

# examen-microservices-distributed-systems

---

**ENSET-M    II-BDCC 2    EL AAMIRI Essadeq**

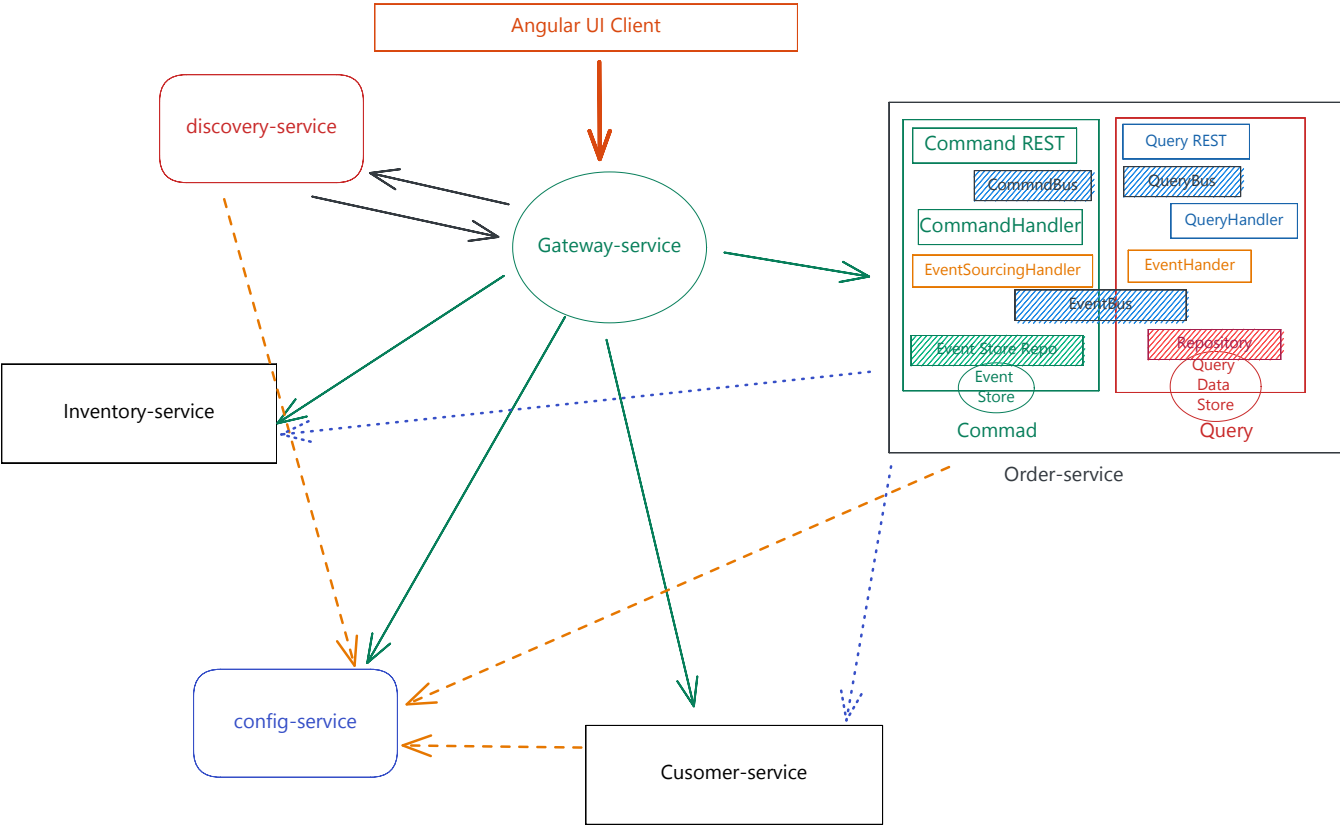
---

Développer les fonctionnalités qui vous semblent les plus importantes pour ce projet et rendre un rapport et le code source de l'application répondant aux questions suivantes :

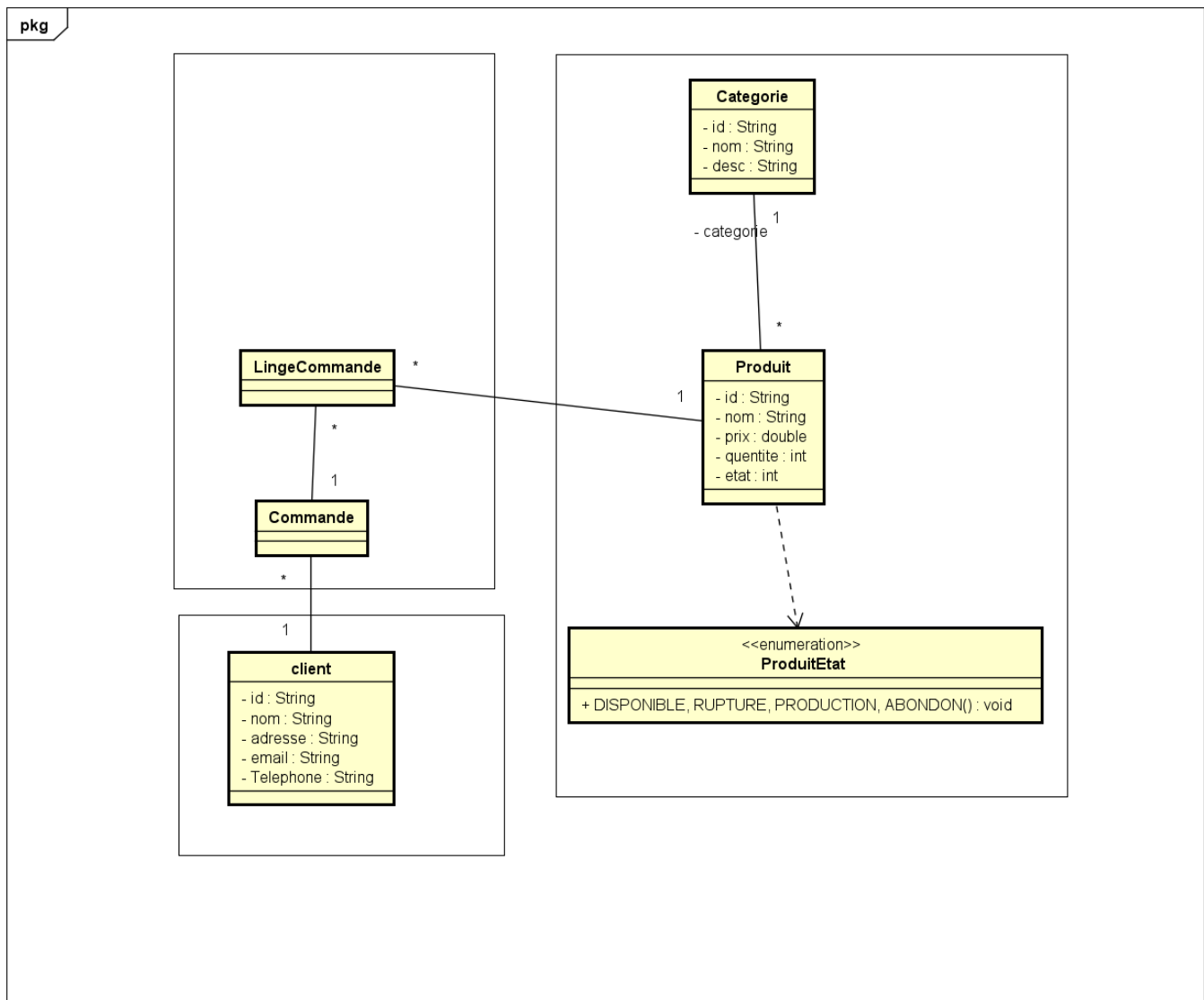
- 1. [Command side](#)
  - 1.1. [Commands](#)
  - 1.2. [Events](#)
  - 1.3. [Dtos](#)
  - 1.4. [Aggregate](#)
  - 1.5. [Controller](#)
  - 1.6. [Tests](#)
- 2. [Query side](#)
  - 2.1. [Queries](#)
  - 2.2. [Entities](#)
  - 2.3. [Repositories](#)
  - 2.4. [Eventhandler Service](#)
  - 2.5. [Queryhandler Service](#)
  - 2.6. [Controllers](#)
  - 2.7. [Tests](#)
- 3. [Command side](#)
  - 3.1. [Commands](#)
  - 3.2. [Events](#)
  - 3.3. [Aggregates](#)
  - 3.4. [Controllers](#)
  - 3.5. [tests](#)
- 4. [Query side](#)
  - 4.1. [Queries](#)
  - 4.2. [Entities](#)
  - 4.3. [Repositories](#)
  - 4.4. [EventHandler Services](#)
  - 4.5. [QueryHandler Services](#)
  - 4.6. [Query Controllers](#)
  - 4.7. [Tests](#)
- 5. [Discovery service](#)
- 6. [Gateway service](#)

