R5.08 - Qualité de développement

Enseignant: Dalaigre C.

Baptiste HAUDEBOURG - Colin PILET - Melchor RUIZ GONZÁLEZ

2024 - 2025

TP 1

Tâche 1 : Ségrégation des responsabilités

Quels sont les principaux domaines métiers de l'application Order flow ?

Les principaux domaines métiers de l’application Order Flow sont les domaines principaux, domaines de support, domaines génériques

Comment les microservices sont-ils conçus pour implémenter les domaines métiers ?

Les microservices sont conçus pour implémenter les domaines métiers en utilisant des principes de Domain-Driven Design (DDD) et d'architecture événementielle (Event-Driven Architecture). Chaque microservice est responsable d'un domaine métier spécifique et encapsule la logique métier, les commandes, les événements et les requêtes associés à ce domaine.

Quelles sont les responsabilités des conteneurs de code ?

- of-api-gateway : Ce conteneur expose la logique métier sous forme d'API HTTP. Il agit comme une passerelle API pour les autres microservices, permettant aux clients externes d'interagir avec le système.
- apps/of-product-registry-microservices/product.registry : Ce microservice gère les commandes liées au registre des produits. Il est responsable de l'enregistrement, de la mise à jour et de la suppression des produits dans le registre.
- apps/of-product-registry-microservices/product.registry.read : Ce microservice gère les requêtes de lecture liées au registre des produits. Il est responsable de la projection des événements dans un modèle de lecture et de la fourniture des données de produit aux clients.
- event-sourcing : Cette bibliothèque contient la logique de gestion des événements. Elle fournit des interfaces et des implémentations pour le stockage et la lecture des événements, permettant aux microservices de persister et de rejouer les événements.
- published-language : Cette bibliothèque définit le langage partagé entre les microservices. Elle contient les objets de valeur, les commandes, les événements et les autres types partagés utilisés pour la communication entre les microservices.

Tâche 2 : Identifier les concepts principaux

Quels sont les concepts principaux utilisés dans l'application Order flow ?

Les principaux concepts utilisés dans l’application Order flow sont CQRS, sourcing d'événements.

Comment les concepts principaux sont-ils implémentés dans les microservices ?

Les concepts principaux dans les microservices sont implémentés en utilisant des principes de conception tels que le Domain-Driven Design (DDD), l'Event Sourcing, et la communication asynchrone.

Que fait la bibliothèque `libs/event-sourcing` ? Comment est-elle utilisée dans les microservices (relation entre métier et structure du code) ?

La bibliothèque event-sourcing est responsable de la gestion des événements dans le système. Elle fournit des interfaces et des implémentations pour le stockage, la lecture et la manipulation des événements.

Comment l'implémentation actuelle de l'event-sourcing assure-t-elle la fiabilité des états internes de l'application ?

L'implémentation de l'Event Sourcing assure la fiabilité des états internes de l'application en capturant chaque changement d'état sous forme d'événement immuable.

Tâche 3 : Identifier les problèmes de qualité

Immutabilité partielle

Les champs name et productDescription sont déclarés comme final, ce qui est bien pour l'immutabilité. Cependant, il n'y a pas de méthode pour modifier ces champs, ce qui pourrait être une limitation si des mises à jour sont nécessaires.

Manque de tests unitaires

Il n'y a pas de tests unitaires visibles pour vérifier le comportement de la classe Product.

Manque de gestion des exceptions

Le constructeur ne gère pas les exceptions potentielles qui pourraient survenir lors de l'initialisation des champs.

TP 2

Tâche 1 : Compléter les commentaires et la Javadoc

Le microservice de registre de produits est un microservice côté écriture qui gère les produits disponibles à la vente et leur disponibilité pour l'ajout dans un catalogue.
Le microservice n'est pas complètement documenté. Vous devez compléter les commentaires et la Javadoc du microservice.

Tâche 4 : Modifier les Entités (couche de persistance) pour utiliser des champs privés avec des accesseurs

Tâche 4.1 : Champs publics sur les entités Panache

```java
import io.quarkus.mongodb.panache.common.Mon

@MongoEntity(collection = "products")
public class ProductEntity {
  public ObjectId id;
  public String productId;
  public String name;
  public String description;
  public long version;
  public long updatedAt;
  public long registeredAt;
}
```

```java
@MongoEntity(collection = "product_registry_events")
public abstract class ProductRegistryEventEntity {
  public ObjectId id;
  public String eventId;
  public String eventType = getEventType();
  public String aggregateRootId;
  public long version;
  public long timestamp;

  abstract String getEventType();
}
```

TP 3