Bivouac N' Go

LIVRABLE - 2

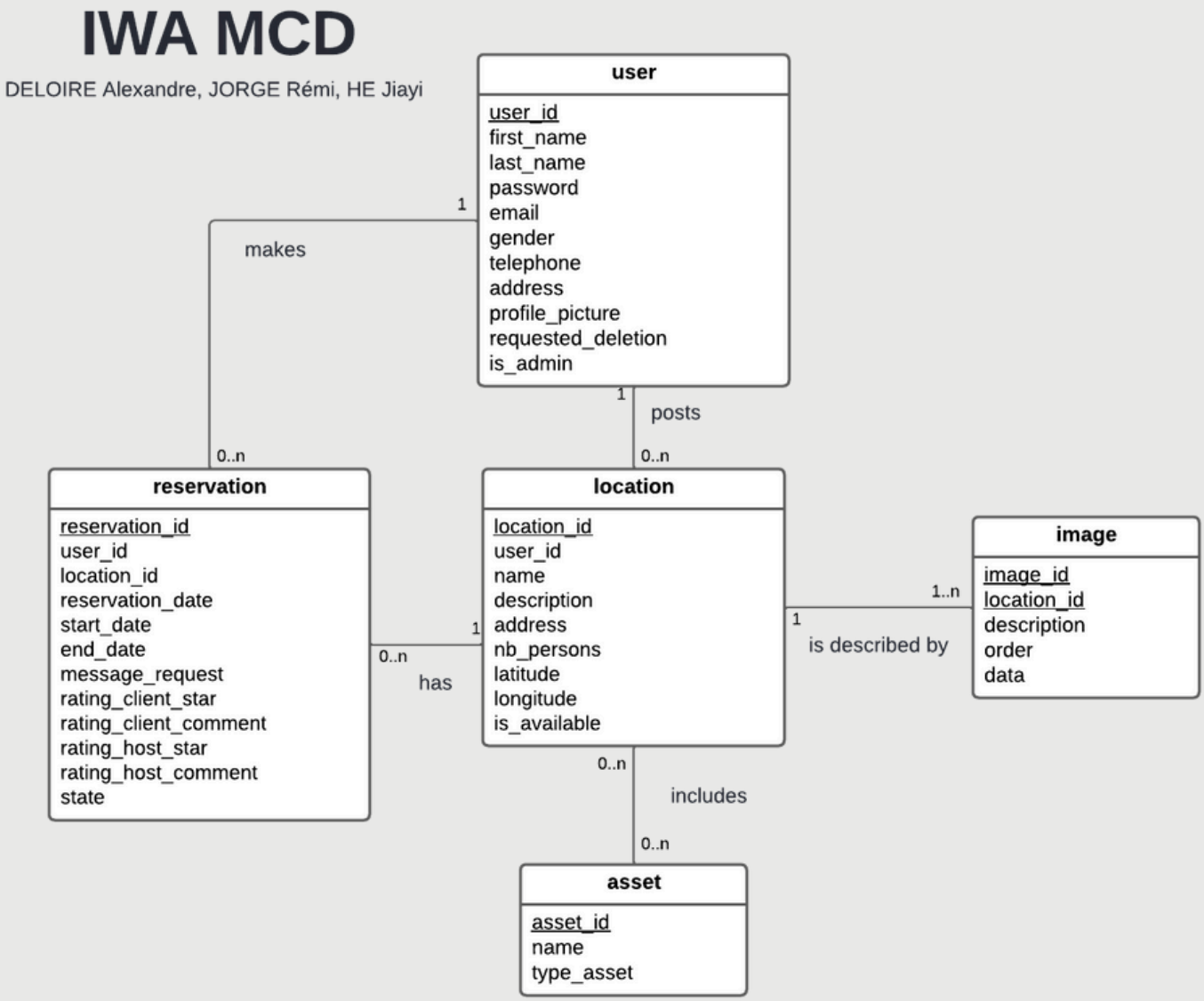
Professeur
NAUROY Christophe

Groupe
DELOIRE Alexandre
JORGE Rémi
HE Jiayi

Sommaire

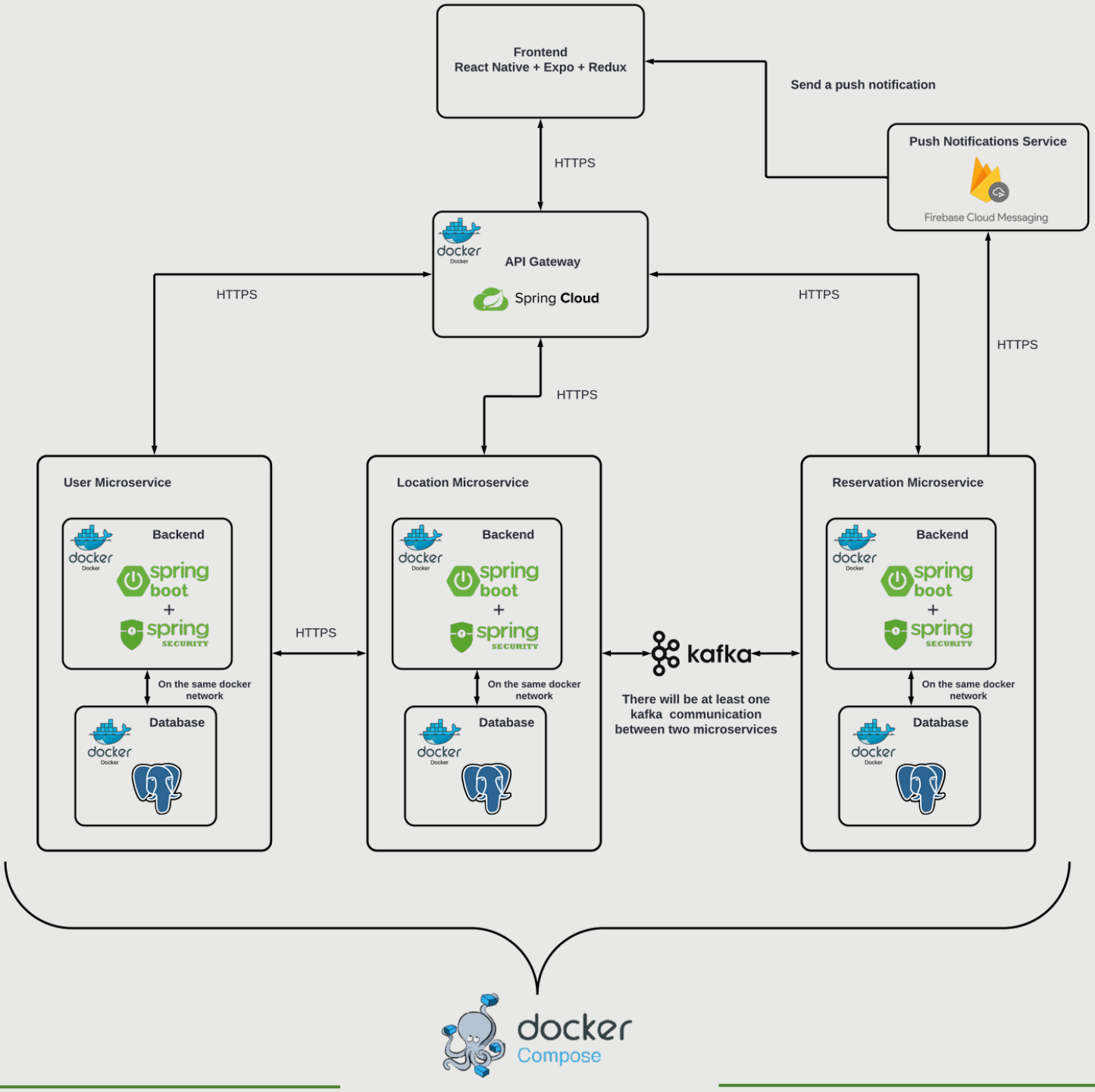
LIVRABLE - 2

01 Rappel du MCD



02 Architecture Microservices

Afin d'assurer une meilleure séparation des responsabilités et une organisation claire des fonctionnalités, nous avons décidé de structurer l'application autour de trois microservices distincts, conformément au modèle MCD conçu dans le livrable 1.