## Établir une architecture technique du projet

---



# Établir un diagramme de classe global du projet



## Déployer le serveur AXON Server ou KAFKA Broker

- Le serveur est démarré en tant que Docker container

Showing 8 items							
	NAME	IMAGE	STATUS	PORT(S)	STARTED	ACTIONS	
<input type="checkbox"/>	axonservice 15ca6f41377c	axoniq/axonservice:4.6.2-jdk-11-dev	Running	8024,8...	2 hours ago		
<input type="checkbox"/>	mykeycloak						

- Dependences utilisées:

```

<dependency>
  <groupId>org.axonframework</groupId>
  <artifactId>axon-spring-boot-starter</artifactId>
  <version>4.6.1</version>
  <!--      <exclusions>-->
  <!--      <exclusion>-->
  <!--          <groupId>org.axonframework</groupId>-->
  <!--          <artifactId>axon-server-connector</artifactId>-->
  <!--      </exclusion>-->
  
```

```
<!--           </exclusions>-->
</dependency>
```

# Développer le micro-service Customer-Service

---

## 1. Command side

### 1.1. Commands

- CreateCustomerCommand

```
package me.elaamiri.commands.customerCommands;

import lombok.Getter;
import me.elaamiri.commands.BaseCommand;

public class CreateCustomerCommand extends BaseCommand<String> {

    @Getter
    private String nom;
    @Getter
    private String adresse;
    @Getter
    private String email;
    @Getter
    private String telephone;

    public CreateCustomerCommand(String id, String nom, String adresse, String
email, String telephone) {
        super(id);
        this.nom = nom;
        this.adresse = adresse;
        this.email = email;
        this.telephone = telephone;
    }
}
```

- UpdateCustomerCommand

```
package me.elaamiri.commands.customerCommands;

import lombok.Getter;
import me.elaamiri.commands.BaseCommand;

public class UpdateCustomerCommand extends BaseCommand<String> {
    @Getter
    private String nom;
```

```
    @Getter
    private String adresse;
    @Getter
    private String email;
    @Getter
    private String telephone;

    public UpdateCustomerCommand(String id, String nom, String adresse, String
email, String telephone) {
        super(id);
        this.nom = nom;
        this.adresse = adresse;
        this.email = email;
        this.telephone = telephone;
    }
}
```

- DeleteCustomerCommand

```
package me.elaamiri.commands.customerCommands;

import me.elaamiri.commands.BaseCommand;

public class DeleteCustomerCommand extends BaseCommand<String> {
    public DeleteCustomerCommand(String id) {
        super(id);
    }
}
```

## 1.2. Events

- CustomerCreatedEvent

```
package me.elaamiri.events.customerEvents;

import lombok.Getter;
import me.elaamiri.events.BaseEvent;

public class CustomerCreatedEvent extends BaseEvent<String> {
    @Getter
    private String nom;
    @Getter
    private String adresse;
    @Getter
    private String email;
    @Getter
    private String telephone;
}
```

```
    public CustomerCreatedEvent(String id, String nom, String adresse, String
email, String telephone) {
        super(id);
        this.nom = nom;
        this.adresse = adresse;
        this.email = email;
        this.telephone = telephone;
    }
}
```

- CustomerUpdatedEvent

```
package me.elaamiri.events.customerEvents;

import lombok.Getter;
import me.elaamiri.events.BaseEvent;

public class CustomerUpdatedEvent extends BaseEvent<String> {
    @Getter
    private String nom;
    @Getter
    private String adresse;
    @Getter
    private String email;
    @Getter
    private String telephone;

    public CustomerUpdatedEvent(String id, String nom, String adresse, String
email, String telephone) {
        super(id);
        this.nom = nom;
        this.adresse = adresse;
        this.email = email;
        this.telephone = telephone;
    }
}
```

- CustomerDeletedEvent

```
package me.elaamiri.events.customerEvents;

import me.elaamiri.events.BaseEvent;

public class CustomerDeletedEvent extends BaseEvent<String> {
    public CustomerDeletedEvent(String id) {
        super(id);
    }
}
```

```
}  
}
```

### 1.3. Dtos

- CreateCustomerRequestDTO

```
package me.elaamiri.dtos.customerDtos;  
  
import lombok.*;  
  
@Data  
@AllArgsConstructor @NoArgsConstructor @Builder  
public class CreateCustomerRequestDTO {  
  
    private String nom;  
  
    private String adresse;  
  
    private String email;  
  
    private String telephone;  
}
```

