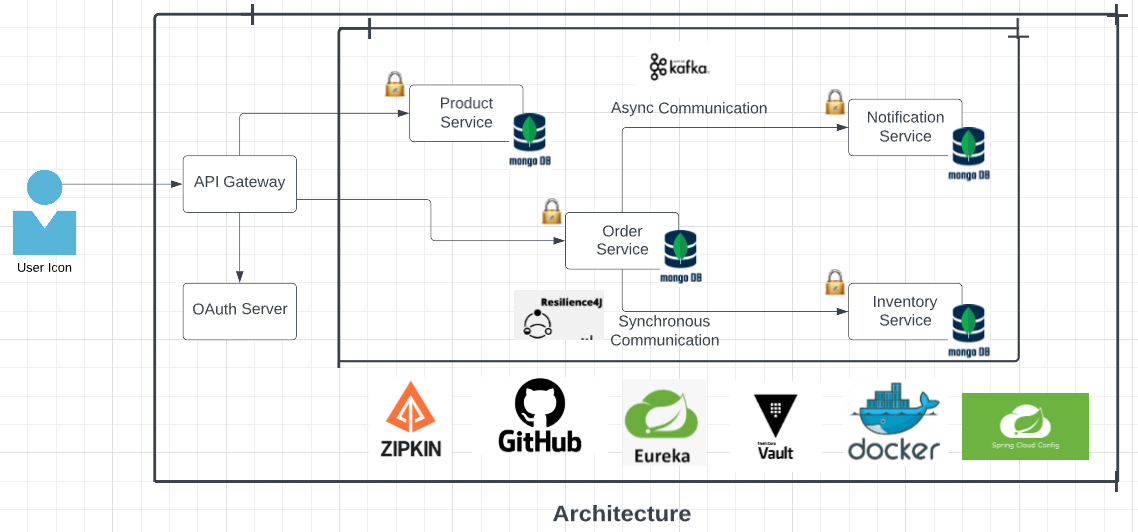
**Architecture diagram**

****

**DFD**

**Diagram

Description automatically generated**

Mainly there are 4 services as below:

1. Product Service: create and view products, acts as a product catalog.
2. Order Service: can order products.
3. Inventory Service: can check if product is in stock or not.
4. Notification Service: Can send notifications, after order is placed.

Order service, Inventory Service and Notification Service are going to interact with each other in synchronous and Asynchronous communication. Each Service has their own database i.e., MongoDB.

**Discovery Server:** This is an application which holds the information about all client service applications. Every Micro service will register into the Eureka server and Eureka server knows all the client applications running on each port and IP address. Eureka Server is also known as Discovery Server.

Graphical user interface, text

Description automatically generated with medium confidence

Día: Discovery Server

**API Gateway**: Its acts like a gatekeeper which sends requests to different services ex. When user wants to communicate with services, they no need to provide any IP address, instead they will call API Gateway and from there the requests are redirected to respective services. Our applications also secured with this approach since at gateway level we are using authorization server called Keycloak.

**Api Gateway Routing to Services as below:**

Inorder to Route to respective service after received a request from the user we need to add properties as below in API GATEWAY application.properties

