

**│ │ │ │ └── ProductRepository.java**

**│ │ │ └── resources**

**│ │ │ └── application.properties**

**│ │ └── test**

**│ │ └── java**

**│ │ └── com**

**│ │ └── ecommerce**

**│ │ └── AppTest.java**

**├── user-service**

**│ ├── Dockerfile**

**│ ├── pom.xml**

**│ ├── src**

**│ │ ├── main**

**│ │ │ ├── java**

**│ │ │ │ └── com**

**│ │ │ │ └── ecommerce**

**│ │ │ │ └── userservice**

**│ │ │ │ ├── controller**

**│ │ │ │ │ └── AuthController.java**

**│ │ │ │ ├── entity**

**│ │ │ │ │ └── User.java**

**│ │ │ │ ├── repository**

**│ │ │ │ │ └── UserRepository.java**

**│ │ │ │ └── security**

**│ │ │ │ └── JwtUtil.java**

**│ │ │ └── resources**

**│ │ │ └── application.properties**

**│ │ └── test**

**│ │ └── java**

**│ │ └── com**

**│ │ └── ecommerce**

**│ │ └── AppTest.java**

* Write the code following the folder structure mentioned above.

## 1️⃣ PostgreSQL Configuration in docker-compose.yml

Make sure PostgreSQL is correctly set up in Docker Compose:

yaml

CopyEdit

services:

postgres:

image: postgres:17.4

container\_name: postgres\_db

environment:

POSTGRES\_USER: myuser

POSTGRES\_PASSWORD: mypassword

POSTGRES\_DB: ecommercedb

ports:

- "5432:5432"

networks:

- ecommerce-network

This creates a PostgreSQL instance with:

* Username: myuser
* Password: mypassword
* Database: ecommercedb
* Accessible on port 5432

## 2️⃣ Database Connection in application.properties

Each microservice should have the following PostgreSQL connection settings inside its src/main/resources/application.properties:

properties

CopyEdit

# PostgreSQL Configuration

spring.datasource.url=jdbc:postgresql://postgres\_db:5432/ecommercedb

spring.datasource.username=myuser

spring.datasource.password=mypassword

spring.datasource.driver-class-name=org.postgresql.Driver

# Hibernate Properties

spring.jpa.database-platform=org.hibernate.dialect.PostgreSQLDialect

spring.jpa.hibernate.ddl-auto=update

spring.jpa.show-sql=true

spring.sql.init.mode=always

This ensures: ✔ The microservices connect to PostgreSQL via postgres\_db:5432  
✔ Hibernate manages database schema updates automatically  
✔ SQL queries are logged for debugging

## 3️⃣ Running PostgreSQL with Docker Compose

Start PostgreSQL and all services using:

sh

CopyEdit

docker-compose up -d

Check running containers:

sh

CopyEdit

docker ps

To stop:

sh

CopyEdit

docker-compose down

## 4️⃣ Verify PostgreSQL Connection

Once running, access PostgreSQL inside the container:

sh

CopyEdit

docker exec -it postgres\_db psql -U myuser -d ecommercedb

Run:

sql

CopyEdit

\dt

Expected Output:  
It should list tables like:

pgsql

CopyEdit

List of relations

Schema | Name | Type | Owner

--------+--------+-------+--------

public | orders | table | myuser

(1 row)

If no tables are found, ensure that: ✅ The microservices are running  
✅ Hibernate is configured to create tables

**Build the code**

1) cd C:\Users\Administrator\Desktop\Microservices\user-service

mvn clean package -DskipTests

2) cd C:\Users\Administrator\Desktop\Microservices\product-service

mvn clean package -DskipTests

3) cd C:\Users\Administrator\Desktop\Microservices\order-service

mvn clean package -DskipTests

4) cd C:\Users\Administrator\Desktop\Microservices\payment-service

mvn clean package -DskipTests

**Running with Docker Compose**

Navigate to the root directory where docker-compose.yml is located and build the images:

docker-compose up -d –build

This will: ✅ Build all images

✅ Start the PostgreSQL database, Kafka, and all microservices

To check running containers:

docker ps

To see logs for a specific service (e.g., user-service):

docker logs user-service --tail 50

To stop and remove all services:

docker-compose down -v