- UpdateCustomerRequestDTO

```
package me.elaamiri.dtos.customerDtos;  
  
@Data  
@AllArgsConstructor  
@NoArgsConstructor  
@Builder  
public class UpdateCustomerRequestDTO {  
    private String nom;  
  
    private String adresse;  
  
    private String email;  
  
    private String telephone;  
}
```

### 1.4. Aggregate

- CustomerAggregate

```
package me.elaamiri.customerservice.command.aggregates;

@Aggregate
@Slf4j
public class CustomerAggregate {

    @AggregateIdentifier
    @Getter
    private String id;
    @Getter
    private String nom;
    @Getter
    private String adresse;
    @Getter
    private String email;
    @Getter
    private String telephone;

    public CustomerAggregate() {
    }

    @CommandHandler
    public CustomerAggregate(CreateCustomerCommand command) {
        // validations

        AggregateLifecycle.apply(new CustomerCreatedEvent(
            command.getId(),
            command.getNom(),
            command.getAdresse(),
            command.getEmail(),
            command.getTelephone()
        ));
    }

    @EventSourcingHandler
    public void on(CustomerCreatedEvent event){
        this.id = event.getId();
        this.nom = event.getNom();
        this.adresse = event.getAdresse();
        this.email = event.getEmail();
        this.telephone = event.getTelephone();
    }

    @CommandHandler
    public void handle(UpdateCustomerCommand command){
        // validations
        if(command.getId() == null || command.getId().isBlank()) throw new
EntryNotFoundException("Customer Not found");
        AggregateLifecycle.apply(new CustomerUpdatedEvent(
            command.getId(),
            command.getNom(),
```



```

        command.getAdresse(),
        command.getEmail(),
        command.getTelephone()
    ));
}

@EventSourcingHandler
public void on(CustomerUpdatedEvent event){
    this.id = event.getId();
    this.nom = event.getNom();
    this.adresse = event.getAdresse();
    this.email = event.getEmail();
    this.telephone = event.getTelephone();
}

@CommandHandler
public void handle(DeleteCustomerCommand command){
    if(command.getId() == null || command.getId().isBlank()) throw new
EntryNotFoundException("Customer Not found");
    AggregateLifecycle.apply(new CustomerDeletedEvent(
        command.getId()
    ));
}

@EventSourcingHandler
public void on(CustomerDeletedEvent event){
    log.warn("Deleting Customer: "+ event.getId());
}
}

```

## 1.5. Controller

- CustomerCommandController

```

package me.elaamiri.customerservice.command.controllers;

@RestController
@AllArgsConstructor
@RequestMapping("/commands/customer")
public class CustomerCommandController {
    private CommandGateway commandGateway;

    @PostMapping("/create")
    public CompletableFuture<String> create(@RequestBody CreateCustomerRequestDTO
customerRequestDTO){
        CompletableFuture<String> response = commandGateway.send(
            new CreateCustomerCommand(
                UUID.randomUUID().toString(),
                customerRequestDTO.getNom(),

```

```
        customerRequestDTO.getAdresse(),
        customerRequestDTO.getEmail(),
        customerRequestDTO.getTelephone()
    )
);

return response;
}

@PutMapping("/update/{id}")
public CompletableFuture<String> update(@RequestBody UpdateCustomerRequestDTO
customerRequestDTO, @PathVariable String id){
    CompletableFuture<String> response = commandGateway.send(
        new UpdateCustomerCommand(
            id,
            customerRequestDTO.getNom(),
            customerRequestDTO.getAdresse(),
            customerRequestDTO.getEmail(),
            customerRequestDTO.getTelephone()
        )
    );

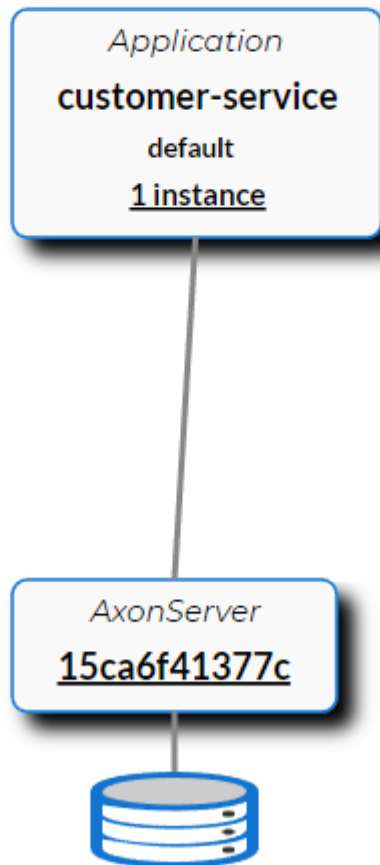
    return response;
}

>DeleteMapping("/delete/{id}")
public CompletableFuture<String> delete( @PathVariable String id){
    CompletableFuture<String> response = commandGateway.send(
        new DeleteCustomerCommand(
            id
        )
    );

    return response;
}
}
```

## 1.6. Tests

- Our service in Axon



- Create Customer

http://localhost:8081/commands/customer/create


POST ▼ http://localhost:8081/commands/customer/create

Params Authorization Headers (9) **Body** ● Pre-request Script Tests Settings

☐ none ☐ form-data ☐ x-www-form-urlencoded ☒ raw ☐ binary ☐ GraphQL **JSON** ▼

```
1 {  
2   ... "nom": "Essadeq",  
3   ... "adresse": "B-152-Medoa",  
4   ... "email": "essadeq@gmail.com",  
5   ... "telephone": "0700763650"  
6 }
```

Body Cookies Headers (5) Test Results

Pretty Raw Preview Visualize Text ▼ 

```
1 3eab14f1-2f07-4070-a56a-e1da4daaf670
```

- Update Customer

http://localhost:8081/commands/customer/update/3eab14f1-2f07-4070-a56a-e1da4daaf670


PUT ▼ http://localhost:8081/commands/customer/update/3eab14f1-2f07-4070-a56a-e1da4daaf670

Params Authorization Headers (9) **Body** ● Pre-request Script Tests Settings

☐ none ☐ form-data ☐ x-www-form-urlencoded ☒ raw ☐ binary ☐ GraphQL **JSON** ▼

```
1 {  
2   ... "nom": "Ahmed",  
3   ... "adresse": "B-152-Medoa",  
4   ... "email": "ahmed@gmail.com",  
5   ... "telephone": "0700763650"  
6 }
```

Body Cookies Headers (4) Test Results

Pretty Raw Preview Visualize Text ▼ 

```
1
```

- Delete Customer

POST http://localhost:8081

PUT http://localhost:8081/c

DEL http://localhost:8081/c

+

...

