ECOLE NORMALE SUPÉRIEURE DE
L'ENSEIGNEMENT TECHNIQUE DE MOHAMMEDIA

UNIVERSITÉ HASSAN II DE CASABLANCA

ENSET

المدرسة العليا لأساتذة التعليم التقني
المحمدية
جامعة الحسن الثاني بالدار البيضاء

DEPARTEMENT MATHEMATIQUES ET INFORMATIQUE

Activité Pratique N° 2

Systèmes Distribués

Filière :

« Génie du Logiciel et des Systèmes Informatiques Distribués »

GLSID

Architectures Micro-services avec Spring cloud

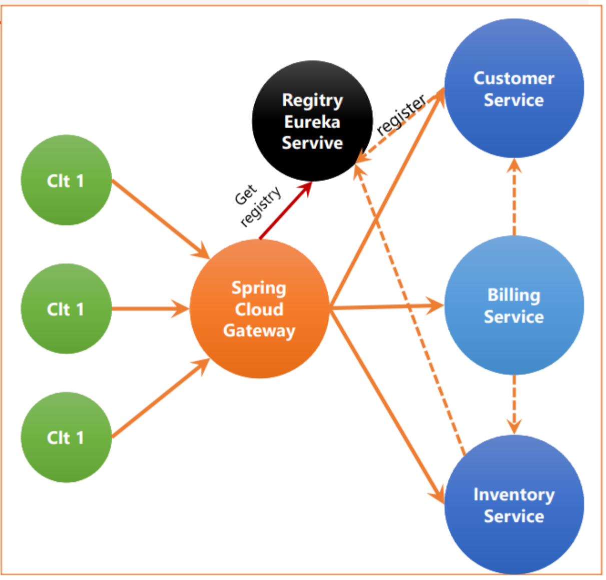
Réalisé par :

Najat ES-SAYYAD

Année Universitaire : 2023-2024

Activité Pratique : Travail à faire

1. Créer le micro service Customer-service
   - Créer l'entité Customer
   - Créer l'interface CustomerRepository basée sur Spring Data
   - Déployer l'API Restful du micro-service en utilisant Spring Data Rest
   - Tester le Micro service
2. Créer le micro service Inventory-service
   - Créer l'entité Product
   - Créer l'interface ProductRepository basée sur Spring Data
   - Déployer l'API Restful du micro-service en utilisant Spring Data Rest
   - Tester le Micro service
3. Créer la Gateway service en utilisant Spring Cloud Gateway
   1. Tester la Service proxy en utilisant une configuration Statique basée sur le fichier application.yml
   2. Tester la Service proxy en utilisant une configuration Statique basée une configuration Java
4. Créer l'annuaire Registry Service basé sur NetFlix Eureka Server
5. Tester le proxy en utilisant une configuration dynamique de Gestion des routes vers les micro services enregistrés dans l'annuaire Eureka Server
6. Créer Le service Billing-Service en utilisant Open Feign pour communiquer avec les services Customer-service et Inventory-service
7. Créer un client Angular qui permet d'afficher une facture

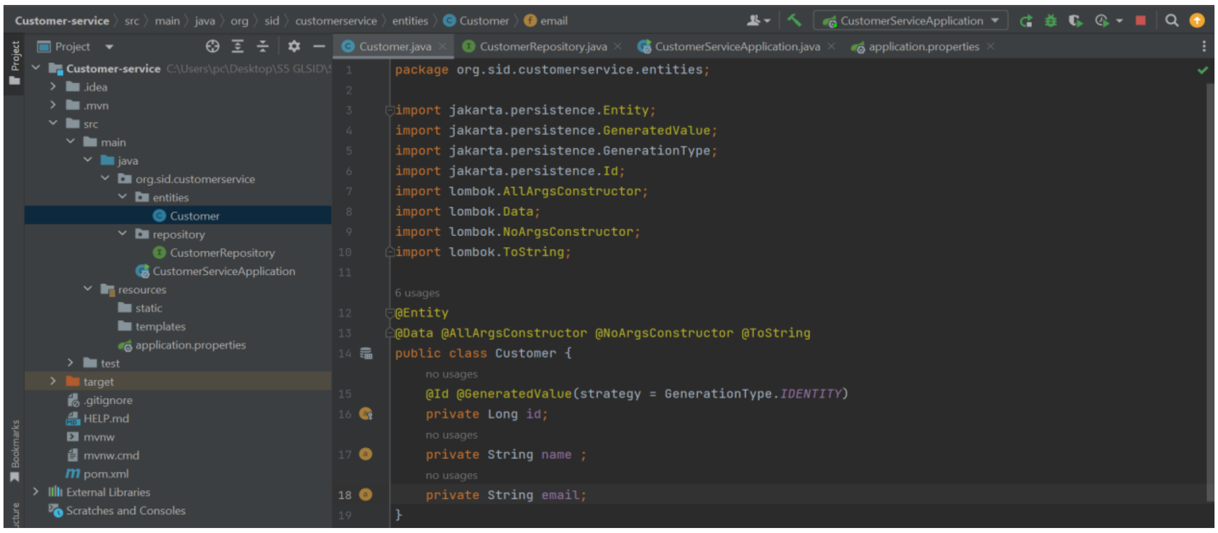


Première partie

(Customer-Service, Inventory-Service, Spring Cloud Gateway, Eureka Discovery)

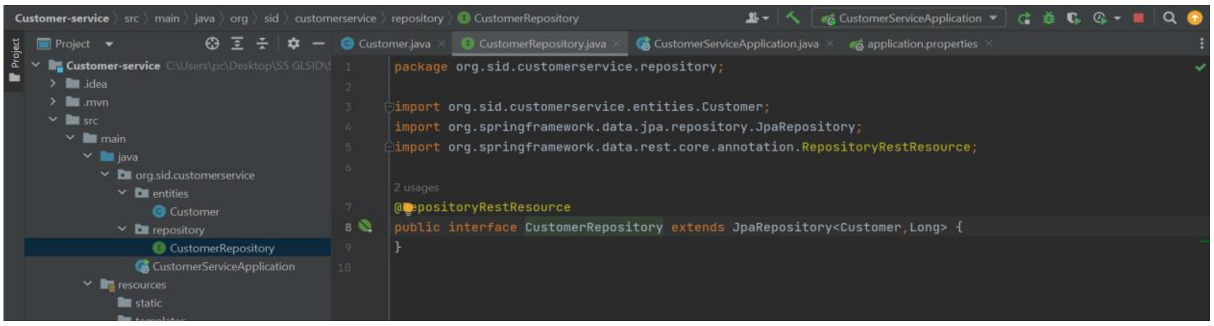
Customer service :

Entité JPA Customer :



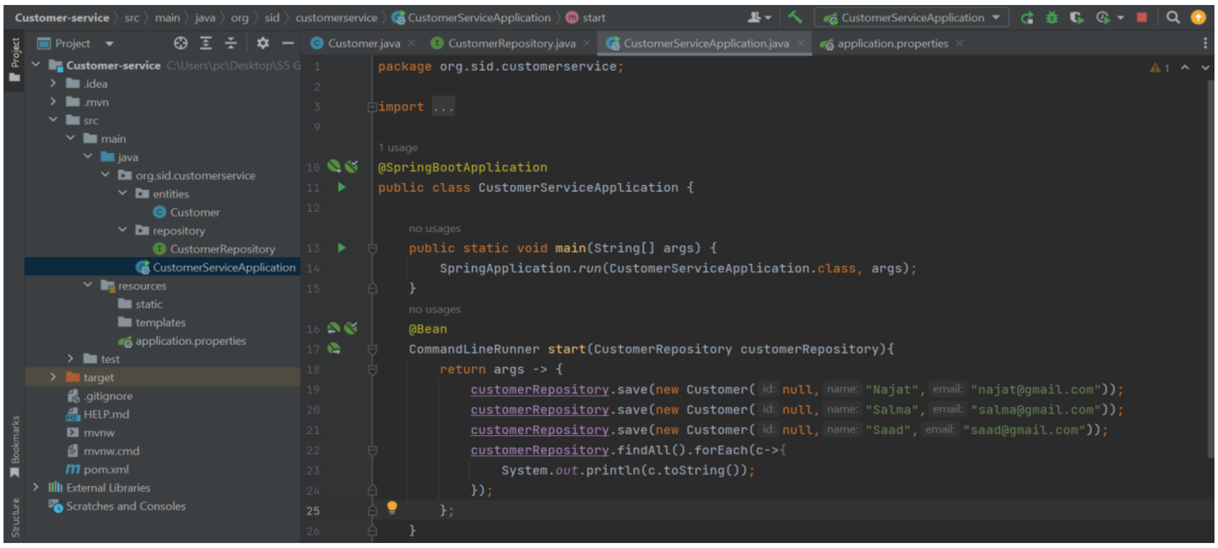
```java
package org.sid.customerservice.entities;

import jakarta.persistence.Entity;
import jakarta.persistence.GeneratedValue;
import jakarta.persistence.GenerationType;
import jakarta.persistence.Id;
import lombok.AllArgsConstructor;
import lombok.Data;
import lombok.NoArgsConstructor;
import lombok.ToString;

@Entity
@Data @AllArgsConstructor @NoArgsConstructor @ToString
public class Customer {
    @Id @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;
    private String name ;
    private String email;
}
```

CustomerRepository :



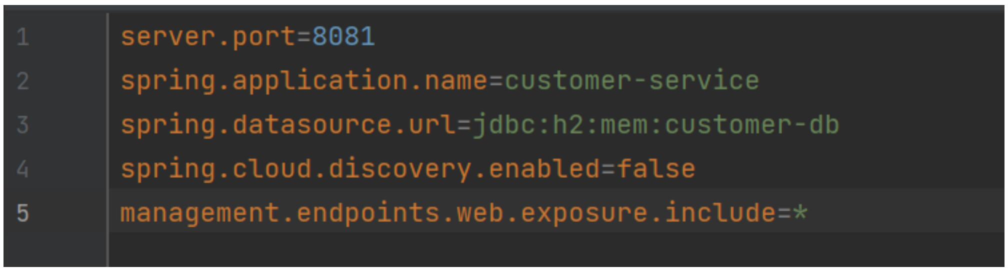
```java
package org.sid.customerservice.repository;

import org.sid.customerservice.entities.Customer;
import org.springframework.data.jpa.repository.JpaRepository;
import org.springframework.data.rest.core.annotation.RepositoryRestResource;

@RepositoryRestResource
public interface CustomerRepository extends JpaRepository<Customer,Long> {
}
```

Application :



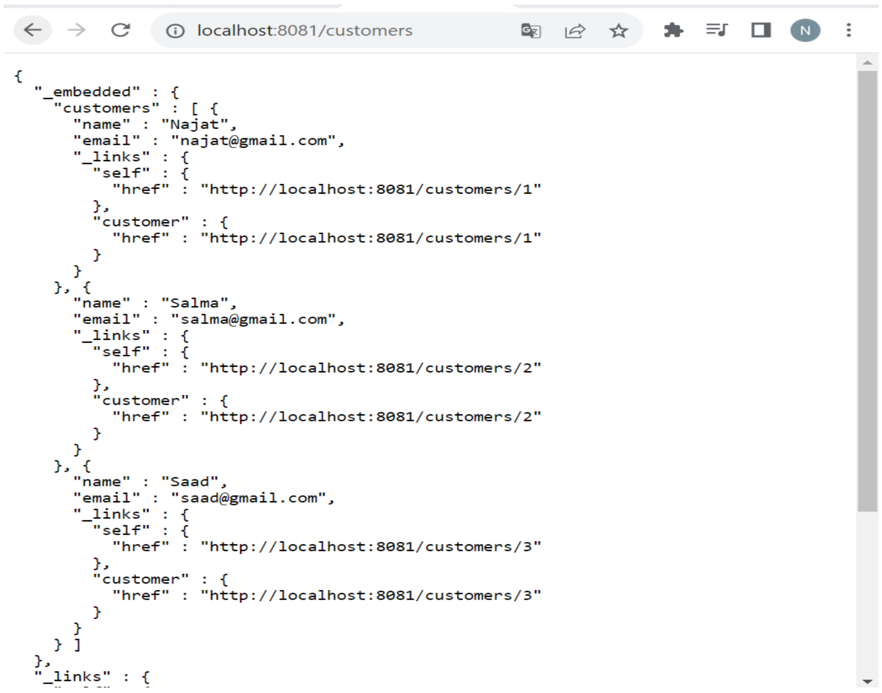
```java
package org.sid.customerservice;

import ...

@SpringBootApplication
public class CustomerServiceApplication {

    public static void main(String[] args) {
        SpringApplication.run(CustomerServiceApplication.class, args);
    }
    @Bean
    CommandLineRunner start(CustomerRepository customerRepository){
        return args -> {
            customerRepository.save(new Customer( null, "Najat", "najat@gmail.com"));
            customerRepository.save(new Customer( null, "Salma", "salma@gmail.com"));
            customerRepository.save(new Customer( null, "Saad", "saad@gmail.com"));
            customerRepository.findAll().forEach(c->{
                System.out.println(c.toString());
            });
        };
    }
```

La configuration :



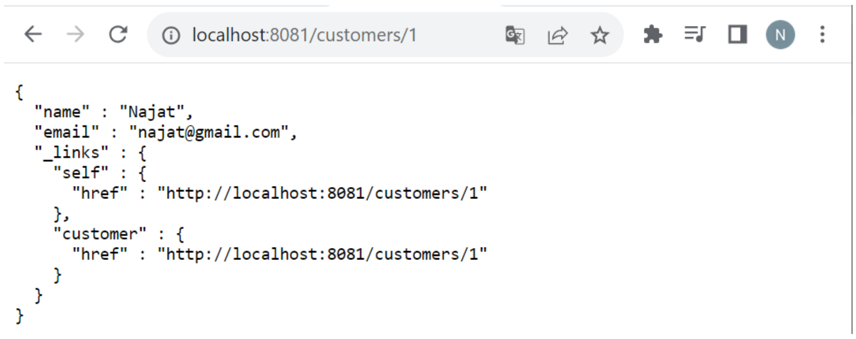
```
server.port=8081
spring.application.name=customer-service
spring.datasource.url=jdbc:h2:mem:customer-db
spring.cloud.discovery.enabled=false
management.endpoints.web.exposure.include=*
```

Résultat :



