LOG430 – Architecture Logicielle (E´t´e 2025) Laboratoire 6 – Auto-´evaluation d´etaill´ee

Nom : Maksym Pravdin Code permanent : PRAM86290201

# Informations g´en´erales

* URL du d´epˆot GitHub/GitLab : https://github.com/PraMaks/LOG430\_Labo\_0/
* Langage et framework principaux (Spring Boot, Quarkus, NestJS, . . .) : Django (frontend) Express.js (backend)
* Composant d’orchestration (service d´edi´e, workflow engine, . . .) : Service dédié: Orchestr-sales-service
* Bus de messages / m´ecanisme de communication (file, ´echange direct, . . .) : Logger avec pino
* Technologie de persistance de la machine d’´etat (base SQL/NoSQL, in-memory, . . .) : MongoDB
* Outil(s) d’observabilit´e (collecte de m´etriques, traces, visualisation) : Prometheus + Grafana

# E´valuation technique par composant

* 1. **Conception de la saga**

|  |  |  |  |
| --- | --- | --- | --- |
| Crit`ere | Oui | Non | Commentaire / Justification |
| Le sc´enario m´etier (4 ´etapes max.) est clairement  d´ecrit dans le code et le README | ✓ |  | Oui, scénario d’enregistrement d’une commande en ligne est choisi et est indiqué dans le code, le rapport et le README |
| Un service orchestrateur central coordonne la  s´equence d’´etapes | ✓ |  | Oui, service orchestr-sales-service est utilisée. |
| L’orchestrateur publie un *´ev´enement de*  *d´emarrage* et met `a jour l’´etat `a chaque  ´etape | ✓ |  | Oui, les évènements sont mis à jour au cours de l’évènement et son état est changé en cours de route. |

* 1. **Impl´ementation des ´ev´enements et de la machine d’´etat**

|  |  |  |  |
| --- | --- | --- | --- |
| Crit`ere | Oui | Non | Commentaire / Justification |
| Chaque microservice publie un ´ev´enement de  succ`es ou d’´echec | ✓ |  | Oui, après chaque service un état est assigné à l’evenement en cours selon le résultat |
| L’´etat courant de la commande est persistant et  interrogeable | ✓ |  | Oui, l’état est mis à jour au cours de route et est utilisée à partir d’un enum |
| Un diagramme de la machine d’´etat est fourni  dans le rapport | ✓ |  | Oui. Un tableau est fourni dans le rapport pour expliquer les états |
| Les ´ev´enements sont idempotents (gestion des  doublons) | ✓ |  | Oui, au cours d’un evenement, la même instance est mis à jour et une nouvelle est crée uniquement quand c’est une nouvelle commande. |

* 1. **Gestion des ´echecs et compensations**

|  |  |  |  |
| --- | --- | --- | --- |
| Crit`ere | Oui | Non | Commentaire / Justification |
| Des ´echecs simul´es (stock insuffisant, paiement re-  fus´e, . . .) sont document´es | ✓ |  | Oui, les tests avec des résultats variées sont utilisées. |
| Des actions de compensation ou de rollback sont  impl´ement´ees | ✓ |  | Oui, si une étape échoue il y a des actions de rollback qui se produisent |
| Les temps de d´etection d’´echec et de compensa-  tion sont mesur´es | ✓ |  | Oui, ceci peut être obtenu à partir de Grafana + Prometheus |
| Aucun processus ne reste bloqu´e en ´etat in-  term´ediaire | ✓ |  | Oui, il y a des timeouts pour les processus. |

* 1. **Observabilit´e et tra¸cabilit´e**

|  |  |  |  |
| --- | --- | --- | --- |
| Crit`ere | Oui | Non | Commentaire / Justification |
| Un syst`eme de collecte de m´etriques est branch´e  `a l’orchestrateur et aux services | ✓ |  | Oui Prometheus et Grafana sont utilisées |
| Des indicateurs cl´es (dur´ee moyenne d’une saga,  taux d’´echecs, ´etapes atteintes) sont expos´es | ✓ |  | Oui, Prometheus utilise des métrqiues custom que j’ai créé |
| Un tableau de bord regroupe ces m´etriques (vi-  sualisation personnalis´ee ou template) | ✓ |  | Oui, dashboard grafana est utilisée pourles afficher |
| Les logs structur´es permettent de suivre une saga  de bout en bout (trace ID, correlation ID) | ✓ |  | Oui, il est possible de suivre la saga avec les events et les logs |

* 1. **Tests de sc´enarios**

|  |  |  |  |
| --- | --- | --- | --- |
| Crit`ere | Oui | Non | Commentaire / Justification |
| Un jeu de tests automatis´es couvre les sc´enarios  de succ`es | ✓ |  | Oui, il y a des tests pour les succès |
| Des tests couvrent les ´echecs et v´erifient les com-  pensations | ✓ |  | Oui, il y a des tests d’échecs |
| Les tests sont int´egr´es `a la CI/CD | ✓ |  | Oui, ces tests sont ajoutés aux pipeline et la creation de l’image |
| Des rapports ou captures d’ex´ecution des tests  sont inclus dans le livrable | ✓ |  | Oui, ceci est dans le rapport |

# R´eflexion personnelle approfondie

1. **Choix d’orchestration :** Pourquoi avoir opt´e pour une saga orchestr´ee (et non chor´egraphi´ee) ? Quels avantages ou compromis avez-vous observ´es ?

La saga orchestrée permet d’avoir un contrôle sur toutes les étapes et gérer les échecs. Ainsi, on gange en termes de visibilité aussi pour Grafana + Prometheus. Par contre, cela peut rendre le service d’orchestration comme un bottleneck et ajoute du couplage entre les services.

1. **Machine d’´etat :** Comment avez-vous mod´elis´e et persist´e l’´etat ? Quelles difficult´es li´ees `a la coh´erence et `a la r´ecup´eration apr`es panne avez-vous rencontr´ees ?

Pour la persistance la base de donnnées MongoDB est utilisée pour chaque evenement d’enregistrement d’une commande en ligne et celle-ci est mis à jour avec la gestion des enums et de logs au cours de la même saga. Il a fallu ajouter des scenarios de rollback dans le cas que ça crash ou milieu du traitement.

1. **Compensation :** Quelle strat´egie de compensation avez-vous retenue ? Dans quel cas votre solution pourrait-elle ´echouer ou engendrer des inconsistances ?

Si la commande échoue, celle-ci possède l’état d’échec et s’arrête clairement avec des rollback s’il sont nécessaires. La solution pourrait échouer lorsque le service est surchargé en termes de trafic sans nécessairement être horsligne.

1. **Observabilit´e :** Quels indicateurs se sont r´ev´el´es les plus utiles pour diagnostiquer un inci- dent dans la saga ? Comment les visualiser rapidement ?

Grafana est utilisée pour visualiser les métriques custom de Prometheus et le plus utile était le temps moyen d’une saga et le nombre d’état traitées.

1. **Tests :** Quelles astuces avez-vous mises en place pour fiabiliser les tests des sc´enarios com- portant des ´echecs al´eatoires ?

Des tests d’échecs pour la non-connection au serveur était mis en place ainsi qu’une simulation d’échec en plein milieu de la saga qui nécessite un rollback.

1. **R´eutilisation professionnelle :** Quelles pratiques ou outils de ce laboratoire comptez-vous appliquer dans un contexte industriel (ex. e-commerce, finance) ?

Continuer d’utiliser Grafana+Prometheus pour la visibilité des performances et la notion d’orchestration pour les systèmes plus complexes dans une architecture orientée en services.