No Environme

http://localhost:8081/commands/customer/delete/3eab14f1-2f07-4070-a56a-e1da4daaf670

Save

DELETE http://localhost:8081/commands/customer/delete/3eab14f1-2f07-4070-a56a-e1da4daaf670

Params Authorization Headers (9) Body Pre-request Script Tests Settings

Query Params

KEY	VALUE	DESCRIPTION
Key	Value	Description


Body Cookies Headers (4) Test Results

Status: 200 OK Time: 93 ms Size: 123 B

Pretty Raw Preview Visualize Text

1

- All 3 events

 Axon Server

Search: ☒ Events ☐ Snapshots

Enter your query here

About the query language

token	eventIdIdentifier	aggregateIdentifier	aggregateSequ...	aggregateType	payloadType	payloadRevision
54	ac01360d-446a-462c-81...	3eab14f1-2f07-4070-a56...	2	CustomerAggregate	me.elaamiri.events.customerEvents.CustomerDelete...	
53	95d3bba5-1d6d-4c13-b8...	3eab14f1-2f07-4070-a56...	1	CustomerAggregate	me.elaamiri.events.customerEvents.CustomerUpdate...	
52	67b94136-6333-42ab-9b...	3eab14f1-2f07-4070-a56...	0	CustomerAggregate	me.elaamiri.events.customerEvents.CustomerCreate...	

## 2. Query side

### 2.1. Queries

- GetAllCustomersQuery

```
package me.elaamiri.queries.customerQueries;

public class GetAllCustomersQuery {
}
```

- GetCustomerByldQuery

```
package me.elaamiri.queries.customerQueries;
```

```
import lombok.Getter;

public class GetCustomerByIdQuery {

    @Getter
    private String id;

    public GetCustomerByIdQuery(String id) {
        this.id = id;
    }
}
```

## 2.2. Entities

- Customer

```
package me.elaamiri.customerservice.query.entities;

@Entity
@Data
@AllArgsConstructor @NoArgsConstructor @Builder
public class Customer {

    @Id
    private String id;

    private String nom;

    private String adresse;

    private String email;

    private String telephone;
}
```

## 2.3. Repositories

- CustomerRepository

```
package me.elaamiri.customerservice.query.repositories;

public interface CustomerRepository extends JpaRepository<Customer, String> {
}
```

## 2.4. Eventhandler Service

- CustomerEventHandlerService

```
package me.elaamiri.customerservice.query.services;

@Service
@Slf4j
@AllArgsConstructor
public class CustomerEventHandlerService {
    private CustomerRepository customerRepository;

    @EventHandler
    public void on(CustomerCreatedEvent event){
        Customer customer = Customer.builder()
            .id(event.getId())
            .adresse(event.getAdresse())
            .email(event.getEmail())
            .telephone(event.getTelephone())
            .nom(event.getNom())
            .build();
        customerRepository.save(customer);
    }

    @EventHandler
    public void on(CustomerUpdatedEvent event){
        Customer customer = customerRepository.findById(event.getId())
            .orElseThrow(() -> new EntryNotFoundException("Customer Not
Found"));
        customer.setNom(event.getNom());
        customer.setAdresse(event.getAdresse());
        customer.setEmail(event.getEmail());
        customer.setTelephone(event.getTelephone());
        // customer.setId(event.getId());
        customerRepository.save(customer);
    }

    @EventHandler
    public void on(CustomerDeletedEvent event){
        Customer customer = customerRepository.findById(event.getId())
            .orElseThrow(() -> new EntryNotFoundException("Customer Not
Found"));
        customerRepository.delete(customer);
    }
}
```

## 2.5. Queryhandler Service

- CustomerQueryHandler

```
package me.elaamiri.customerservice.query.services;

@Service
@Slf4j
@AllArgsConstructor
public class CustomerQueryHandler {
    private CustomerRepository customerRepository;

    @QueryHandler
    public List<Customer> handle(GetAllCustomersQuery query){
        return customerRepository.findAll();
    }

    @QueryHandler
    public Customer handle(GetCustomerIdQuery query){
        return customerRepository.findById(query.getId())
            .orElseThrow(() -> new EntryNotFoundException("Customer Not
Found"));
    }
}
```

## 2.6. Controllers

- CustomerQueryController

```
package me.elaamiri.customerservice.query.controllers;

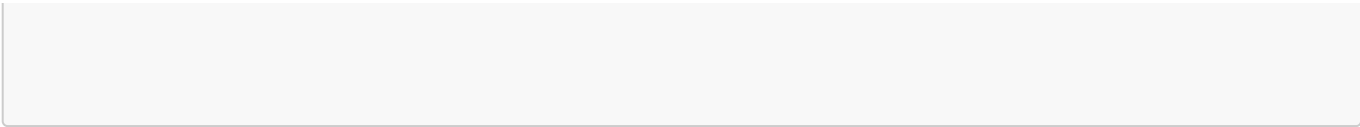
@RestController
@AllArgsConstructor
@RequestMapping("/query/customers")
public class CustomerQueryController {

    private QueryGateway queryGateway;

    @GetMapping("")
    public List<Customer> getAllCustomers(){
        List<Customer> customers = queryGateway
            .query(new GetAllCustomersQuery(),
ResponseTypes.multipleInstancesOf(Customer.class)).join();
        return customers;
    }

    @GetMapping("/{id}")
    public Customer getCustomerId(@PathVariable String id){
        Customer customer = queryGateway
            .query(new GetCustomerIdQuery(id),
ResponseTypes.instanceOf(Customer.class)).join();
        return customer;
    }
}
```





2.7. Tests

- Get all customers

POST http://localhost:8081/PUT http://localhost:8081/cDEL http://localhost:8081/cGET http://localhost:8081/qGET http://localhost:8081/q+...

http://localhost:8081/query/customers

GEThttp://localhost:8081/query/customers

ParamsAuthorizationHeaders (9)BodyPre-request ScriptTestsSettings

Query Params

KEY	VALUE	DESCRIPTION
Key	Value	Description

BodyCookiesHeaders (5)Test Results

PrettyRawPreviewVisualizeJSON

```
1  [
2    {
3      "id": "5d588872-65ed-4f10-a32f-2c2ddaa52823",
4      "nom": "Salama",
5      "adresse": "B 152 Medoa",
6      "email": "s1m@gmail.com",
7      "telephone": "0700763650"
8    },
9    {
10     "id": "b73284b3-ae55-4320-8f8a-552ed2af3b5a",
11     "nom": "Essadeq",
12     "adresse": "B 152 Medoa",
13     "email": "essadeq@gmail.com",
14     "telephone": "0700763650"
15   }
16 ]
```