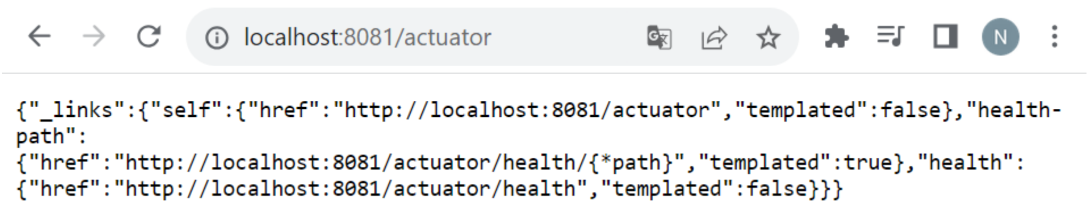
localhost:8081/customers

```
{
  "_embedded" : {
    "customers" : [ {
      "name" : "Najat",
      "email" : "najat@gmail.com",
      "_links" : {
        "self" : {
          "href" : "http://localhost:8081/customers/1"
        },
        "customer" : {
          "href" : "http://localhost:8081/customers/1"
        }
      }
    }, {
      "name" : "Salma",
      "email" : "salma@gmail.com",
      "_links" : {
        "self" : {
          "href" : "http://localhost:8081/customers/2"
        },
        "customer" : {
          "href" : "http://localhost:8081/customers/2"
        }
      }
    }, {
      "name" : "Saad",
      "email" : "saad@gmail.com",
      "_links" : {
        "self" : {
          "href" : "http://localhost:8081/customers/3"
        },
        "customer" : {
          "href" : "http://localhost:8081/customers/3"
        }
      }
    } ]
  },
  "_links" : {
```



localhost:8081/customers/1

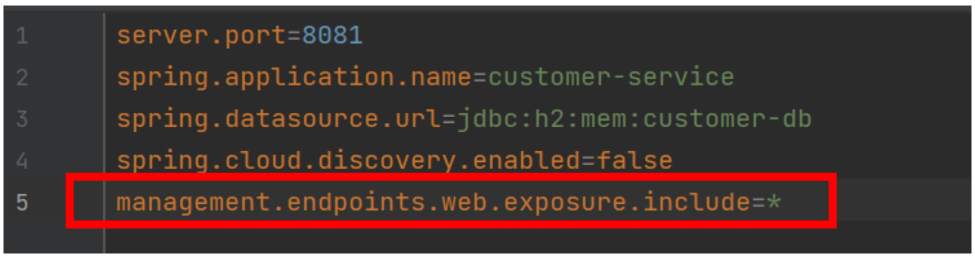
```
{
  "name" : "Najat",
  "email" : "najat@gmail.com",
  "_links" : {
    "self" : {
      "href" : "http://localhost:8081/customers/1"
    },
    "customer" : {
      "href" : "http://localhost:8081/customers/1"
    }
  }
}
```



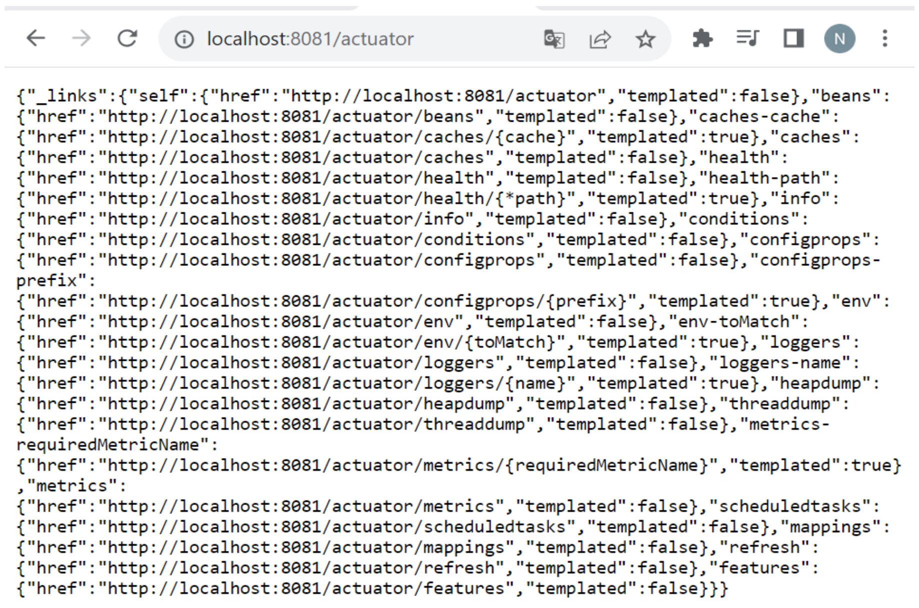
localhost:8081/actuator

```
{"_links":{"self":{"href":"http://localhost:8081/actuator","templated":false},"health-path":
{"href":"http://localhost:8081/actuator/health/{*path}","templated":true},"health":
{"href":"http://localhost:8081/actuator/health","templated":false}}}
```

On ajoute management.endpoints.web.exposure.include=*

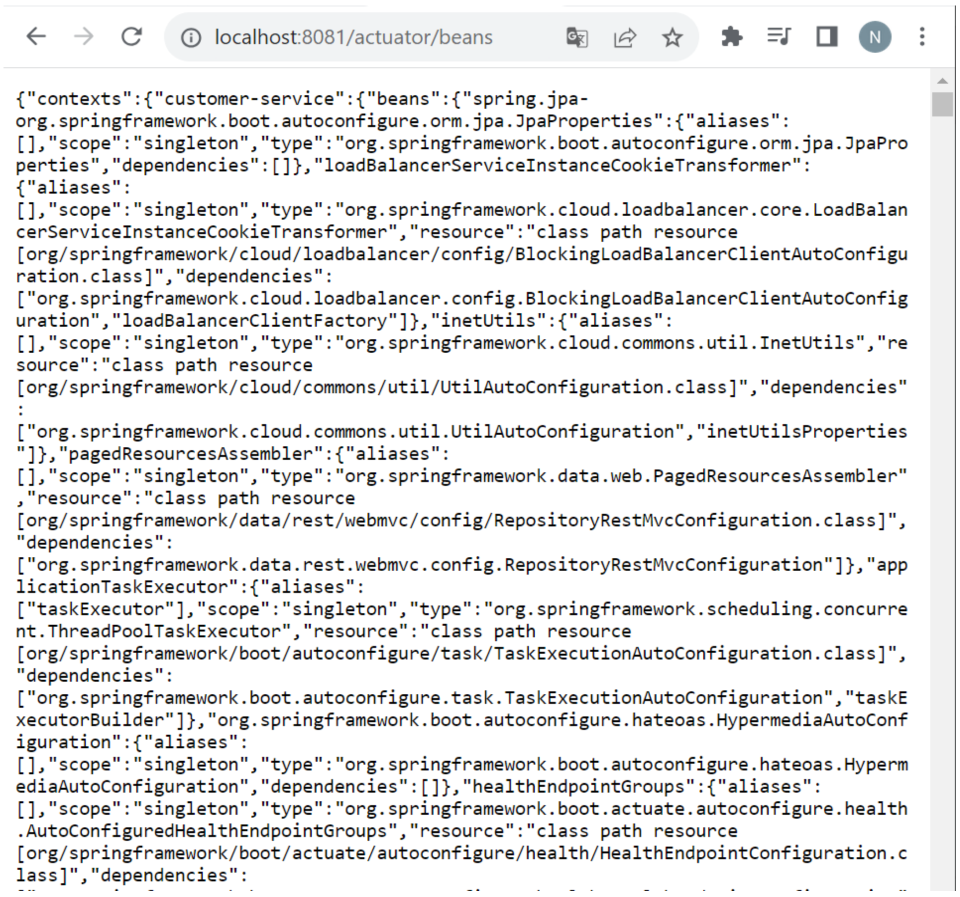


```
server.port=8081
spring.application.name=customer-service
spring.datasource.url=jdbc:h2:mem:customer-db
spring.cloud.discovery.enabled=false
management.endpoints.web.exposure.include=*
```



localhost:8081/actuator

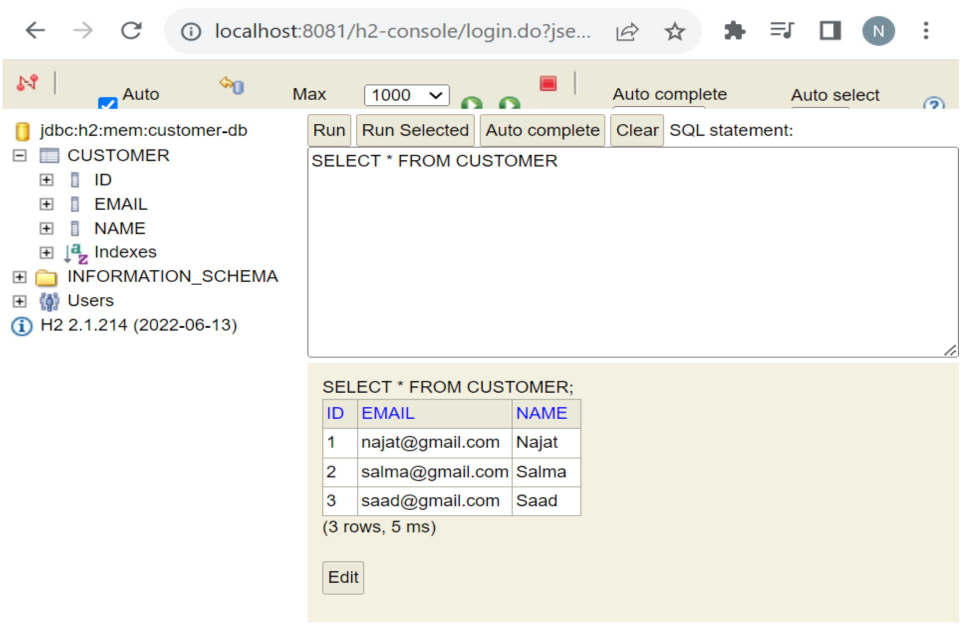
```
{"_links":{"self":{"href":"http://localhost:8081/actuator","templated":false},"beans":
{"href":"http://localhost:8081/actuator/beans","templated":false},"caches-cache":
{"href":"http://localhost:8081/actuator/caches/{cache}","templated":true},"caches":
{"href":"http://localhost:8081/actuator/caches","templated":false},"health":
{"href":"http://localhost:8081/actuator/health","templated":false},"health-path":
{"href":"http://localhost:8081/actuator/health/{*path}","templated":true},"info":
{"href":"http://localhost:8081/actuator/info","templated":false},"conditions":
{"href":"http://localhost:8081/actuator/conditions","templated":false},"configprops":
{"href":"http://localhost:8081/actuator/configprops","templated":false},"configprops-
prefix":
{"href":"http://localhost:8081/actuator/configprops/{prefix}","templated":true},"env":
{"href":"http://localhost:8081/actuator/env","templated":false},"env-toMatch":
{"href":"http://localhost:8081/actuator/env/{toMatch}","templated":true},"loggers":
{"href":"http://localhost:8081/actuator/loggers","templated":false},"loggers-name":
{"href":"http://localhost:8081/actuator/loggers/{name}","templated":true},"heapdump":
{"href":"http://localhost:8081/actuator/heapdump","templated":false},"threaddump":
{"href":"http://localhost:8081/actuator/threaddump","templated":false},"metrics-
requiredMetricName":
{"href":"http://localhost:8081/actuator/metrics/{requiredMetricName}","templated":true}
,"metrics":
{"href":"http://localhost:8081/actuator/metrics","templated":false},"scheduledtasks":
{"href":"http://localhost:8081/actuator/scheduledtasks","templated":false},"mappings":
{"href":"http://localhost:8081/actuator/mappings","templated":false},"refresh":
{"href":"http://localhost:8081/actuator/refresh","templated":false},"features":
{"href":"http://localhost:8081/actuator/features","templated":false}}}
```



localhost:8081/actuator/beans

```
{"contexts":{"customer-service":{"beans":{"spring.jpa-
org.springframework.boot.autoconfigure.orm.jpa.JpaProperties":{"aliases":
[],"scope":"singleton","type":"org.springframework.boot.autoconfigure.orm.jpa.JpaPro
perties","dependencies":[]},"loadBalancerServiceInstanceCookieTransformer":
{"aliases":
[],"scope":"singleton","type":"org.springframework.cloud.loadbalancer.core.LoadBalan
cerServiceInstanceCookieTransformer","resource":"class path resource
[org/springframework/cloud/loadbalancer/config/BlockingLoadBalancerClientAutoConfigu
ration.class]","dependencies":
["org.springframework.cloud.loadbalancer.config.BlockingLoadBalancerClientAutoConfig
uration","loadBalancerClientFactory"]},"inetUtils":{"aliases":
[],"scope":"singleton","type":"org.springframework.cloud.commons.util.InetUtils","re
source":"class path resource
[org/springframework/cloud/commons/util/UtilAutoConfiguration.class]","dependencies"
:
["org.springframework.cloud.commons.util.UtilAutoConfiguration","inetUtilsProperties
"]},"pagedResourcesAssembler":{"aliases":
[],"scope":"singleton","type":"org.springframework.data.web.PagedResourcesAssembler"
,"resource":"class path resource
[org/springframework/data/rest/webmvc/config/RepositoryRestMvcConfiguration.class]",
"dependencies":
["org.springframework.data.rest.webmvc.config.RepositoryRestMvcConfiguration"]},"app
licationTaskExecutor":{"aliases":
["taskExecutor"],"scope":"singleton","type":"org.springframework.scheduling.concurre
nt.ThreadPoolTaskExecutor","resource":"class path resource
[org/springframework/boot/autoconfigure/task/TaskExecutionAutoConfiguration.class]",
"dependencies":
["org.springframework.boot.autoconfigure.task.TaskExecutionAutoConfiguration","taskE
xecutorBuilder"]},"org.springframework.boot.autoconfigure.hateoas.HypermediaAutoConf
iguration":{"aliases":
[],"scope":"singleton","type":"org.springframework.boot.autoconfigure.hateoas.Hyperm
ediaAutoConfiguration","dependencies":[]},"healthEndpointGroups":{"aliases":
[],"scope":"singleton","type":"org.springframework.boot.actuate.autoconfigure.health
.AutoConfiguredHealthEndpointGroups","resource":"class path resource
[org/springframework/boot/actuate/autoconfigure/health/HealthEndpointConfiguration.c
lass]","dependencies":
```

La base de données :



jdbc:h2:mem:customer-db
CUSTOMER
ID
EMAIL
NAME
Indexes
INFORMATION_SCHEMA
Users
H2 2.1.214 (2022-06-13)

```
SELECT * FROM CUSTOMER
```

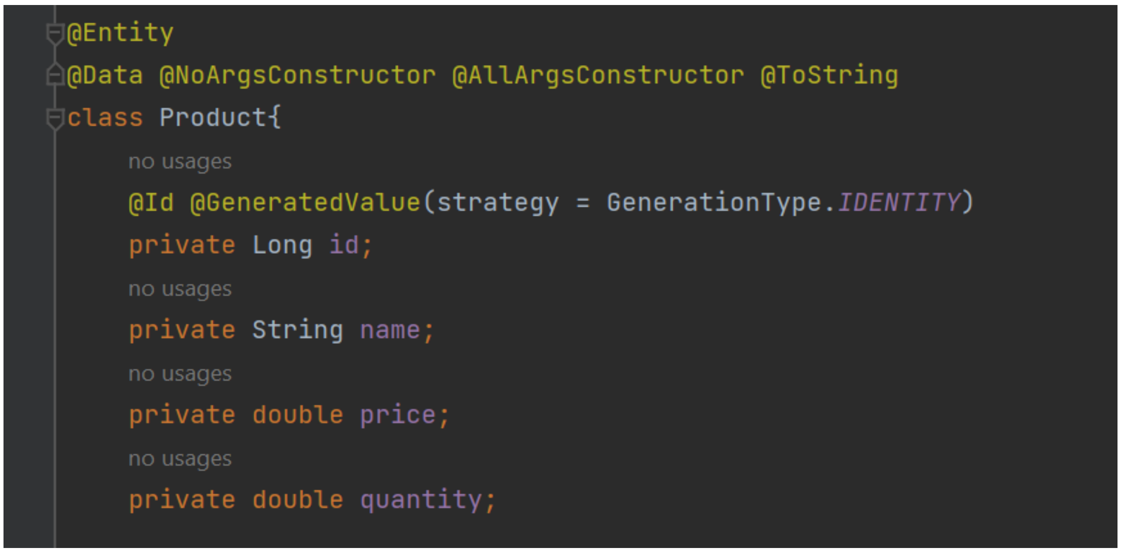
SELECT * FROM CUSTOMER;

ID	EMAIL	NAME
1	najat@gmail.com	Najat
2	salma@gmail.com	Salma
3	saad@gmail.com	Saad

(3 rows, 5 ms)