- **Communication**

Nous allons implémenter une **API Gateway** comme point d'entrée unique pour toutes les requêtes externes adressées à l'API. Pour cela, nous allons utiliser **Spring Cloud**, qui permettra de centraliser la gestion des requêtes et le routage vers les microservices appropriés.

Pour la communication entre les microservices, nous allons utiliser des requêtes HTTPS et **Apache Kafka**.

- **Sécurité**

Nous allons utiliser **Spring Security**, une dépendance pour mettre en place des mécanismes de sécurité (authentification et contrôles d'accès) au sein de chaque microservice.

- **Notifications Push**

Nous allons utiliser **Firebase Cloud Messaging** pour assurer les notifications push de l'application.

- **Base de données**

Chaque microservice possède une base de données **PostgreSQL**.

- **Conteneurisation et Orchestration des conteneurs**

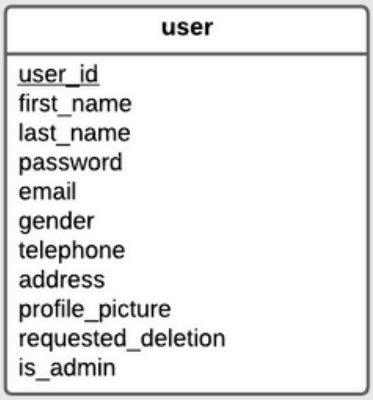
Chaque microservice est composé de deux images **Docker**, une pour l'application Spring Boot et une pour la base de données. L'API Gateway sera aussi dans un conteneur Docker.

Comme les microservices seront déployés sur une seule machine, les conteneurs seront orchestrés avec **Docker Compose**.

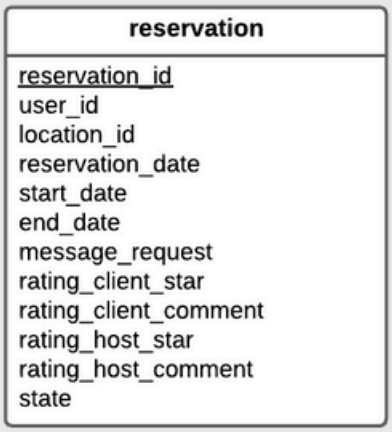
03 MCD Par Microservice

Après la séparation des fonctionnalités par microservice, nous avons réorganisé notre base de données pour répondre à cette nouvelle architecture.