- Get One by ID

POST http://localhost:8081/PUT http://localhost:8081/cDEL http://localhost:8081/cGET http://localhost:8081/qGET http://localhost:8081/q+...No Err

http://localhost:8081/query/customers/b73284b3-ae55-4320-8f8a-552ed2af3b5a

GEThttp://localhost:8081/query/customers/b73284b3-ae55-4320-8f8a-552ed2af3b5a

ParamsAuthorizationHeaders (9)BodyPre-request ScriptTestsSettings

Query Params

KEY	VALUE	DESCRIPTION
Key	Value	Description

BodyCookiesHeaders (5)Test Results

PrettyRawPreviewVisualizeJSON

```
1  {
2    "id": "b73284b3-ae55-4320-8f8a-552ed2af3b5a",
3    "nom": "Essadeq",
4    "adresse": "B 152 Medoa",
5    "email": "essadeq@gmail.com",
6    "telephone": "0700763650"
7  }
```

- In the Database

`SELECT * FROM `customer``

☐ Profiling [ [Edit inline](#) ] [ [Edit](#) ] [ [Explain SQL](#) ] [ [Create PHP code](#) ] [ [Refresh](#) ]

☐ Show all | Number of rows: 25 | Filter rows: Search this table | Sort by key: None

Extra options

		id	adresse	email	nom	telephone		
<input type="checkbox"/>	Edit	Copy	Delete	5d588872-65ed-4f10-a32f-2c2ddaa52823	B 152 Medoa	slm@gmail.com	Salama	0700763650
<input type="checkbox"/>	Edit	Copy	Delete	b73284b3-ae55-4320-8f8a-552ed2af3b5a	B 152 Medoa	essadeq@gmail.com	Essadeq	0700763650

☐ Check all | With selected: Edit Copy Delete Export

## Développer le micro-service Inventory-Service

### 3. Command side

#### 3.1. Commands

- CreateProductCommand

```
package me.elaamiri.commands.productCommands;

public class CreateProductCommand extends BaseCommand<String> {

    @Getter
    private String nom;
    @Getter
    private double prix;
    @Getter
    private int quantiteStoque;
    @Getter
    private ProductStatus productStatus;

    @Getter
    private String categoryId;

    public CreateProductCommand(String id, String nom, double prix, int
quantiteStoque, ProductStatus productStatus, String categoryId) {
        super(id);
        this.nom = nom;
        this.prix = prix;
        this.quantiteStoque = quantiteStoque;
        this.productStatus = productStatus;
        this.categoryId = categoryId;
    }
}
```

- UpdateProductCommand

```
package me.elaamiri.commands.productCommands;

public class UpdateProductCommand extends BaseCommand<String> {
    @Getter
    private String nom;
    @Getter
    private double prix;
    @Getter
    private int quantiteStocke;
    @Getter
    private ProductStatus productStatus;

    @Getter
    private String categoryId;

    public UpdateProductCommand(String id, String nom, double prix, int
quantiteStocke, ProductStatus productStatus, String categoryId) {
        super(id);
        this.nom = nom;
        this.prix = prix;
        this.quantiteStocke = quantiteStocke;
        this.productStatus = productStatus;
        this.categoryId = categoryId;
    }
}
```

- DeleteProductCommand

```
package me.elaamiri.commands.productCommands;

public class DeleteProductCommand extends BaseCommand<String> {
    public DeleteProductCommand(String id) {
        super(id);
    }
}
```

- CreateCategoryCommand

```
package me.elaamiri.commands.categoryCommands;
```

```
public class CreateCategoryCommand extends BaseCommand<String> {

    @Getter
    private String nom;

    @Getter
    private String description;

    public CreateCategoryCommand(String id, String nom, String description) {
        super(id);
        this.nom = nom;
        this.description = description;
    }
}
```

- UpdateCategoryCommand

```
package me.elaamiri.commands.categoryCommands;

public class UpdateCategoryCommand extends BaseCommand<String> {
    @Getter
    private String nom;

    @Getter
    private String description;

    public UpdateCategoryCommand(String id, String nom, String description) {
        super(id);
        this.nom = nom;
        this.description = description;
    }
}
```

- DeleteCategoryCommand

```
package me.elaamiri.commands.categoryCommands;

public class DeleteCategoryCommand extends BaseCommand<String> {
    public DeleteCategoryCommand(String id) {
        super(id);
    }
}
```

## 3.2. Events

- ProductCreatedEvent

```
package me.elaamiri.events.productEvents;

public class ProductCreatedEvent extends BaseEvent<String> {
    @Getter
    private String nom;
    @Getter
    private double prix;
    @Getter
    private int quantiteStoque;
    @Getter
    private ProductStatus productStatus;

    @Getter
    private String categoryId;

    public ProductCreatedEvent(String id, String nom, double prix, int
quantiteStoque, ProductStatus productStatus, String categoryId) {
        super(id);
        this.nom = nom;
        this.prix = prix;
        this.quantiteStoque = quantiteStoque;
        this.productStatus = productStatus;
        this.categoryId = categoryId;
    }
}
```

- ProductUpdatedEvent

```
package me.elaamiri.events.productEvents;

public class ProductUpdatedEvent extends BaseEvent<String> {
    @Getter
    private String nom;
    @Getter
    private double prix;
    @Getter
    private int quantiteStoque;
    @Getter
    private ProductStatus productStatus;

    @Getter
    private String categoryId;

    public ProductUpdatedEvent(String id, String nom, double prix, int
quantiteStoque, ProductStatus productStatus, String categoryId) {
        super(id);
        this.nom = nom;
    }
}
```

```
        this.prix = prix;
        this.quentiteStoque = quentiteStoque;
        this.productStatus = productStatus;
        this.categoryId = categoryId;
    }
}
```

- ProductDeletedEvent

```
package me.elaamiri.events.productEvents;

public class ProductDeletedEvent extends BaseEvent<String> {
    public ProductDeletedEvent(String id) {
        super(id);
    }
}
```

- CategoryCreatedEvent

```
package me.elaamiri.events.categoryEvents;

public class CategoryCreatedEvent extends BaseEvent<String> {
    @Getter
    private String nom;

    @Getter
    private String description;

    public CategoryCreatedEvent(String id, String nom, String description) {
        super(id);
        this.nom = nom;
        this.description = description;
    }
}
```

- CategoryUpdatedEvent

```
package me.elaamiri.events.categoryEvents;

public class CategoryUpdatedEvent extends BaseEvent<String> {
    @Getter
    private String nom;

    @Getter
```

```
private String description;

public CategoryUpdatedEvent(String id, String nom, String description) {
    super(id);
    this.nom = nom;
    this.description = description;
}
}
```

- CategoryDeletedEvent

```
package me.elaamiri.events.categoryEvents;

import me.elaamiri.events.BaseEvent;

public class CategoryDeletedEvent extends BaseEvent<String> {
    public CategoryDeletedEvent(String id) {
        super(id);
    }
}
```