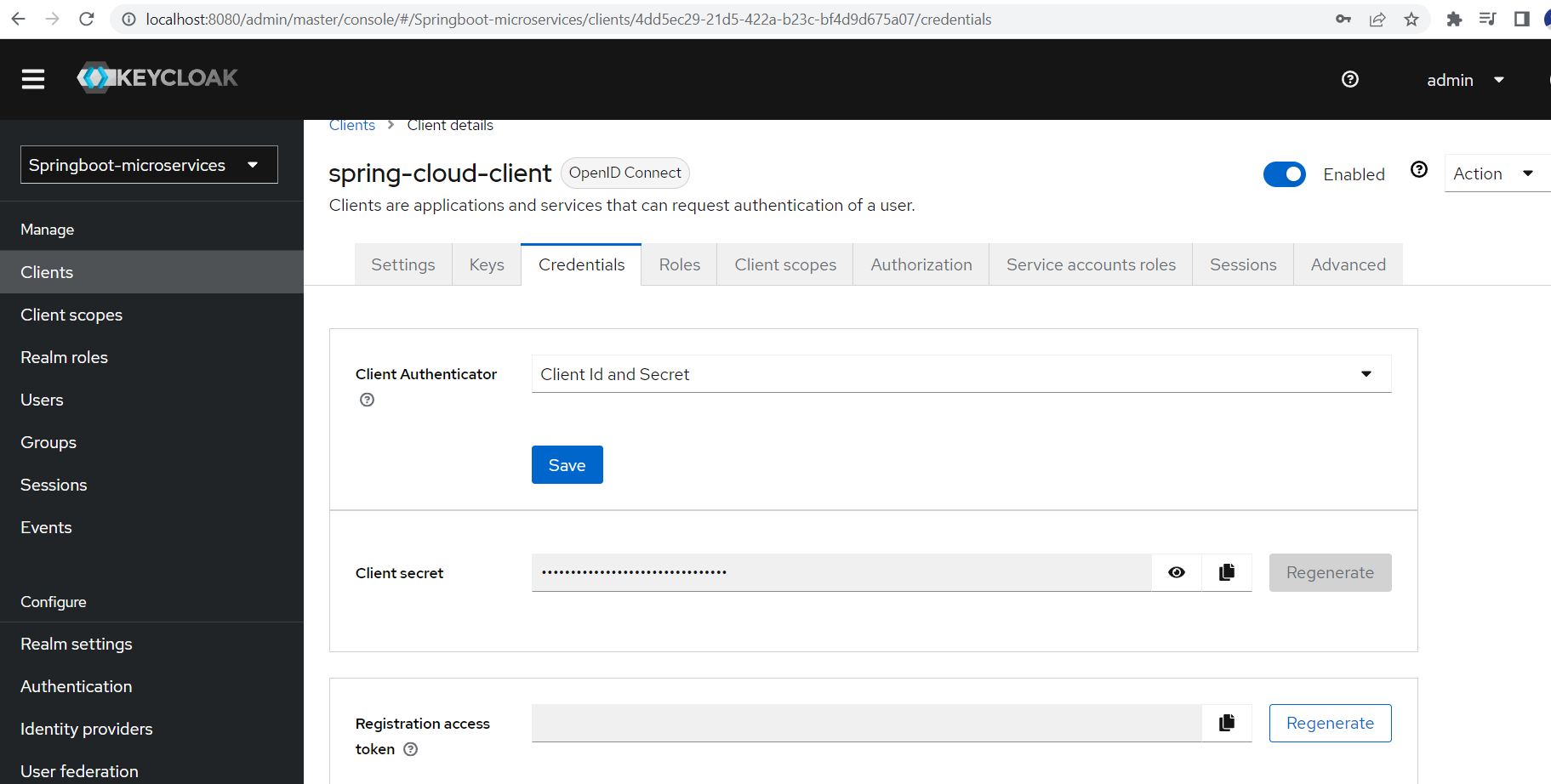
Text

Description automatically generated

**Authentication and Authorization Using KEYCLOAK**

Here we will add credentials in Keycloak and from services we have to connect to this server inorder to validate the requests. In application.properties we have to add as below.

spring.security.oauth2.resourceserver.jwt.issuer-uri= http://localhost:8080/realms/Springboot-microservices



API Authentication Process as below:

1. **If any Unauthorized request came in, then we will get 401 Unauthorized response as below**