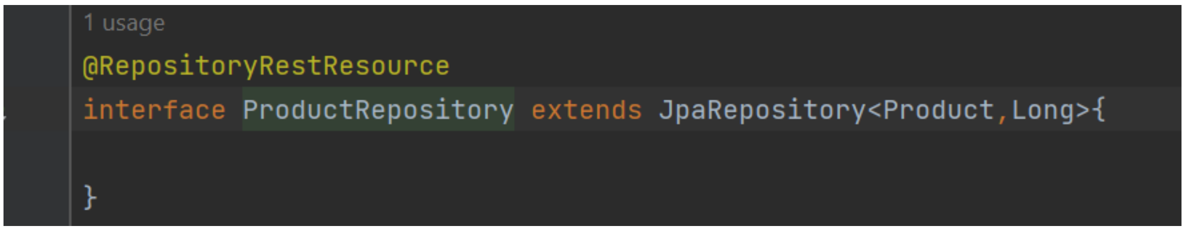
<u>**Product service :**</u>

Etité JPA Product :



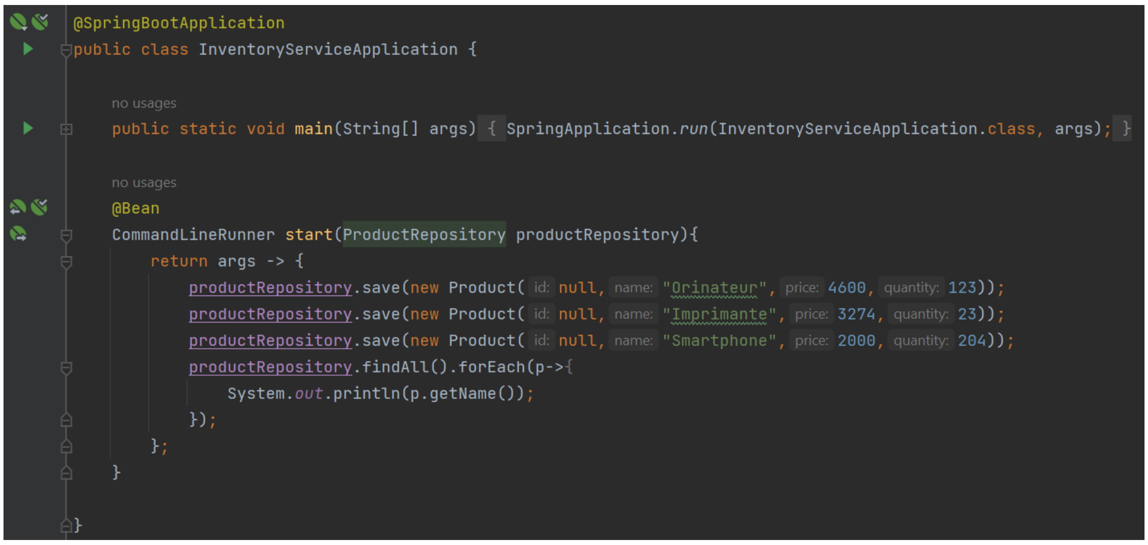
```
@Entity
@Data @NoArgsConstructor @AllArgsConstructor @ToString
class Product{
    no usages
    @Id @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;
    no usages
    private String name;
    no usages
    private double price;
    no usages
    private double quantity;
```

ProductRepository :



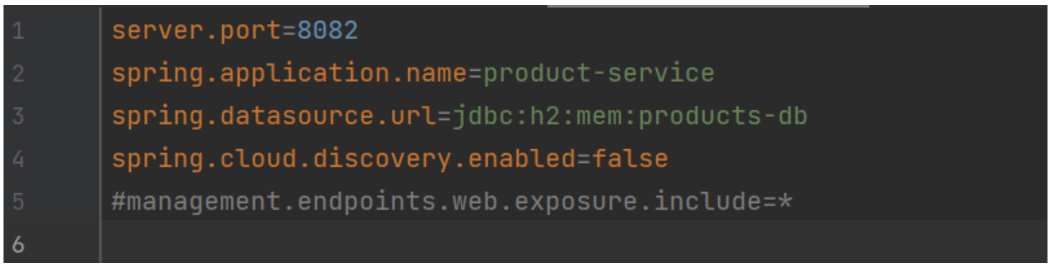
```
1 usage
@RepositoryRestResource
interface ProductRepository extends JpaRepository<Product,Long>{

}
```

Application :



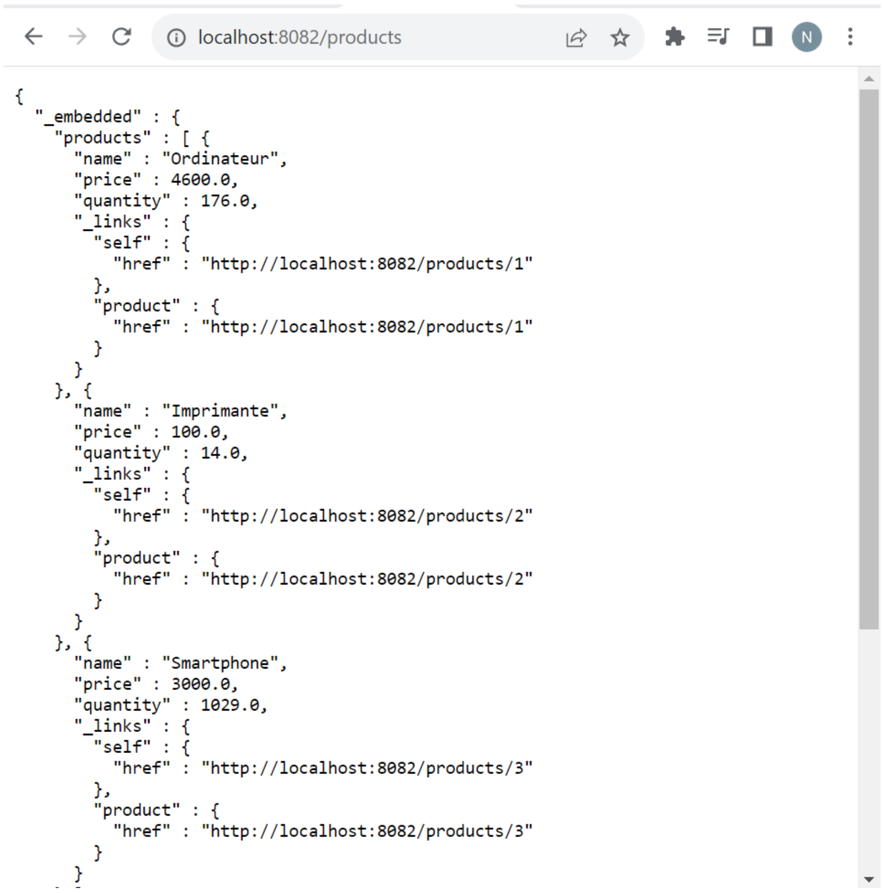
```
@SpringBootApplication
public class InventoryServiceApplication {

    no usages
    public static void main(String[] args) { SpringApplication.run(InventoryServiceApplication.class, args); }

    no usages
    @Bean
    CommandLineRunner start(ProductRepository productRepository){
        return args -> {
            productRepository.save(new Product( id: null, name: "Orinateur", price: 4600, quantity: 123));
            productRepository.save(new Product( id: null, name: "Imprimante", price: 3274, quantity: 23));
            productRepository.save(new Product( id: null, name: "Smartphone", price: 2000, quantity: 204));
            productRepository.findAll().forEach(p->{
                System.out.println(p.getName());
            });
        };
    }

}
```

Configuration :



```
server.port=8082
spring.application.name=product-service
spring.datasource.url=jdbc:h2:mem:products-db
spring.cloud.discovery.enabled=false
#management.endpoints.web.exposure.include=*

```

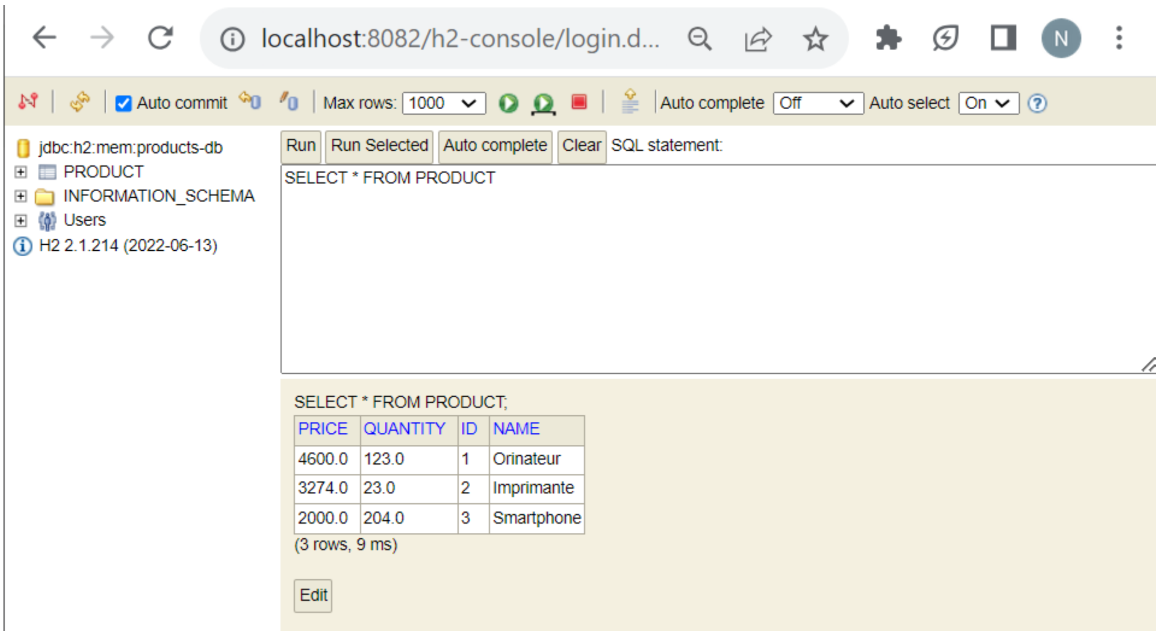
Résultat :



localhost:8082/products

```
{
  "_embedded" : {
    "products" : [ {
      "name" : "Ordinateur",
      "price" : 4600.0,
      "quantity" : 176.0,
      "_links" : {
        "self" : {
          "href" : "http://localhost:8082/products/1"
        },
        "product" : {
          "href" : "http://localhost:8082/products/1"
        }
      }
    }, {
      "name" : "Imprimante",
      "price" : 100.0,
      "quantity" : 14.0,
      "_links" : {
        "self" : {
          "href" : "http://localhost:8082/products/2"
        },
        "product" : {
          "href" : "http://localhost:8082/products/2"
        }
      }
    }, {
      "name" : "Smartphone",
      "price" : 3000.0,
      "quantity" : 1029.0,
      "_links" : {
        "self" : {
          "href" : "http://localhost:8082/products/3"
        },
        "product" : {
          "href" : "http://localhost:8082/products/3"
        }
      }
```

La base de données :



jdbc:h2:mem:products-db
PRODUCT
INFORMATION_SCHEMA
Users
H2 2.1.214 (2022-06-13)

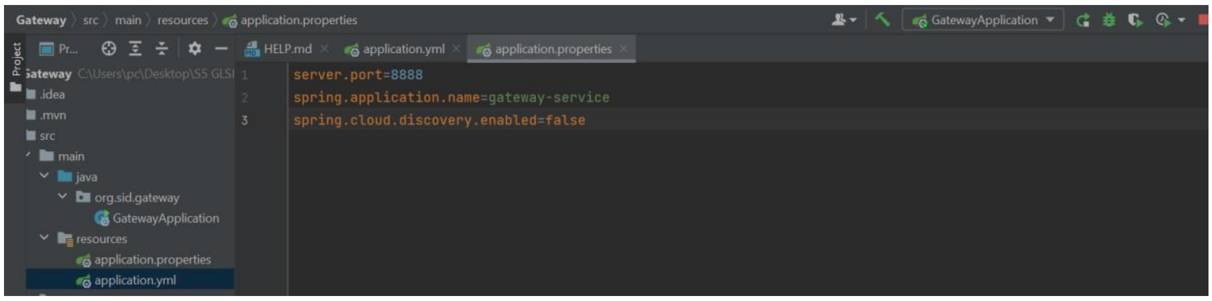
```
SELECT * FROM PRODUCT
```

SELECT * FROM PRODUCT;

PRICE	QUANTITY	ID	NAME
4600.0	123.0	1	Orinateur
3274.0	23.0	2	Imprimante
2000.0	204.0	3	Smartphone

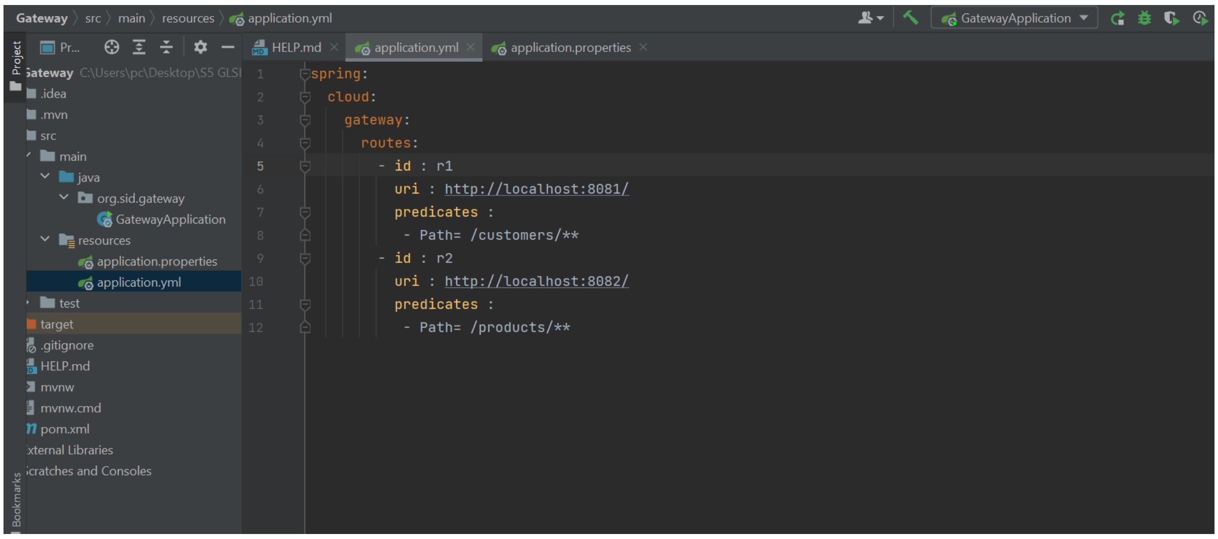
(3 rows, 9 ms)

Gateway-service :