### 3.3. Aggregates

- ProductAggregate

```
package me.elaamiri.inventoryservice.command.aggregates;

@Aggregate @Slf4j
public class ProductAggregate {
    @AggregateIdentifier
    @Getter
    private String id;
    @Getter
    private String nom;
    @Getter
    private double prix;
    @Getter
    private int quantiteStocke;
    @Getter
    private ProductStatus productStatus;

    @Getter
    private String categoryId;

    public ProductAggregate() {
    }
}
```

```
@CommandHandler
public ProductAggregate(CreateProductCommand command) {
    // validations
    AggregateLifecycle.apply(new ProductCreatedEvent(
        command.getId(),
        command.getNom(),
        command.getPrix(),
        command.getQuantiteStoque(),
        command.getProductStatus(),
        command.getCategoryId()
    ));
}
```

```
@EventSourcingHandler
```

```
public void on(ProductCreatedEvent event){
    this.id = event.getId();
    this.productStatus = event.getProductStatus();
    this.nom = event.getNom();
    this.prix = event.getPrix();
    this.quantiteStoque = event.getQuantiteStoque();
    this.categoryId = event.getCategoryId();
}
```

```
@CommandHandler
```

```
public void handle(UpdateProductCommand command){

    if(command.getId() == null || command.getId().isBlank()) throw new
EntryNotFoundException("Product Not Found.");
    // validations
    AggregateLifecycle.apply(new ProductUpdatedEvent(
        command.getId(),
        command.getNom(),
        command.getPrix(),
        command.getQuantiteStoque(),
        command.getProductStatus(),
        command.getCategoryId()
    ));
}
```

```
@EventSourcingHandler
```

```
public void on(ProductUpdatedEvent event){
    this.id = event.getId();
    this.productStatus = event.getProductStatus();
    this.nom = event.getNom();
    this.prix = event.getPrix();
    this.quantiteStoque = event.getQuantiteStoque();
    this.categoryId = event.getCategoryId();
}
```

```
@CommandHandler
```

```
public void handle>DeleteProductCommand command){
```



```

        if(command.getId() == null || command.getId().isBlank()) throw new
EntryNotFoundException("Product Not Found.");
        // validations
        AggregateLifecycle.apply(new ProductDeletedEvent(
            command.getId()));
    }

    @EventSourcingHandler
    public void on(ProductDeletedEvent event){
        log.warn("Deleting Product: "+ event.getId());
    }

}

```

- CategoryAggregate

```

package me.elaamiri.inventoryservice.command.aggregates;

@Aggregate @Slf4j
public class CategoryAggregate {

    @AggregateIdentifier
    @Getter
    private String id;
    @Getter
    private String nom;

    @Getter
    private String description;

    public CategoryAggregate() {
    }

    @CommandHandler
    public CategoryAggregate(CreateCategoryCommand command) {
        // validations
        AggregateLifecycle.apply(new CategoryCreatedEvent(
            command.getId(),
            command.getNom(),
            command.getDescription()
        ));
    }

    @EventSourcingHandler

    public void on(CategoryCreatedEvent event){
        this.id = event.getId();
        this.nom = event.getNom();
        this.description=event.getDescription();
    }
}

```

```

@CommandHandler
public void handle(UpdateCategoryCommand command){

    if(command.getId() == null || command.getId().isBlank()) throw new
EntryNotFoundException("Category Not Found.");
    // validations
    AggregateLifecycle.apply(new CategoryUpdatedEvent(
        command.getId(),
        command.getNom(),
        command.getDescription()
    ));
}

@EventSourcingHandler

public void on(CategoryUpdatedEvent event){
    this.id = event.getId();
    this.description = event.getDescription();
    this.nom = event.getNom();
}

}

@CommandHandler
public void handle>DeleteCategoryCommand command){
    if(command.getId() == null || command.getId().isBlank()) throw new
EntryNotFoundException("Category Not Found.");
    // validations
    AggregateLifecycle.apply(new CategoryDeletedEvent(
        command.getId()));
}

@EventSourcingHandler
public void on(CategoryDeletedEvent event){
    log.warn("Deleting Category: "+ event.getId());
}

}

```

### 3.4. Controllers

- ProductCommandController

```

package me.elaamiri.inventoryservice.command.controllers;

@RestController
@AllArgsConstructor
@RequestMapping(path = "/commands/product")
public class ProductCommandController {

```

```

private CommandGateway commandGateway;

@PostMapping("/create")
public CompletableFuture<String> createProduct(@RequestBody
CreateProductRequestDTO createProductRequestDTO){
    CompletableFuture<String> response = commandGateway.send(new
CreateProductCommand(
        UUID.randomUUID().toString(),
        createProductRequestDTO.getNom(),
        createProductRequestDTO.getPrix(),
        createProductRequestDTO.getQuantiteStoque(),
        createProductRequestDTO.getProductStatus(),
        createProductRequestDTO.getCategoryId()
    ));
    return response;
}

@PutMapping("/update/{id}")
public CompletableFuture<String> updateProduct(@RequestBody
UpdateProductRequestDTO updateProductRequestDTO, @PathVariable String id){
    CompletableFuture<String> response = commandGateway.send(new
UpdateProductCommand(
        id, // entered ID
        updateProductRequestDTO.getNom(),
        updateProductRequestDTO.getPrix(),
        updateProductRequestDTO.getQuantiteStoque(),
        updateProductRequestDTO.getProductStatus(),
        updateProductRequestDTO.getCategoryId()
    ));
    return response;
}

@DeleteMapping("/delete/{id}")
public CompletableFuture<String> deleteProduct(@PathVariable String id){
    CompletableFuture<String> response = commandGateway.send(new
DeleteProductCommand(
        id// entered ID
    ));
    return response;
}

//@ExceptionHandler(Exception.class)
public ResponseEntity<String> exceptionHandler(Exception exception){
    ResponseEntity<String> responseEntity = new ResponseEntity<>(
        exception.getMessage(), HttpStatus.INTERNAL_SERVER_ERROR
    );

    return responseEntity;
}
}