Graphical user interface, text, application, email

Description automatically generated

1. **First we have to get access token by giving credentials**

Graphical user interface, application

Description automatically generated

1. Once provided credentials are authenticated we will receive Authentication complete as below

Graphical user interface, website

Description automatically generated

1. After Authorization we will receive token, then by using token we have to call certain API

Graphical user interface, text, application

Description automatically generated

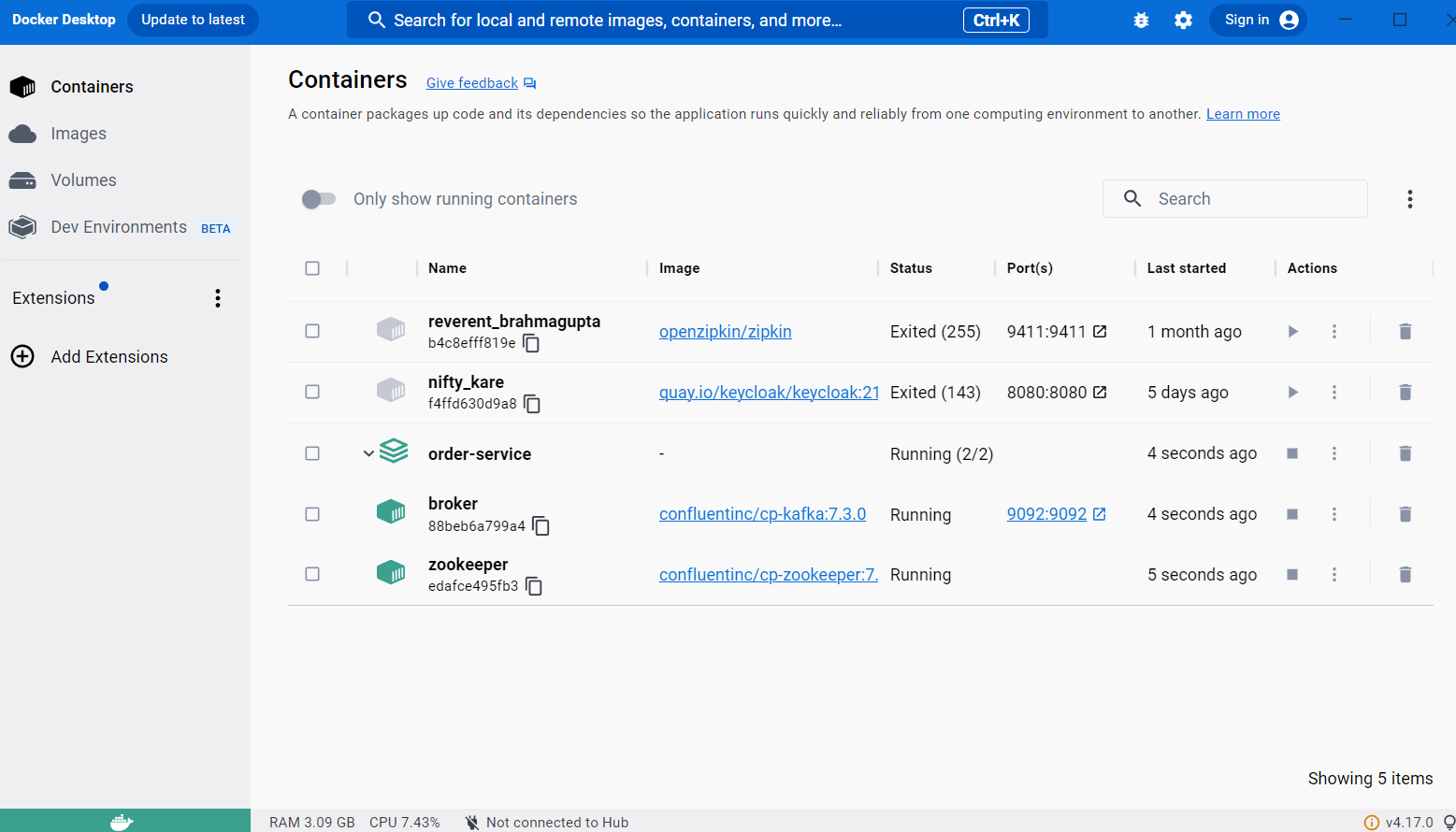
1. Finally Product is created in Mongodb database, we received 201 created

Graphical user interface, text, application, email

Description automatically generated

**Docker containers**

**We have used docker for running kafka broker ,zookeeper and also zipkin and keycloak servers**

****

**Github Repository:**

<https://github.com/premChowdaryNim?tab=repositories>

Request: <http://localhost:57997/api/order>

  {

                    "orderLineItemsDtoList":[

                        {

                        "skuCode":"oppo",

                        "price":999,

                        "quantity":3

                        }

                    ]

}

Client secret : oerayXpAhs9zUJkKDyZSF8GowKCjmJLr

Client id : spring-cloud-client

Client Auth : send as basic auth header

Inventory

GET : http://localhost:58224/api/inventory?skuCode=Samsung\_galaxy&skuCode=Iphone 14 pro max