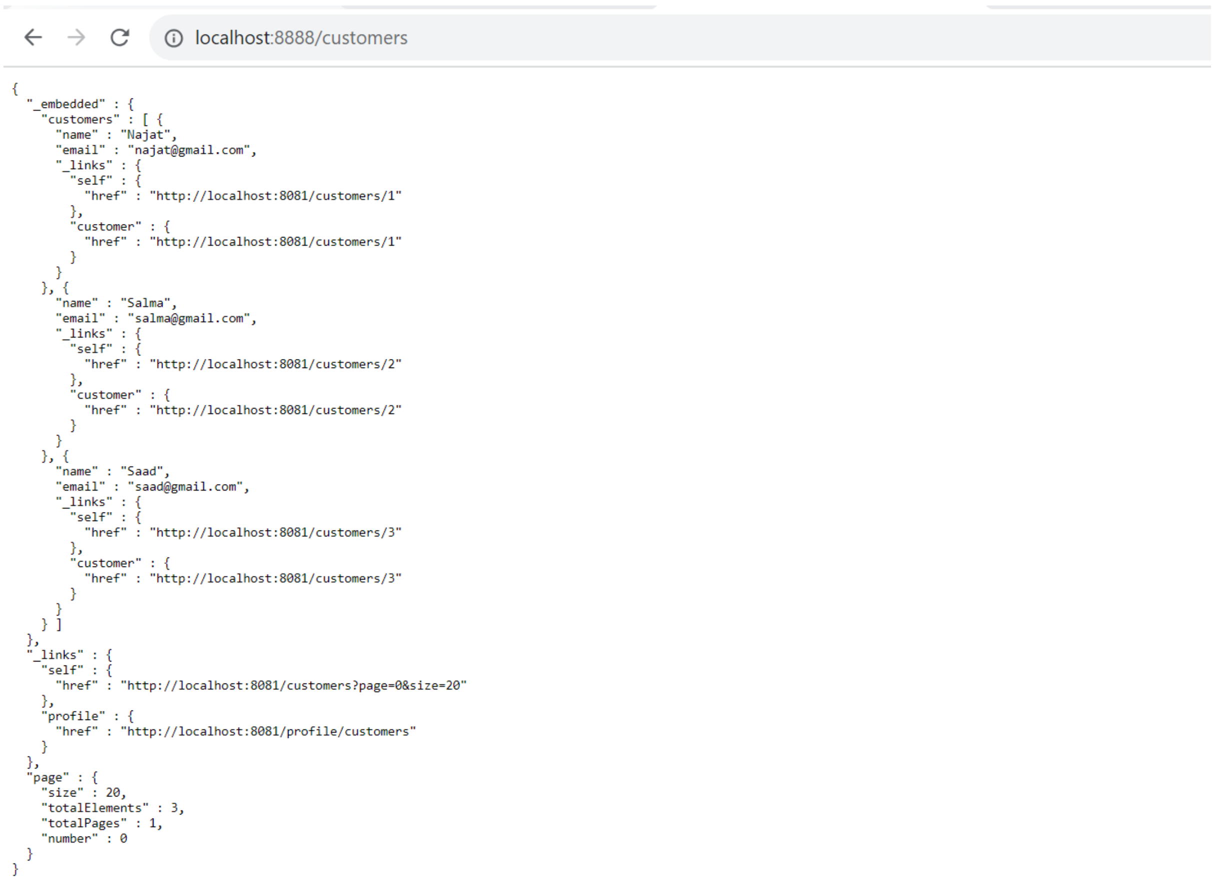
```
server.port=8888
spring.application.name=gateway-service
spring.cloud.discovery.enabled=false
```

Configuration de gateway d'une manière statique on utilisons yml file, on utilise statique quand on veut accéder à des routes qui sont externe à l'application et dont le nom de domaine est déjà connu et l'adresse ça ne change pas.

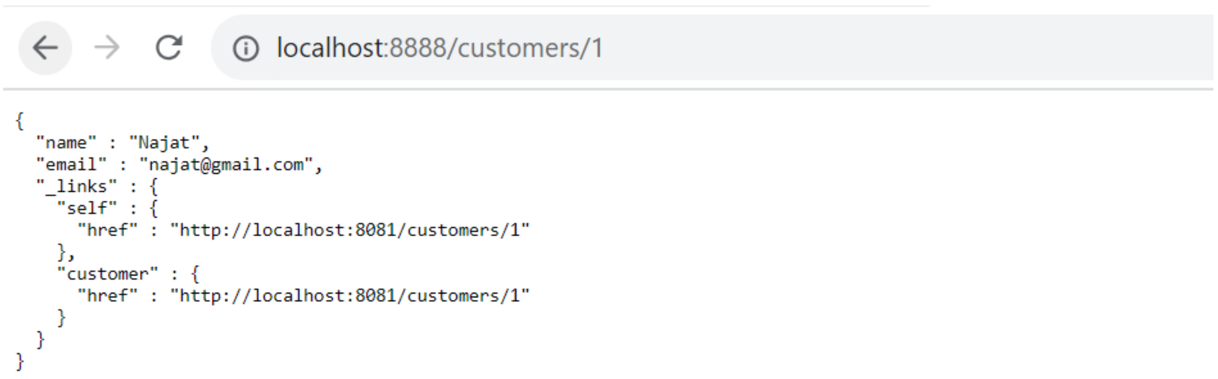


```
spring:
  cloud:
    gateway:
      routes:
        - id : r1
          uri : http://localhost:8081/
          predicates :
            - Path= /customers/**
        - id : r2
          uri : http://localhost:8082/
          predicates :
            - Path= /products/**
```

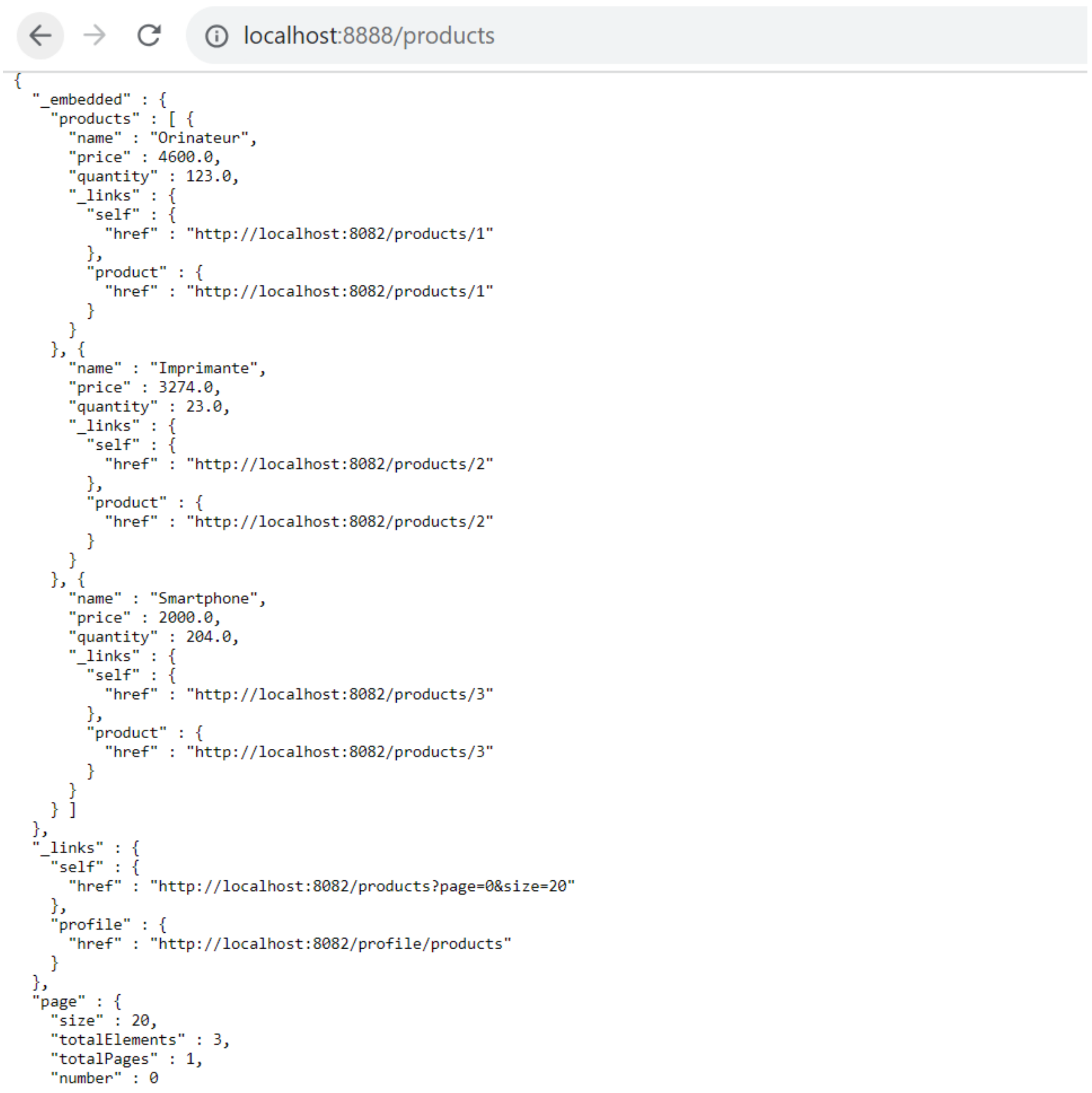
Résultat


localhost:8888/customers

```
{
  "_embedded" : {
    "customers" : [ {
      "name" : "Najat",
      "email" : "najat@gmail.com",
      "_links" : {
        "self" : {
          "href" : "http://localhost:8081/customers/1"
        },
        "customer" : {
          "href" : "http://localhost:8081/customers/1"
        }
      }
    }, {
      "name" : "Salma",
      "email" : "salma@gmail.com",
      "_links" : {
        "self" : {
          "href" : "http://localhost:8081/customers/2"
        },
        "customer" : {
          "href" : "http://localhost:8081/customers/2"
        }
      }
    }, {
      "name" : "Saad",
      "email" : "saad@gmail.com",
      "_links" : {
        "self" : {
          "href" : "http://localhost:8081/customers/3"
        },
        "customer" : {
          "href" : "http://localhost:8081/customers/3"
        }
      }
    } ]
  },
  "_links" : {
    "self" : {
      "href" : "http://localhost:8081/customers?page=0&size=20"
    },
    "profile" : {
      "href" : "http://localhost:8081/profile/customers"
    }
  },
  "page" : {
    "size" : 20,
    "totalElements" : 3,
    "totalPages" : 1,
    "number" : 0
  }
}
```


localhost:8888/customers/1

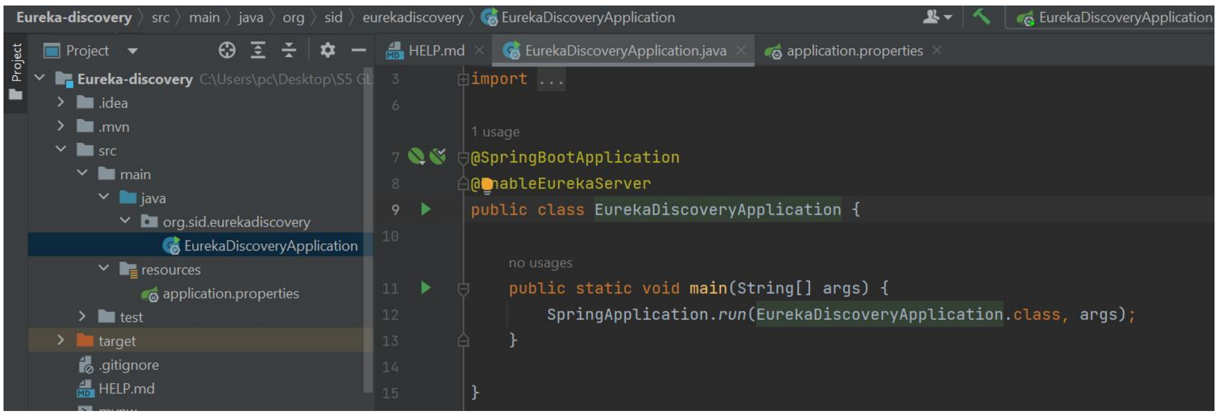
```
{
  "name" : "Najat",
  "email" : "najat@gmail.com",
  "_links" : {
    "self" : {
      "href" : "http://localhost:8081/customers/1"
    },
    "customer" : {
      "href" : "http://localhost:8081/customers/1"
    }
  }
}
```


localhost:8888/products

```
{
  "_embedded" : {
    "products" : [ {
      "name" : "Orinateur",
      "price" : 4600.0,
      "quantity" : 123.0,
      "_links" : {
        "self" : {
          "href" : "http://localhost:8082/products/1"
        },
        "product" : {
          "href" : "http://localhost:8082/products/1"
        }
      }
    }, {
      "name" : "Imprimante",
      "price" : 3274.0,
      "quantity" : 23.0,
      "_links" : {
        "self" : {
          "href" : "http://localhost:8082/products/2"
        },
        "product" : {
          "href" : "http://localhost:8082/products/2"
        }
      }
    }, {
      "name" : "Smartphone",
      "price" : 2000.0,
      "quantity" : 204.0,
      "_links" : {
        "self" : {
          "href" : "http://localhost:8082/products/3"
        },
        "product" : {
          "href" : "http://localhost:8082/products/3"
        }
      }
    } ]
  },
  "_links" : {
    "self" : {
      "href" : "http://localhost:8082/products?page=0&size=20"
    },
    "profile" : {
      "href" : "http://localhost:8082/profile/products"
    }
  },
  "page" : {
    "size" : 20,
    "totalElements" : 3,
    "totalPages" : 1,
    "number" : 0
```

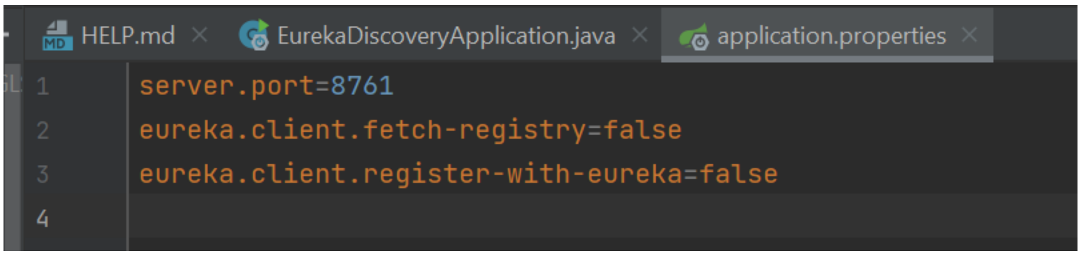
Eureka Discovry :

On ajoute l'annotation **@EnableEurekaServer**


Eureka-discovery 〉 src 〉 main 〉 java 〉 org 〉 sid 〉 eurekadiscovery 〉 EurekaDiscoveryApplication
EurekaDiscoveryApplication
Project
HELP.md
EurekaDiscoveryApplication.java
application.properties
Eureka-discovery C:\Users\pc\Desktop\S5 GL
.idea
.mvn
src
main
java
org.sid.eurekadiscovery
EurekaDiscoveryApplication
resources
application.properties
test
target
.gitignore
HELP.md

```
import ...

1 usage
@SpringBootApplication
@nableEurekaServer
public class EurekaDiscoveryApplication {

    no usages
    public static void main(String[] args) {
        SpringApplication.run(EurekaDiscoveryApplication.class, args);
    }

}
```