```

- CategoryCommandController

```
package me.elaamiri.inventoryservice.command.controllers;

@RestController
@AllArgsConstructor
@RequestMapping(path = "/commands/category")
public class CategoryCommandController {
    private CommandGateway commandGateway;

    @PostMapping("/create")
    public CompletableFuture<String> createCategory(@RequestBody
CreateCategoryRequestDTO createCategoryRequestDTO){
        CompletableFuture<String> response = commandGateway.send(new
CreateCategoryCommand(
            UUID.randomUUID().toString(),
            createCategoryRequestDTO.getNom(),
            createCategoryRequestDTO.getDescription()
        ));
        return response;
    }

    @PutMapping("/update/{id}")
    public CompletableFuture<String> updateCategory(@RequestBody
UpdateCategoryRequestDTO updateCategoryRequestDTO, @PathVariable String id){
        CompletableFuture<String> response = commandGateway.send(new
UpdateCategoryCommand(
            id, // entered ID
            updateCategoryRequestDTO.getNom(),
            updateCategoryRequestDTO.getDescription()));
        return response;
    }

    @DeleteMapping("/delete/{id}")
    public CompletableFuture<String> deleteCategory(@PathVariable String id){
        CompletableFuture<String> response = commandGateway.send(new
DeleteCategoryCommand(
            id// entered ID
        ));
        return response;
    }

    //ExceptionHandler(Exception.class)
    public ResponseEntity<String> exceptionHandler(Exception exception){
        ResponseEntity<String> responseEntity = new ResponseEntity<>(
            exception.getMessage(), HttpStatus.INTERNAL_SERVER_ERROR
        );

        return responseEntity;
    }
}
```

3.5. tests

- Creation Category

http://localhost:8082/commands/category/create

Save

POST

http://localhost:8082/commands/category/create

Params

Authorization

Headers (9)

Body

Pre-request Script

Tests

Settings

none

form-data

x-www-form-urlencoded

raw

binary

GraphQL

JSON

1

2

3

4

...

"nom": "PC",

"description": "PCs"

Body

Cookies

Headers (5)

Test Results

Pretty

Raw

Preview

Visualize

Text

1

f3dbe9d9-7574-493b-9b54-05df95781b8c

- Updating Category

http://localhost:8082/commands/category/update/f3dbe9d9-7574-493b-9b54-05df95781b8c

Save

PUT

http://localhost:8082/commands/category/update/f3dbe9d9-7574-493b-9b54-05df95781b8c

Params

Authorization

Headers (9)

Body

Pre-request Script

Tests

Settings

none

form-data

x-www-form-urlencoded

raw

binary

GraphQL

JSON

1

2

3

4

...

"nom": "PC & printers",

"description": "PCs"

Body

Cookies

Headers (4)

Test Results

Pretty

Raw

Preview

Visualize

Text

1

- Creating Product

http://localhost:8082/commands/product/create

POST

http://localhost:8082/commands/product/create

ParamsAuthorizationHeaders (9)BodyPre-request ScriptTestsSettings

none

form-data

x-www-form-urlencoded

raw

binary

GraphQL

JSON

1

2

3

4

5

6

7

body

```
1  {
2    "nom": "DELL 54",
3    "prix": 15000.25,
4    "quantiteStoque": 54,
5    "productStatus": "DISPONIBLE",
6    "categoryId": "f3dbe9d9-7574-493b-9b54-05df95781b8c"
7  }
```

BodyCookiesHeaders (5)Test Results

200 OK

56

Pretty

Raw

Preview

Visualize

Text

1

58faf7c0-b3bb-4a10-b469-3701f01685a7

Updating Product

http://localhost:8082/commands/product/update/58faf7c0-b3bb-4a10-b469-3701f01685a7

PUT

http://localhost:8082/commands/product/update/58faf7c0-b3bb-4a10-b469-3701f01685a7

ParamsAuthorizationHeaders (9)BodyPre-request ScriptTestsSettings

none

form-data

x-www-form-urlencoded

raw

binary

GraphQL

JSON

1

2

3

4

5

6

7

body

```
1  {
2    "nom": "DELL 54",
3    "prix": 15000.25,
4    "quantiteStoque": 102,
5    "productStatus": "DISPONIBLE",
6    "categoryId": "f3dbe9d9-7574-493b-9b54-05df95781b8c"
7  }
```

BodyCookiesHeaders (4)Test Results

200 OK

170 ms

123 B

Pretty

Raw

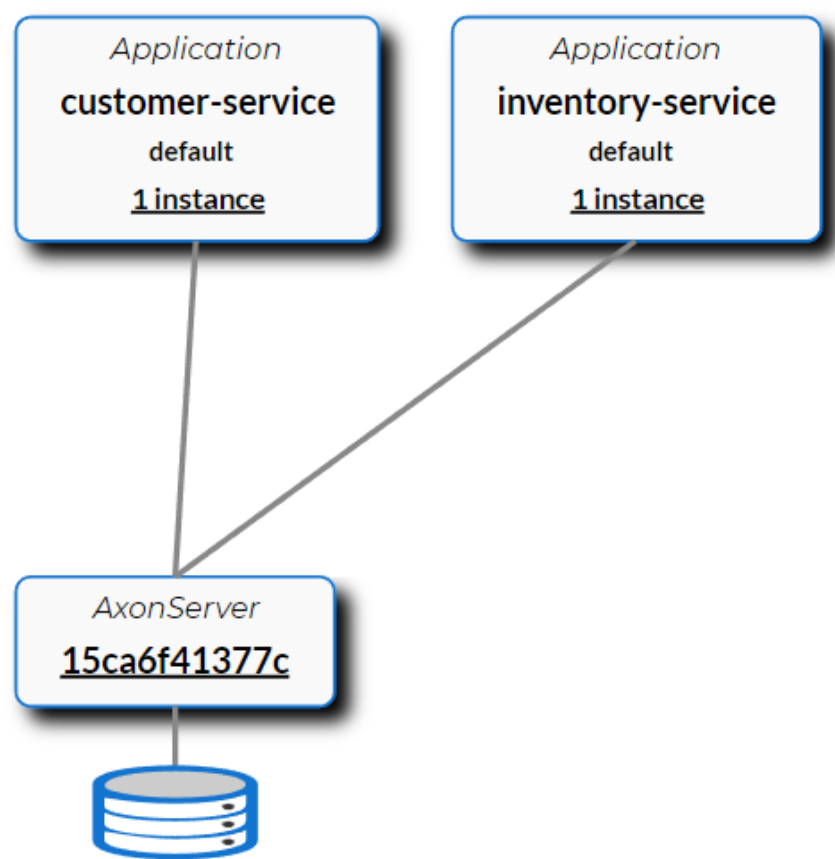
Preview

Visualize

Text

1

- Deleting Product
- Dans le serveur



- Les evenements dans Axon



Search: ☒ Events ☐ Snapshots

Enter your query here

About the query language

token	eventIdentifier	aggregateIdentifier	aggregateSeque...	aggregateType	payloadType	paylo
61	82e0207d-ce6d-416b-a5a6...	32329a38-c364-4fef-a7e2-...	0	ProductAggregate	me.elaamiri.events.productEvents.ProductCreatedEvent	
60	9a2c9847-2d1d-40eb-92f1...	f3dbe9d9-7574-493b-9b54...	1	CategoryAggregate	me.elaamiri.events.categoryEvents.CategoryUpdatedEvent	
59	09c4dacb-68fb-4266-94fc-...	58faf7c0-b3bb-4a10-b469-...	1	ProductAggregate	me.elaamiri.events.productEvents.ProductUpdatedEvent	
58	51b9f4e9-e42f-413a-871b-...	58faf7c0-b3bb-4a10-b469-...	0	ProductAggregate	me.elaamiri.events.productEvents.ProductCreatedEvent	
57	96527147-ddff-4587-8065...	f3dbe9d9-7574-493b-9b54...	0	CategoryAggregate	me.elaamiri.events.categoryEvents.CategoryCreatedEvent	