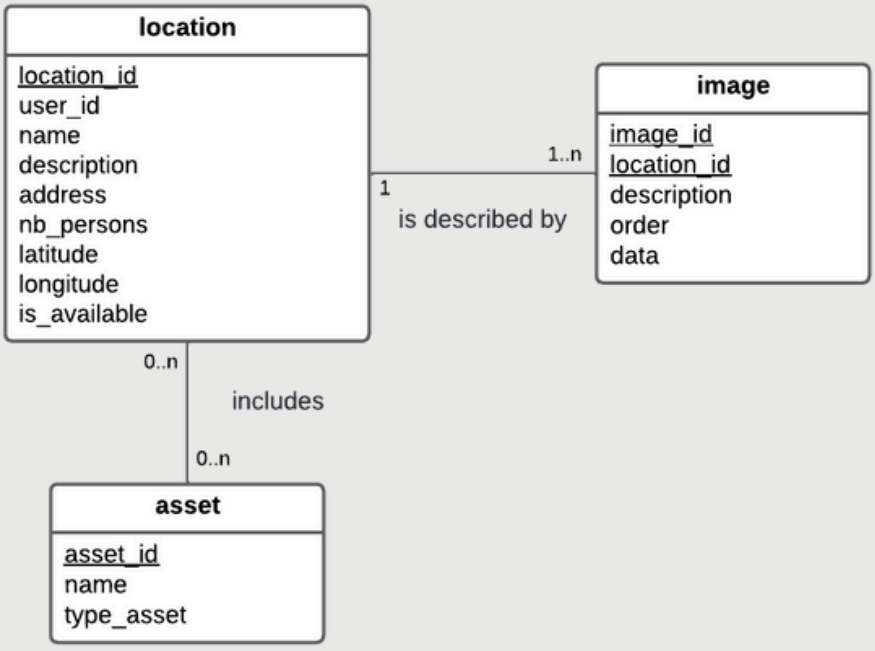
- **Microservice User**



- **Microservice Reservation**



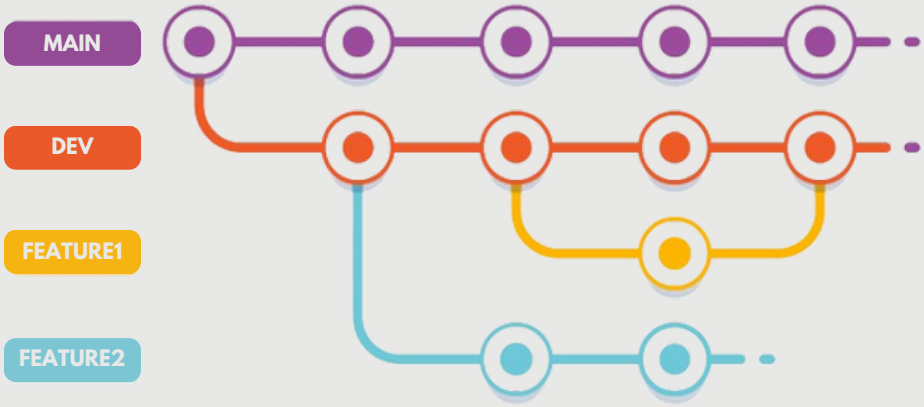
- **Microservice Location**



04 Architecture CI/CD

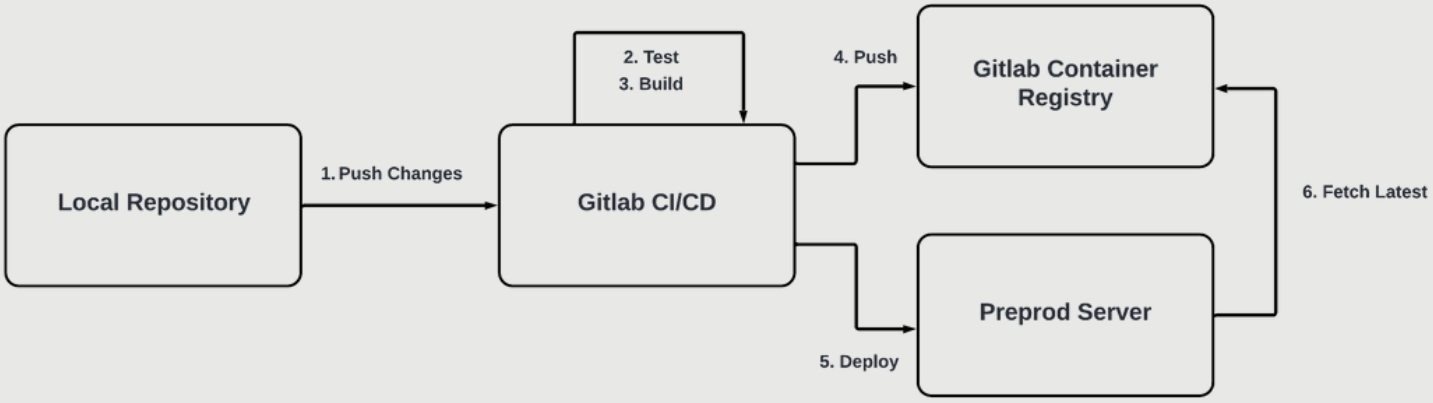
- **Git Flow**

Nous allons utiliser un git flow classique, c'est-à-dire une branche main et dev, et chaque feature aura une branche à partir de la branche dev. Quand on sera satisfait de l'état de la branche dev, on la merge dans main.



- **Gitlab CI/CD**

Nous allons utiliser **Gitlab CI/CD** pour automatiser l'exécution de certains tests, le build et le déploiement sur le serveur de Préprod. Voici un schéma du flow CI/CD.



05 Liens Répos Gitlab

Voici les liens vers le code source :

- **Application Frontend Android**

https://gitlab.polytech.umontpellier.fr/alexandre.deloire01/camping-android-app

- **Ensemble des Microservices**

https://gitlab.polytech.umontpellier.fr/alexandre.deloire01/camping-microservices-app