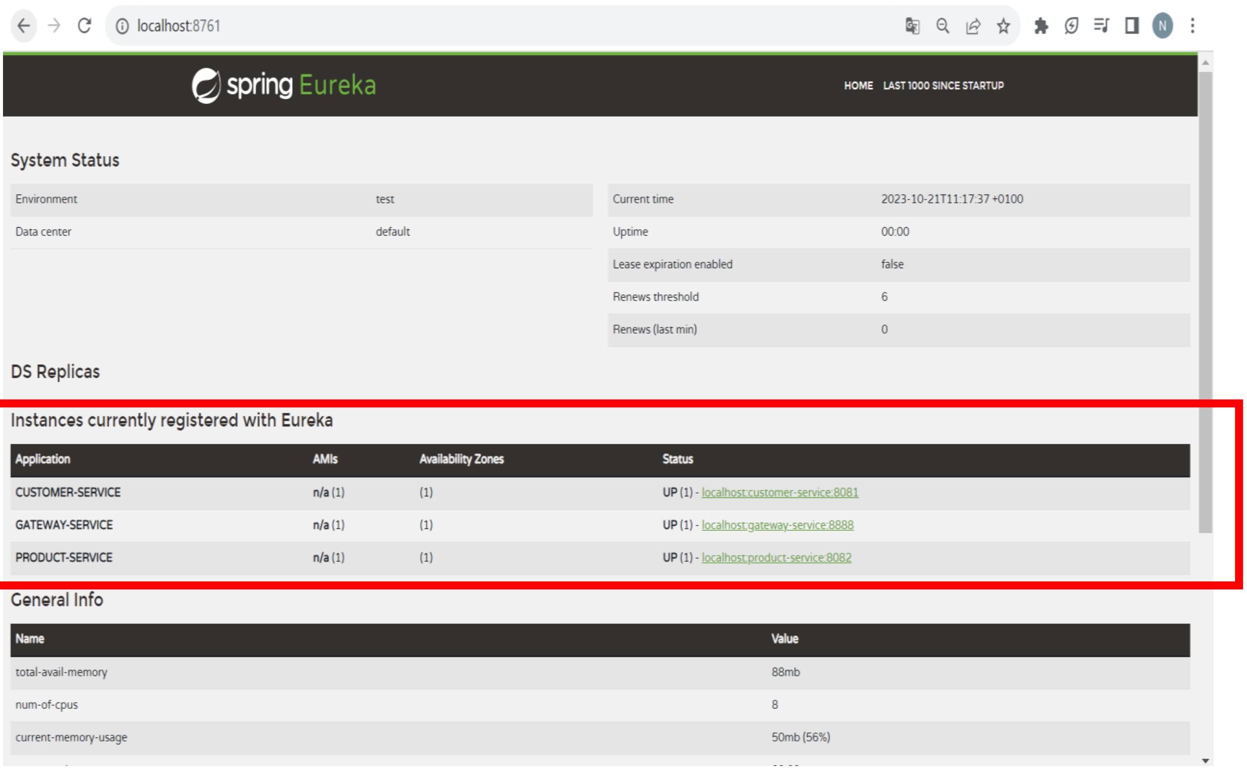
La configuration :


```
server.port=8761
eureka.client.fetch-registry=false
eureka.client.register-with-eureka=false
```

On active le discovery service au niveau de tous les microservices avec :

```
spring.cloud.discovery.enabled=true
```

Résultat :



System Status

Environment	test
Data center	default

Current time	2023-10-21T11:17:37 +0100
Uptime	00:00
Lease expiration enabled	false
Renews threshold	6
Renews (last min)	0

DS Replicas

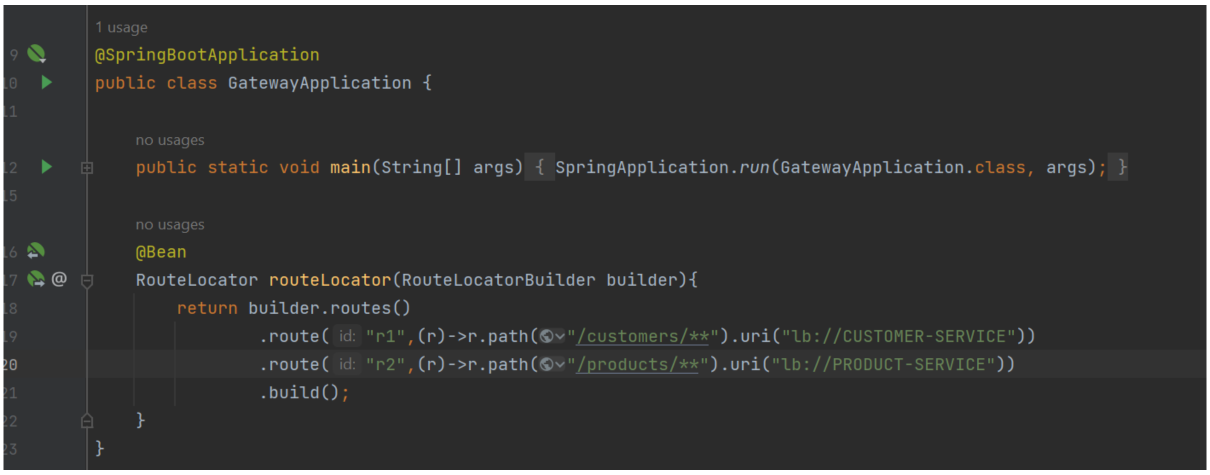
Instances currently registered with Eureka

Application	AMIs	Availability Zones	Status
CUSTOMER-SERVICE	n/a (1)	(1)	UP (1) - localhost:customer-service:8081
GATEWAY-SERVICE	n/a (1)	(1)	UP (1) - localhost:gateway-service:8888
PRODUCT-SERVICE	n/a (1)	(1)	UP (1) - localhost:product-service:8082

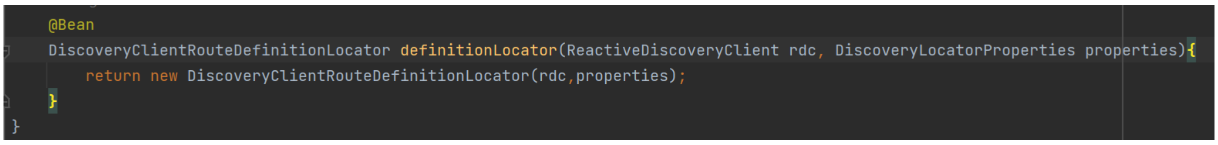
General Info

Name	Value
total-avail-memory	88mb
num-of-cpus	8
current-memory-usage	50mb (56%)

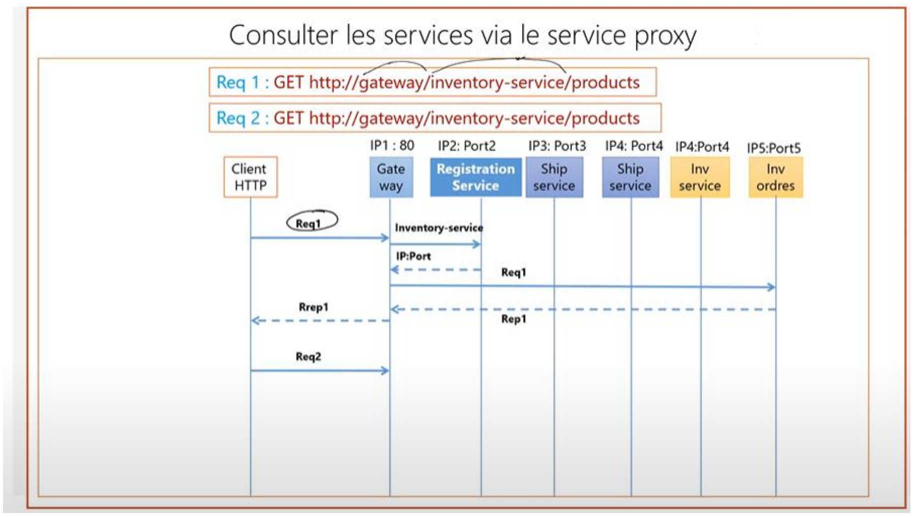
La configuration d'une manière statique

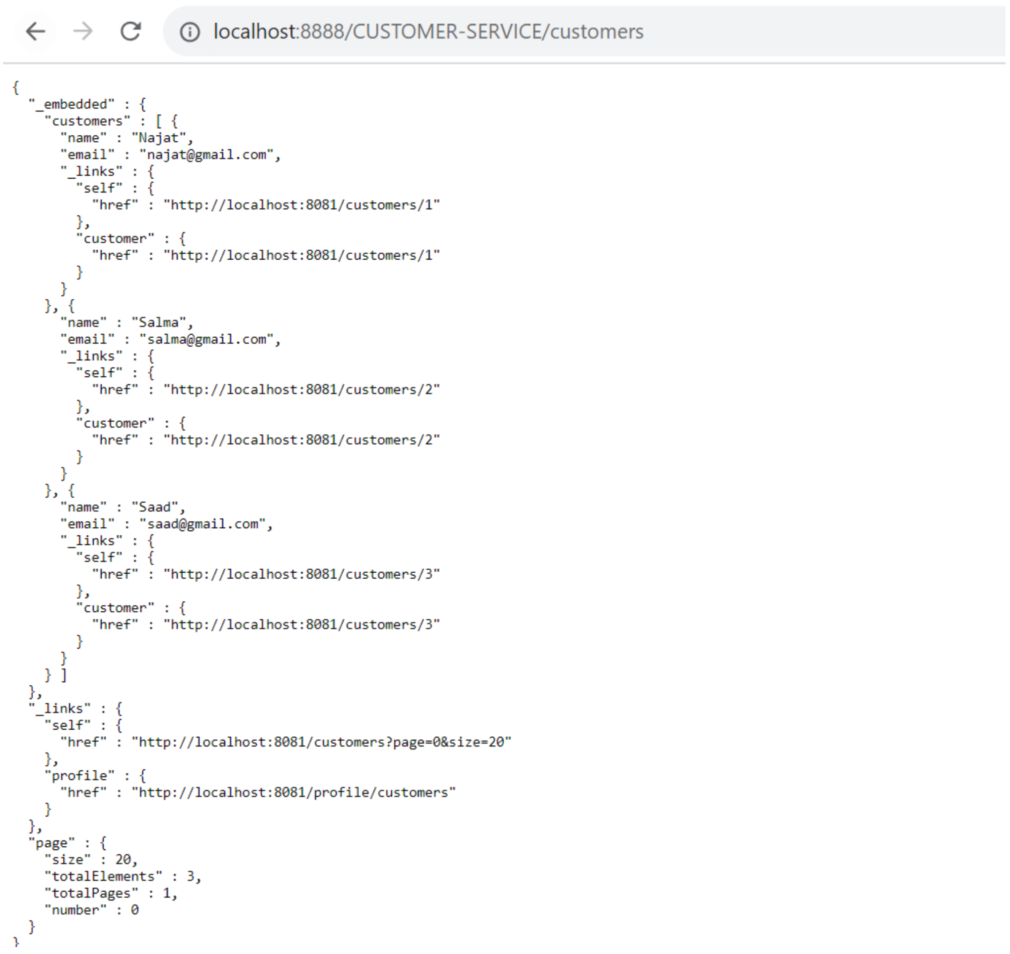

```
1 usage
@SpringBootApplication
public class GatewayApplication {

    no usages
    public static void main(String[] args) { SpringApplication.run(GatewayApplication.class, args); }

    no usages
    @Bean
    RouteLocator routeLocator(RouteLocatorBuilder builder){
        return builder.routes()
                .route( id: "r1",(r)->r.path("/customers/**").uri("lb://CUSTOMER-SERVICE"))
                .route( id: "r2",(r)->r.path("/products/**").uri("lb://PRODUCT-SERVICE"))
                .build();
    }
}
```

La configuration d’une manière dynamique on fait ça une fois pour tous les microservices :



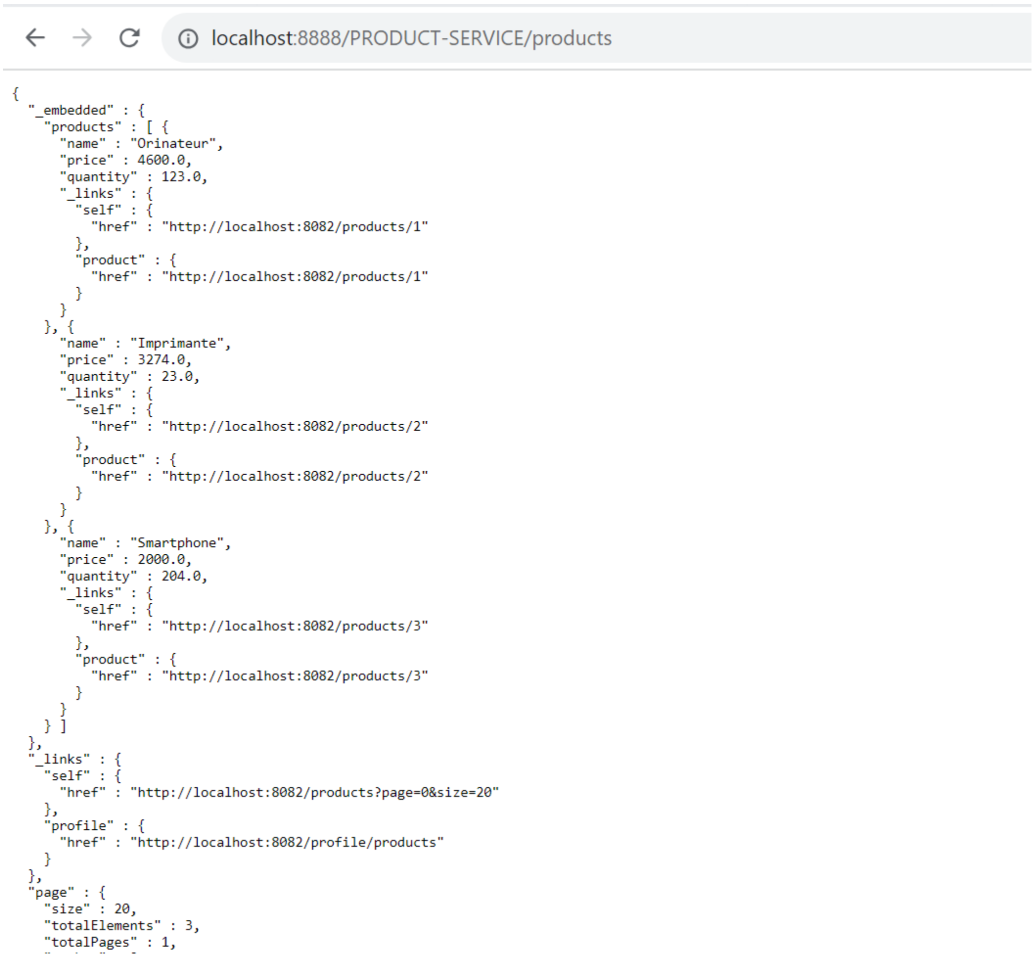
```
    @Bean
    DiscoveryClientRouteDefinitionLocator definitionLocator(ReactiveDiscoveryClient rdc, DiscoveryLocatorProperties properties){
        return new DiscoveryClientRouteDefinitionLocator(rdc,properties);
    }
}
```





localhost:8888/CUSTOMER-SERVICE/customers

```
{
  "_embedded" : {
    "customers" : [ {
      "name" : "Najat",
      "email" : "najat@gmail.com",
      "_links" : {
        "self" : {
          "href" : "http://localhost:8081/customers/1"
        },
        "customer" : {
          "href" : "http://localhost:8081/customers/1"
        }
      }
    }, {
      "name" : "Salma",
      "email" : "salma@gmail.com",
      "_links" : {
        "self" : {
          "href" : "http://localhost:8081/customers/2"
        },
        "customer" : {
          "href" : "http://localhost:8081/customers/2"
        }
      }
    }, {
      "name" : "Saad",
      "email" : "saad@gmail.com",
      "_links" : {
        "self" : {
          "href" : "http://localhost:8081/customers/3"
        },
        "customer" : {
          "href" : "http://localhost:8081/customers/3"
        }
      }
    } ]
  },
  "_links" : {
    "self" : {
      "href" : "http://localhost:8081/customers?page=0&size=20"
    },
    "profile" : {
      "href" : "http://localhost:8081/profile/customers"
    }
  },
  "page" : {
    "size" : 20,
    "totalElements" : 3,
    "totalPages" : 1,
    "number" : 0
  }
}
```