## 4.1. Queries

- GetAllProductsQuery

```
package me.elaamiri.queries.productQueries;

public class GetAllProductsQuery {
}
```

- GetProductByIdQuery

```
package me.elaamiri.queries.productQueries;

import lombok.Getter;

public class GetProductByIdQuery {

    @Getter
    private String id;

    public GetProductByIdQuery(String id) {
        this.id = id;
    }
}
```

- GetAllCategoriesQuery

```
package me.elaamiri.queries.categoryQueries;

public class GetAllCategoriesQuery {
}
```

- GetCategoryByIdQuery

```
package me.elaamiri.queries.categoryQueries;

public class GetCategoryByIdQuery {

    @Getter
    private String id;

    public GetCategoryByIdQuery(String id) {
        this.id = id;
    }
}
```



```
}  
}
```

## 4.2. Entities

- 

```
package me.elaamiri.inventoryservice.query.entities;  
  
@Entity  
@Data @AllArgsConstructor @NoArgsConstructor  
public class Product {  
  
    @Id  
    private String id;  
  
    private String nom;  
  
    private double prix;  
  
    private int quantiteStocke;  
  
    private ProductStatus productStatus;  
  
    private String categoryId;  
    @ManyToOne  
    private Category category;  
}
```

- Category

```
package me.elaamiri.inventoryservice.query.entities;  
  
@Entity  
@Data @AllArgsConstructor @NoArgsConstructor  
public class Category {  
  
    @Id  
    private String id;  
  
    private String nom;  
  
    private String description;  
  
    @OneToMany(mappedBy = "category")
```

```
    @JsonProperty(access = JsonProperty.Access.WRITE_ONLY)
    private Collection<Product> products;
}
```

### 4.3. Repositories

- ProductRepository

```
package me.elaamiri.inventoryservice.query.repositories;

public interface ProductRepository extends JpaRepository<Product, String> {
}
```

- CategoryRepository

```
package me.elaamiri.inventoryservice.query.repositories;

public interface CategoryRepository extends JpaRepository<Category, String> {
}
```

### 4.4. EventHandler Services

- ProductEventHandlerService

```
package me.elaamiri.inventoryservice.query.services;

@Service
@Slf4j
@AllArgsConstructor
public class ProductEventHandlerService {

    private ProductRepository productRepository;
    private CategoryRepository categoryRepository;

    @EventHandler
    public void on(ProductCreatedEvent event){
        Product product = Product.builder()
            .id(event.getId())
            .nom(event.getNom())
            .categoryId(event.getCategoryId())
            .productStatus(event.getProductStatus())
            .prix(event.getPrix())
            .quentiteStocke(event.getQuentiteStocke())
            .build();
    }
}
```

```

        // find Category
        Category category =
categoryRepository.findById(event.getCategoryId()).orElseThrow(() -> new
EntryNotFoundException("Category not found"));
        product.setCategory(category);
        productRepository.save(product);
    }

    @EventHandler
    public void on(ProductUpdatedEvent event){
        Product product = productRepository.findById(event.getId())
            .orElseThrow(() -> new EntryNotFoundException("Product Not
Found"));
        product.setNom(event.getNom());
        product.setProductStatus(event.getProductStatus());
        product.setCategoryId(event.getCategoryId());
        product.setPrix(event.getPrix());
        product.setQuentiteStocke(event.getQuentiteStocke());

        Category category =
categoryRepository.findById(event.getCategoryId()).orElseThrow(() -> new
EntryNotFoundException("Category not found"));
        product.setCategory(category);
        // product.setId(event.getId());
        productRepository.save(product);
    }

    @EventHandler
    public void on(ProductDeletedEvent event){
        Product product = productRepository.findById(event.getId())
            .orElseThrow(() -> new EntryNotFoundException("Product Not
Found"));
        productRepository.delete(product);
    }
}

```

- 

#### 4.5. QueryHandler Services

-

- 

#### 4.6. Query Controllers

- 

- 

#### 4.7. Tests

## Développer le micro-service Order-Service

---

### Mettre en place les services techniques de l'architecture micro-service (Gateway, Eureka ou Consul Discovery service, Config Service)

---

## 5. Discovery service

- pom.xml

```
<dependency>
  <groupId>org.springframework.cloud</groupId>
  <artifactId>spring-cloud-starter-netflix-eureka-server</artifactId>
</dependency>
```

- properties

```
server.port=8761 # it is the default port used by the services to register
# do not register itself as a client
eureka.client.fetch-registry=false
# Does not register itself in the service registry
eureka.client.register-with-eureka=false
```

- Application

```
@EnableEurekaServer
@SpringBootApplication
public class EurekaDiscoveryServiceApplication {

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

}
```

## 6. Gateway service

- Dependency

```
<dependencies>
    <dependency>
        <groupId>org.springframework.cloud</groupId>
        <artifactId>spring-cloud-starter-gateway</artifactId>
    </dependency>
    <dependency>
        <groupId>org.springframework.cloud</groupId>
        <artifactId>spring-cloud-starter-netflix-eureka-client</artifactId>
    </dependency>
    ...
</dependencies>
```

- Properties

```
server.port=8888
spring.application.name=GATEWAY-SERVICE
## the gateway also should be registered in the discovery server
eureka.client.register-with-eureka=true
eureka.instance.prefer-ip-address=true
```

- Configuration

```
package me.elaamiri.gatewayservice;

@Configuration
public class GatewayConfig {
```

```
@Bean
DiscoveryClientRouteDefinitionLocator
discoveryClientRouteDefinitionLocator(ReactiveDiscoveryClient
reactiveDiscoveryClient,

DiscoveryLocatorProperties discoveryLocatorProperties){
    return new DiscoveryClientRouteDefinitionLocator(reactiveDiscoveryClient,
discoveryLocatorProperties);
}
}
```

Développer un micro-service qui permet faire du Real time Data Analytics en utilisant Kafka Streams (Nombre et total des commandes sur une fenêtre temporelle de 5 secondes)

---

Développer votre application Frontend avec Angular ou React

---

Sécuriser votre système avec un système de d'authentification OAuth2, OIDC avec Keycloak ou un service d'authentification basé sur Spring Security et JWT

---

Écrire un script docker-compose.yml pour le déploiement de ce système distribué dans des

---

conteneurs docker.