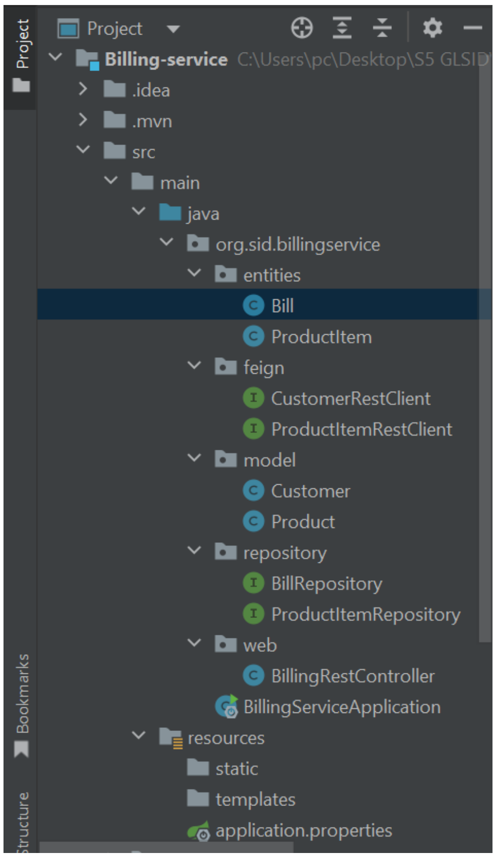

localhost:8888/PRODUCT-SERVICE/products

```
{
  "_embedded" : {
    "products" : [ {
      "name" : "Orinateur",
      "price" : 4600.0,
      "quantity" : 123.0,
      "_links" : {
        "self" : {
          "href" : "http://localhost:8082/products/1"
        },
        "product" : {
          "href" : "http://localhost:8082/products/1"
        }
      }
    }, {
      "name" : "Imprimante",
      "price" : 3274.0,
      "quantity" : 23.0,
      "_links" : {
        "self" : {
          "href" : "http://localhost:8082/products/2"
        },
        "product" : {
          "href" : "http://localhost:8082/products/2"
        }
      }
    }, {
      "name" : "Smartphone",
      "price" : 2000.0,
      "quantity" : 204.0,
      "_links" : {
        "self" : {
          "href" : "http://localhost:8082/products/3"
        },
        "product" : {
          "href" : "http://localhost:8082/products/3"
        }
      }
    } ]
  },
  "_links" : {
    "self" : {
      "href" : "http://localhost:8082/products?page=0&size=20"
    },
    "profile" : {
      "href" : "http://localhost:8082/profile/products"
    }
  },
  "page" : {
    "size" : 20,
    "totalElements" : 3,
    "totalPages" : 1,
```

Deuxième Partie

Billing Service avec Open Feign Rest Client

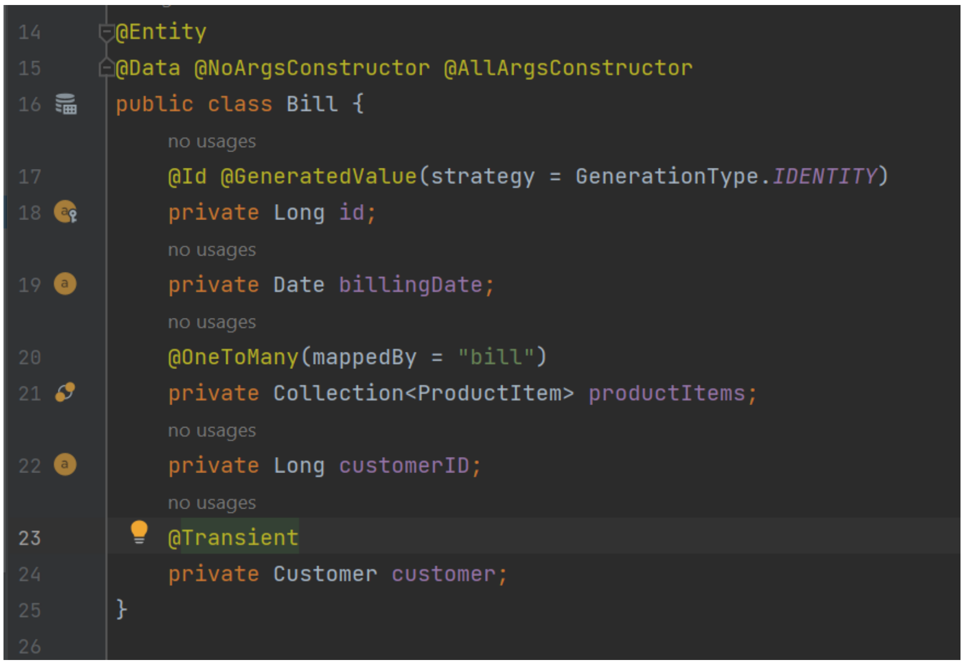
Structure de projet :



```
Project
Billing-service  C:\Users\pc\Desktop\S5 GLSID
  .idea
  .mvn
  src
    main
      java
        org.sid.billingservice
          entities
            Bill
            ProductItem
          feign
            CustomerRestClient
            ProductItemRestClient
          model
            Customer
            Product
          repository
            BillRepository
            ProductItemRepository
          web
            BillingRestController
          BillingServiceApplication
      resources
        static
        templates
        application.properties
```

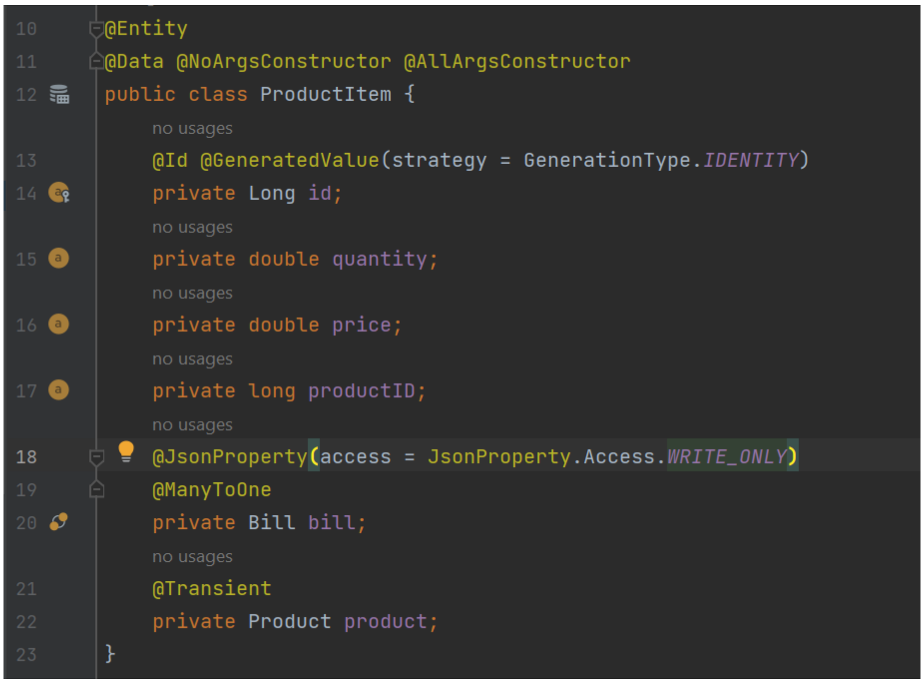
Entities :

Bill :



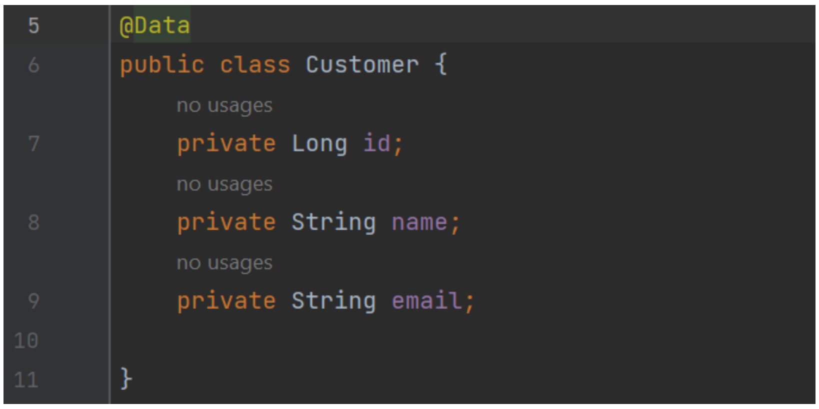
```java
@Entity
@Data @NoArgsConstructor @AllArgsConstructor
public class Bill {
    no usages
    @Id @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;
    no usages
    private Date billingDate;
    no usages
    @OneToMany(mappedBy = "bill")
    private Collection<ProductItem> productItems;
    no usages
    private Long customerID;
    no usages
    @Transient
    private Customer customer;
}
```

ProductItem :

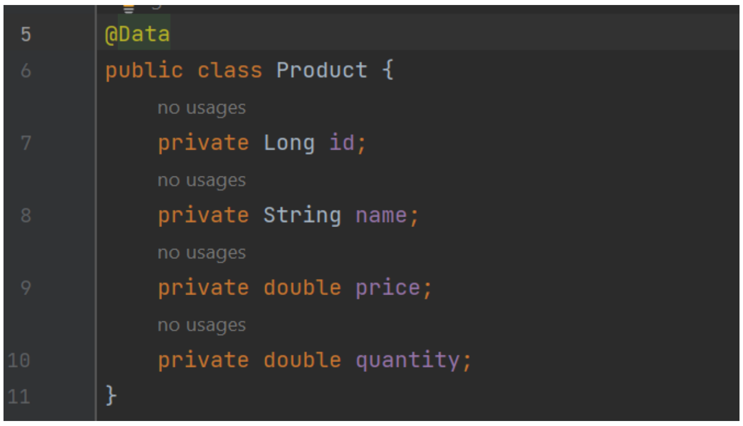

```java
@Entity
@Data @NoArgsConstructor @AllArgsConstructor
public class ProductItem {
    no usages
    @Id @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;
    no usages
    private double quantity;
    no usages
    private double price;
    no usages
    private long productID;
    no usages
    @JsonProperty(access = JsonProperty.Access.WRITE_ONLY)
    @ManyToOne
    private Bill bill;
    no usages
    @Transient
    private Product product;
}
```

Model :

Customer :

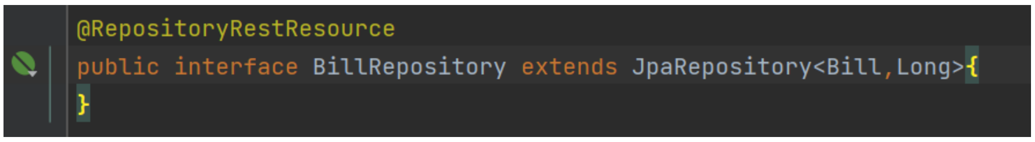

```java
@Data
public class Customer {
    no usages
    private Long id;
    no usages
    private String name;
    no usages
    private String email;

}
```

Product :


```java
@Data
public class Product {
    no usages
    private Long id;
    no usages
    private String name;
    no usages
    private double price;
    no usages
    private double quantity;
}
```

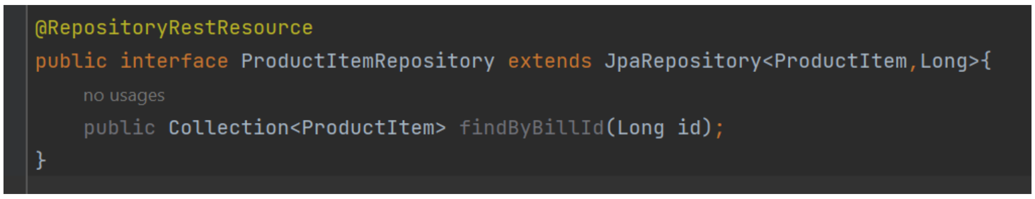
<u>Repository :</u>

BillRepository :



```java
@RepositoryRestResource
public interface BillRepository extends JpaRepository<Bill,Long>{
}
```

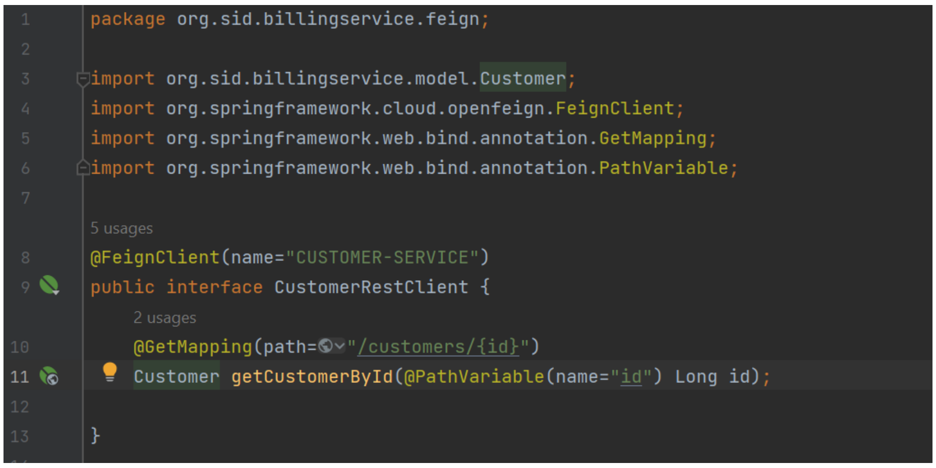
ProductItemRepository :



```java
@RepositoryRestResource
public interface ProductItemRepository extends JpaRepository<ProductItem,Long>{
    no usages
    public Collection<ProductItem> findByBillId(Long id);
}
```

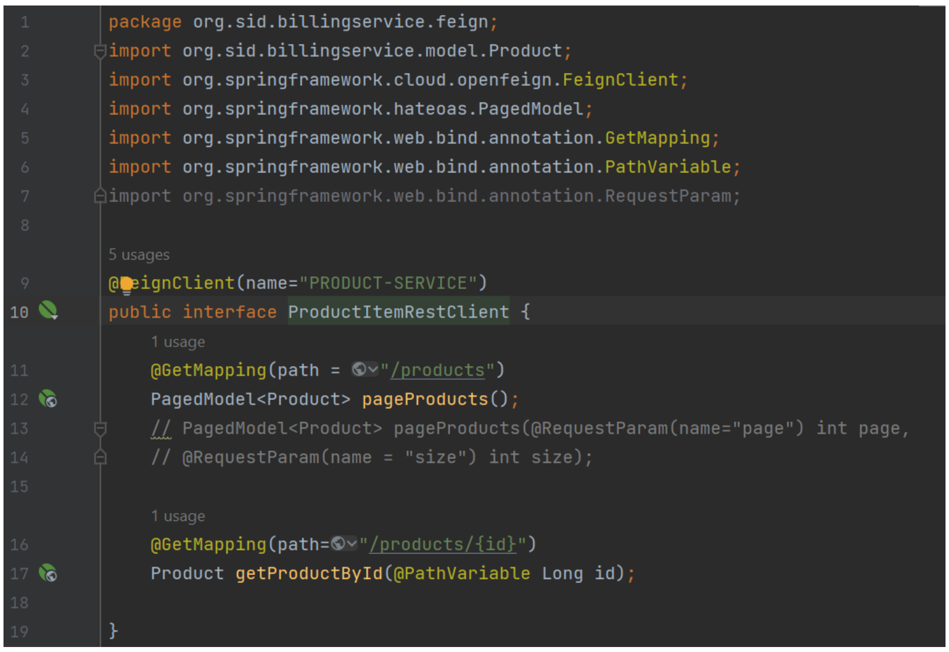
<u>Openfeign :</u>

CustomerRestClient :



```java
package org.sid.billingservice.feign;

import org.sid.billingservice.model.Customer;
import org.springframework.cloud.openfeign.FeignClient;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PathVariable;

5 usages
@FeignClient(name="CUSTOMER-SERVICE")
public interface CustomerRestClient {
    2 usages
    @GetMapping(path="/customers/{id}")
    Customer getCustomerById(@PathVariable(name="id") Long id);

}
```

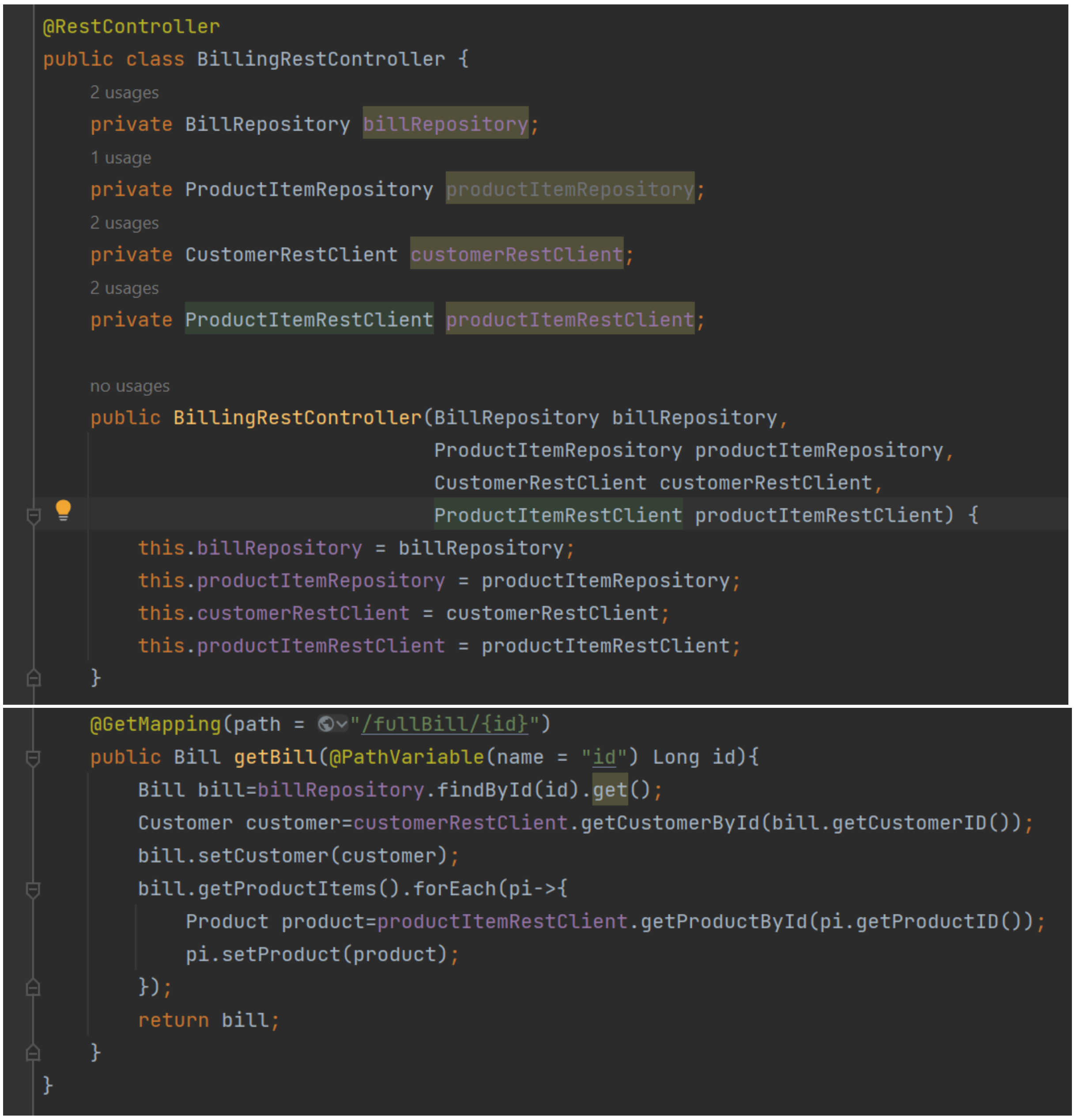
ProductItemRestClient :



```java
package org.sid.billingservice.feign;
import org.sid.billingservice.model.Product;
import org.springframework.cloud.openfeign.FeignClient;
import org.springframework.hateoas.PagedModel;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RequestParam;

5 usages
@FeignClient(name="PRODUCT-SERVICE")
public interface ProductItemRestClient {
    1 usage
    @GetMapping(path = "/products")
    PagedModel<Product> pageProducts();
    // PagedModel<Product> pageProducts(@RequestParam(name="page") int page,
    // @RequestParam(name = "size") int size);

    1 usage
    @GetMapping(path="/products/{id}")
    Product getProductById(@PathVariable Long id);

}
```

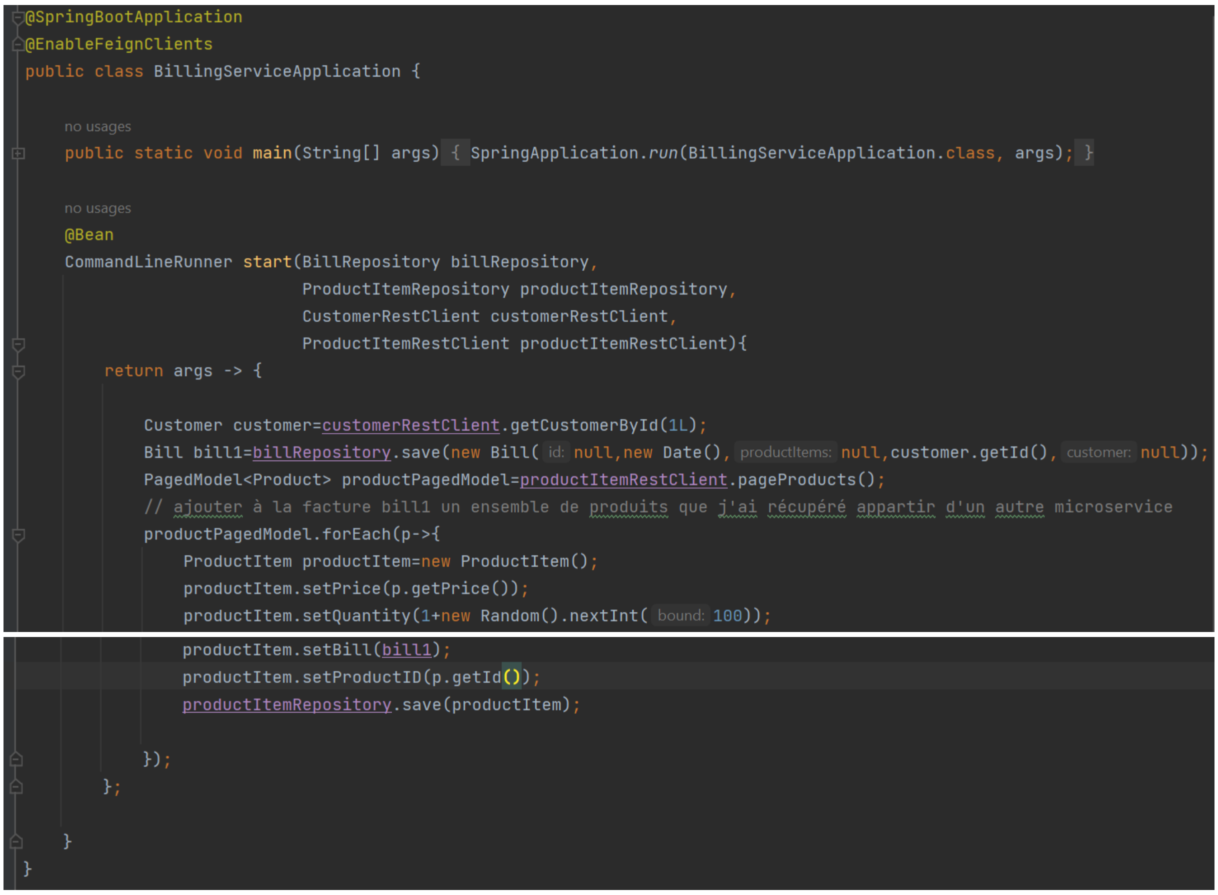
Web :

BillingRestController :



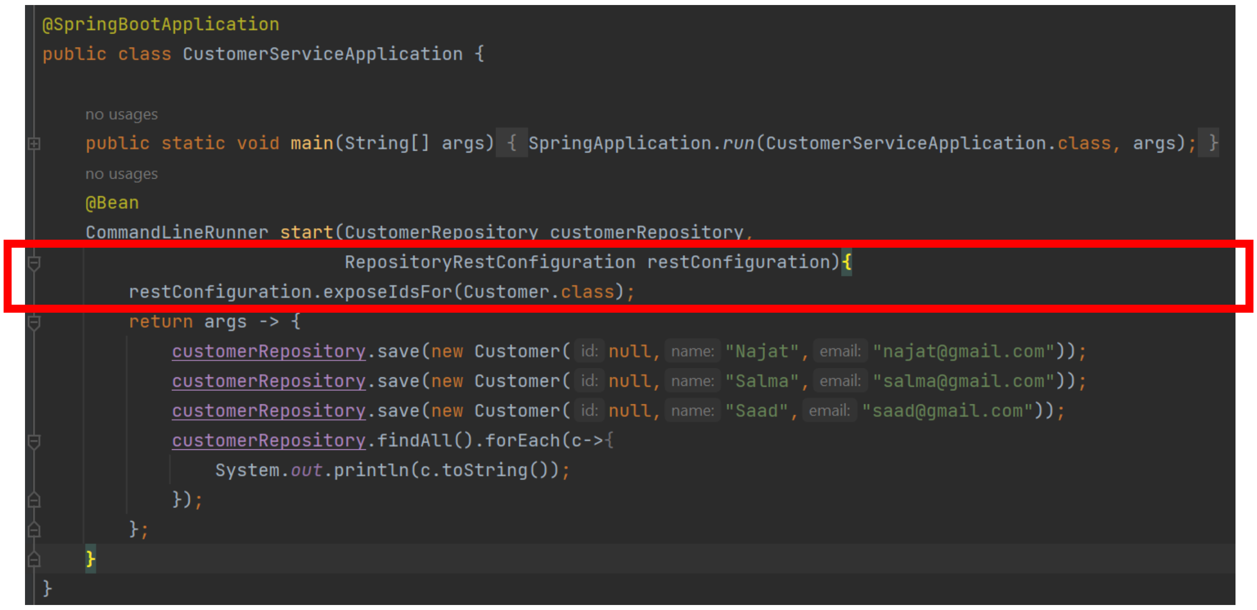
```
@RestController
public class BillingRestController {
    2 usages
    private BillRepository billRepository;
    1 usage
    private ProductItemRepository productItemRepository;
    2 usages
    private CustomerRestClient customerRestClient;
    2 usages
    private ProductItemRestClient productItemRestClient;

    no usages
    public BillingRestController(BillRepository billRepository,
                                 ProductItemRepository productItemRepository,
                                 CustomerRestClient customerRestClient,
                                 ProductItemRestClient productItemRestClient) {
        this.billRepository = billRepository;
        this.productItemRepository = productItemRepository;
        this.customerRestClient = customerRestClient;
        this.productItemRestClient = productItemRestClient;
    }
    @GetMapping(path = "/fullBill/{id}")
    public Bill getBill(@PathVariable(name = "id") Long id){
        Bill bill=billRepository.findById(id).get();
        Customer customer=customerRestClient.getCustomerById(bill.getCustomerID());
        bill.setCustomer(customer);
        bill.getProductItems().forEach(pi->{
            Product product=productItemRestClient.getProductById(pi.getProductID());
            pi.setProduct(product);
        });
        return bill;
    }
}
```

Application :

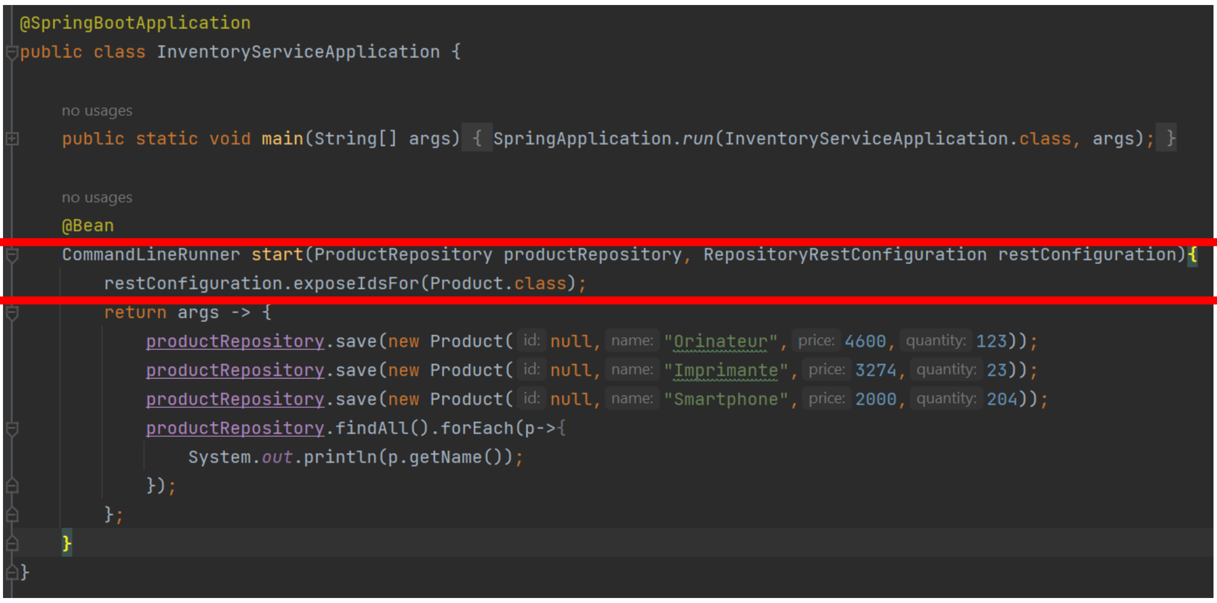


```java
@SpringBootApplication
@EnableFeignClients
public class BillingServiceApplication {

    no usages
    public static void main(String[] args) { SpringApplication.run(BillingServiceApplication.class, args); }

    no usages
    @Bean
    CommandLineRunner start(BillRepository billRepository,
                            ProductItemRepository productItemRepository,
                            CustomerRestClient customerRestClient,
                            ProductItemRestClient productItemRestClient){
        return args -> {

            Customer customer=customerRestClient.getCustomerById(1L);
            Bill bill1=billRepository.save(new Bill( id: null,new Date(), productItems: null,customer.getId(), customer: null));
            PagedModel<Product> productPagedModel=productItemRestClient.pageProducts();
            // ajouter à la facture bill1 un ensemble de produits que j'ai récupéré appartir d'un autre microservice
            productPagedModel.forEach(p->{
                ProductItem productItem=new ProductItem();
                productItem.setPrice(p.getPrice());
                productItem.setQuantity(1+new Random().nextInt( bound: 100));
                productItem.setBill(bill1);
                productItem.setProductID(p.getId());
                productItemRepository.save(productItem);

            });
        };

    }
}
```

Et On ajoute au niveau de Customer et Product service :

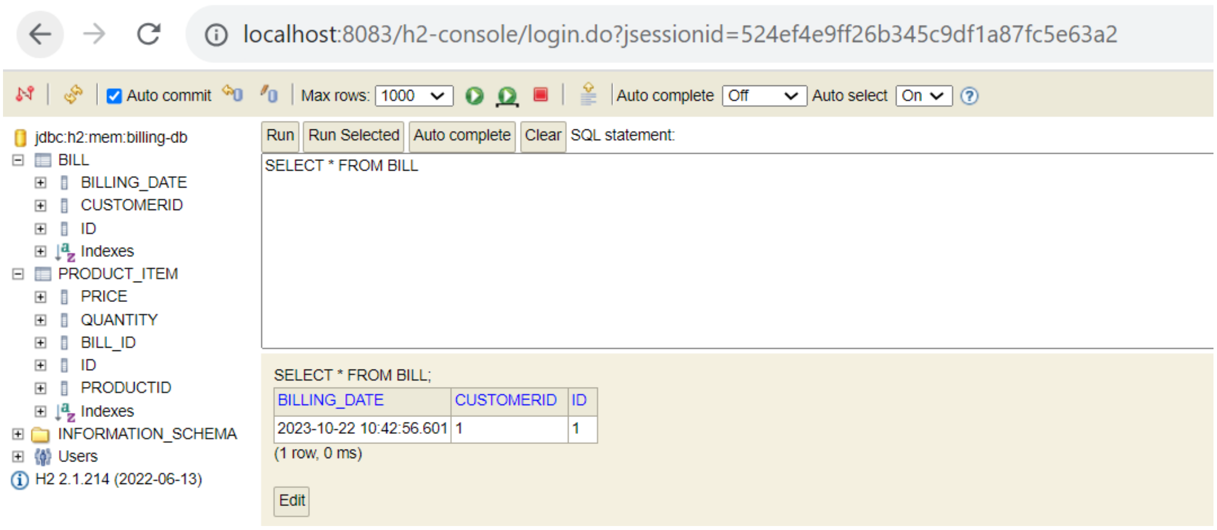


```java
@SpringBootApplication
public class CustomerServiceApplication {

    no usages
    public static void main(String[] args) { SpringApplication.run(CustomerServiceApplication.class, args); }
    no usages
    @Bean
    CommandLineRunner start(CustomerRepository customerRepository,
                            RepositoryRestConfiguration restConfiguration){
        restConfiguration.exposeIdsFor(Customer.class);
        return args -> {
            customerRepository.save(new Customer( id: null, name: "Najat", email: "najat@gmail.com"));
            customerRepository.save(new Customer( id: null, name: "Salma", email: "salma@gmail.com"));
            customerRepository.save(new Customer( id: null, name: "Saad", email: "saad@gmail.com"));
            customerRepository.findAll().forEach(c->{
                System.out.println(c.toString());
            });
        };
    }
}
```



```java
@SpringBootApplication
public class InventoryServiceApplication {

    no usages
    public static void main(String[] args) { SpringApplication.run(InventoryServiceApplication.class, args); }

    no usages
    @Bean
    CommandLineRunner start(ProductRepository productRepository, RepositoryRestConfiguration restConfiguration){
        restConfiguration.exposeIdsFor(Product.class);
        return args -> {
            productRepository.save(new Product( id: null, name: "Orinateur", price: 4600, quantity: 123));
            productRepository.save(new Product( id: null, name: "Imprimante", price: 3274, quantity: 23));
            productRepository.save(new Product( id: null, name: "Smartphone", price: 2000, quantity: 204));
            productRepository.findAll().forEach(p->{
                System.out.println(p.getName());
            });
        };
    }
}
```

La base de données :



localhost:8083/h2-console/login.do?jsessionid=524ef4e9ff26b345c9df1a87fc5e63a2

Auto commit ✓ Max rows: 1000 Auto complete: Off Auto select: On

- jdbc:h2:mem:billing-db
  - BILL: BILLING_DATE, CUSTOMERID, ID, Indexes
  - PRODUCT_ITEM: PRICE, QUANTITY, BILL_ID, ID, PRODUCTID, Indexes
  - INFORMATION_SCHEMA
  - Users
- H2 2.1.214 (2022-06-13)

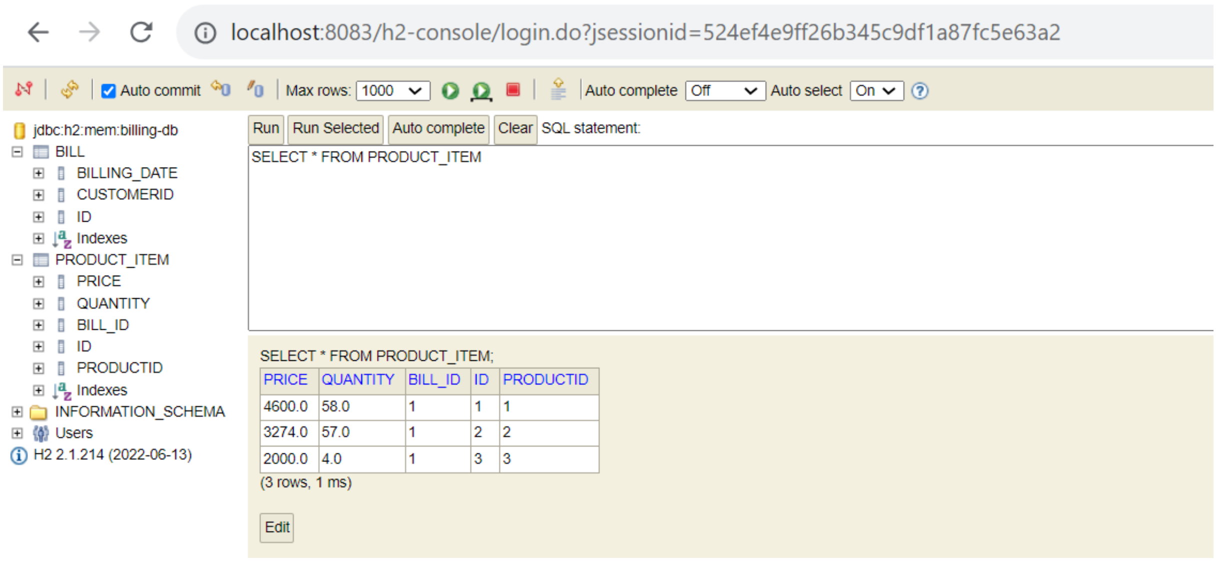
SQL statement:

```sql
SELECT * FROM BILL
```

SELECT * FROM BILL;

BILLING_DATE	CUSTOMERID	ID
2023-10-22 10:42:56.601	1	1

(1 row, 0 ms)



localhost:8083/h2-console/login.do?jsessionid=524ef4e9ff26b345c9df1a87fc5e63a2

SQL statement:

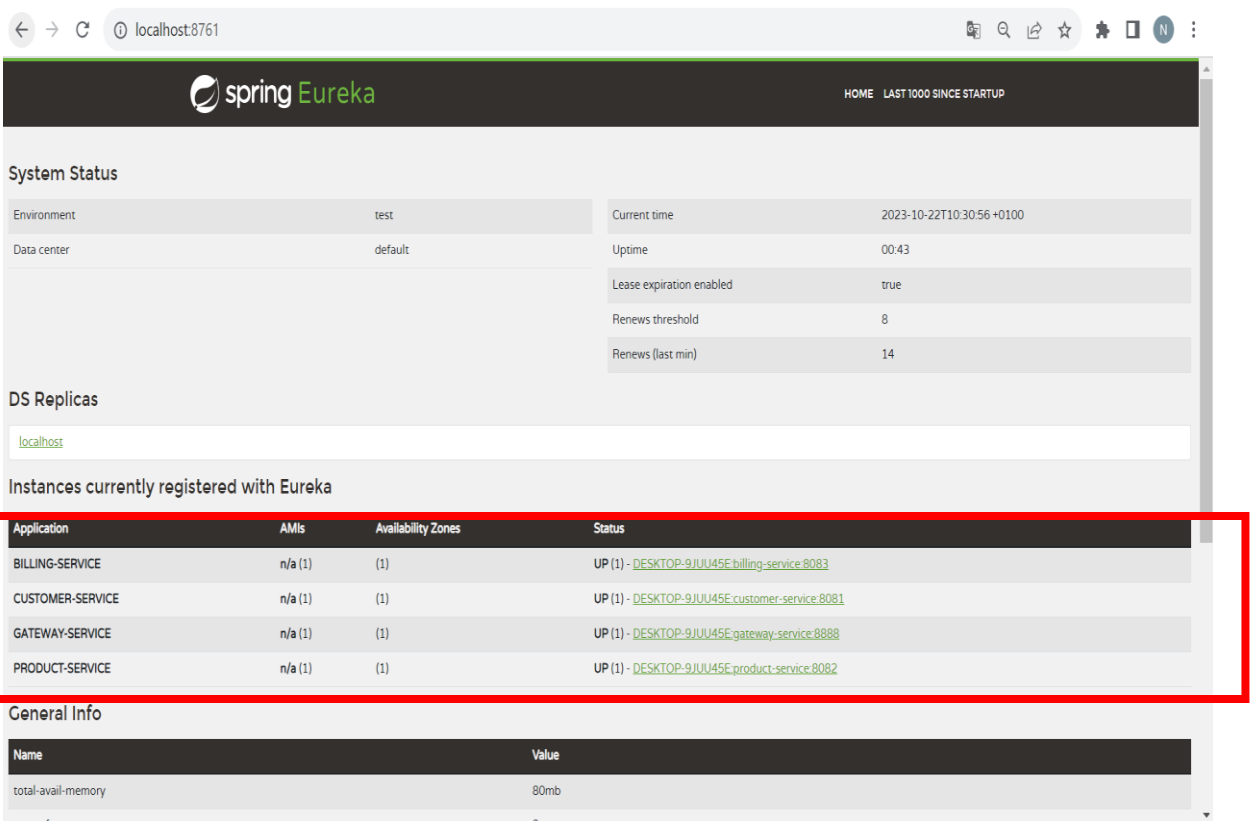
```sql
SELECT * FROM PRODUCT_ITEM
```

SELECT * FROM PRODUCT_ITEM;

PRICE	QUANTITY	BILL_ID	ID	PRODUCTID
4600.0	58.0	1	1	1
3274.0	57.0	1	2	2
2000.0	4.0	1	3	3

(3 rows, 1 ms)

Eureka :



localhost:8761

spring Eureka

HOME LAST 1000 SINCE STARTUP

System Status

Environment	test
Data center	default

Current time	2023-10-22T10:30:56 +0100
Uptime	00:43
Lease expiration enabled	true
Renews threshold	8
Renews (last min)	14

DS Replicas

localhost

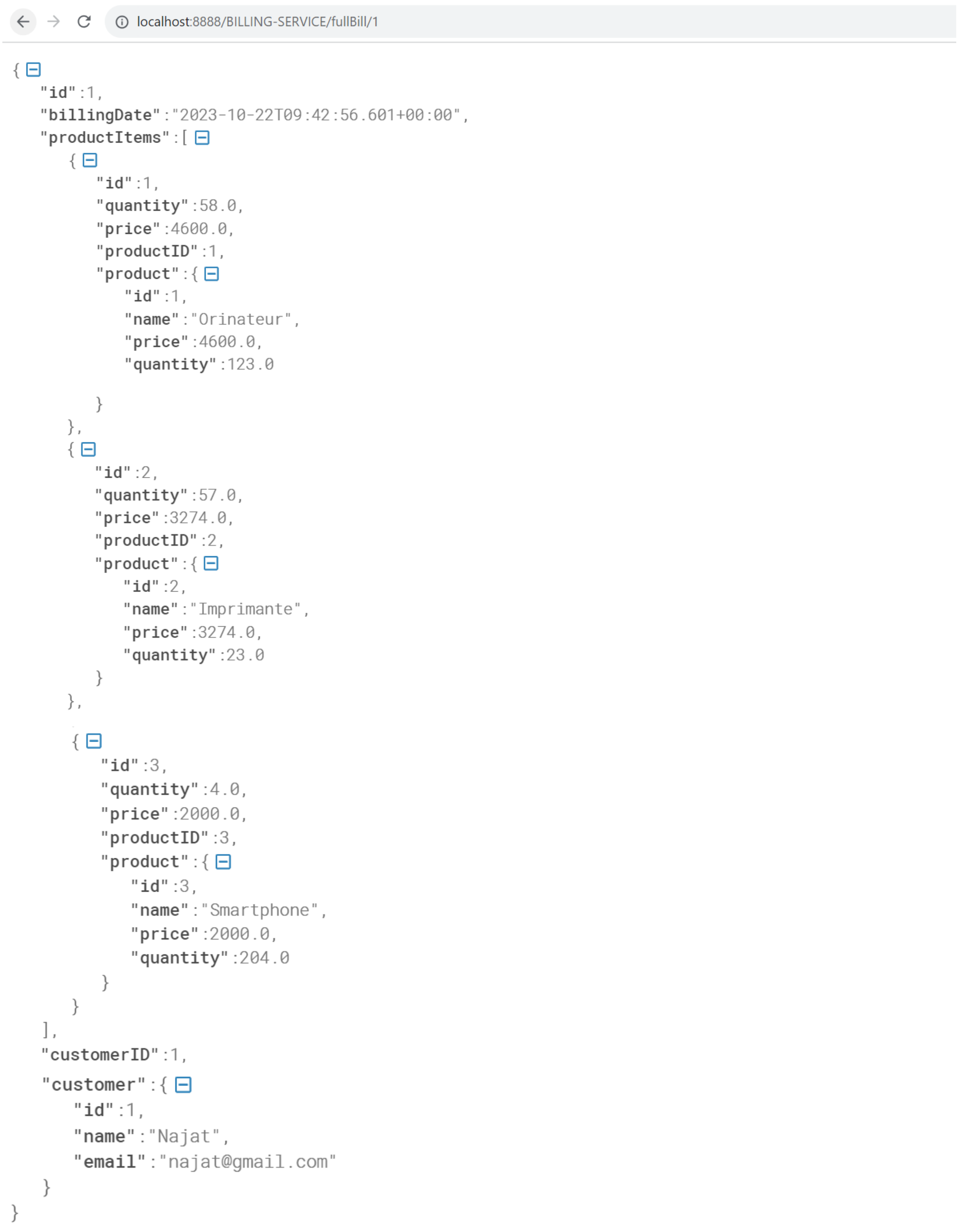
Instances currently registered with Eureka

Application	AMIs	Availability Zones	Status
BILLING-SERVICE	n/a (1)	(1)	UP (1) - DESKTOP-9JUU45E:billing-service:8083
CUSTOMER-SERVICE	n/a (1)	(1)	UP (1) - DESKTOP-9JUU45E:customer-service:8081
GATEWAY-SERVICE	n/a (1)	(1)	UP (1) - DESKTOP-9JUU45E:gateway-service:8888
PRODUCT-SERVICE	n/a (1)	(1)	UP (1) - DESKTOP-9JUU45E:product-service:8082

General Info

Name	Value
total-avail-memory	80mb

Résultats sur la console :



localhost:8888/BILLING-SERVICE/fullBill/1

```
{
   "id":1,
   "billingDate":"2023-10-22T09:42:56.601+00:00",
   "productItems":[
      {
         "id":1,
         "quantity":58.0,
         "price":4600.0,
         "productID":1,
         "product":{
            "id":1,
            "name":"Orinateur",
            "price":4600.0,
            "quantity":123.0

         }
      },
      {
         "id":2,
         "quantity":57.0,
         "price":3274.0,
         "productID":2,
         "product":{
            "id":2,
            "name":"Imprimante",
            "price":3274.0,
            "quantity":23.0
         }
      },

      {
         "id":3,
         "quantity":4.0,
         "price":2000.0,
         "productID":3,
         "product":{
            "id":3,
            "name":"Smartphone",
            "price":2000.0,
            "quantity":204.0
         }
      }
   ],
   "customerID":1,
   "customer":{
      "id":1,
      "name":"Najat",
      "email":"najat@gmail.com"
   }
}
```