CarGoverload

Solution de réservation de wagons de transport de marchandises

Par :

- MASIA Sylvain
- MONTOYA Damien
- PERES Richard
- RIGAUT François

Sommaire

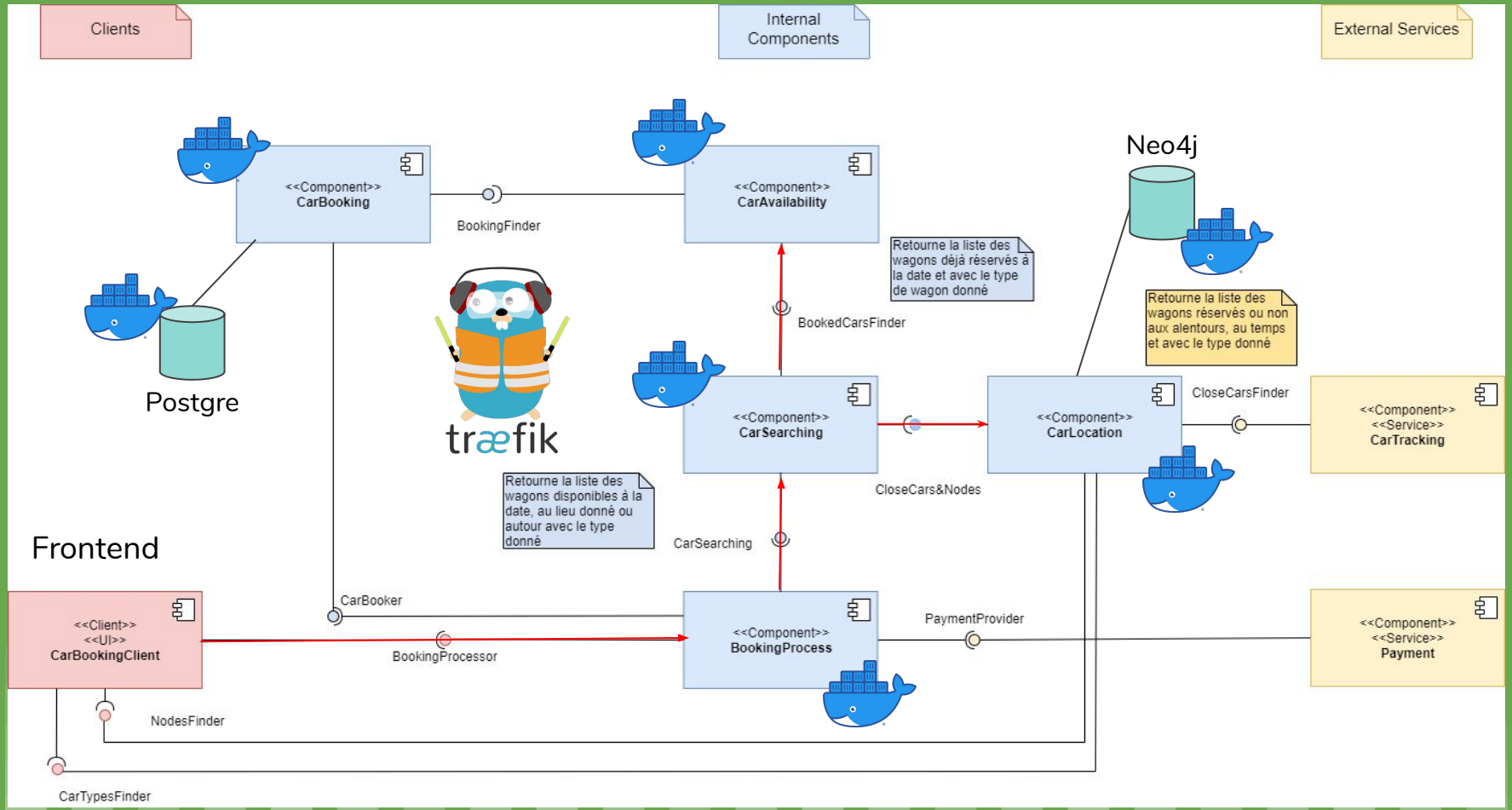
- Sujet et use cases
- Choix architecturaux et technologiques
- Démonstration et dataflow
- Organisation de l’équipe
- Perspectives d’amélioration

Sujet et use cases

Les utilisateurs interagissant avec notre système sont les fournisseurs de marchandises.

- **Réserver** des wagons de transport de marchandises en fonction de leur type, un départ, une arrivée et une date.
- **Gérer** des noeuds de chargement qui peuvent charger certains types de wagons
- **Suggérer** d'autres noeuds si ceux recherchés ne permettent pas de charger le type de wagon recherché

Architecture et choix technologiques



Architecture - technique

- Architecture orientée **composants** (pas assez découplés et résilients pour s'appeler **services**)
- Tous les composants sont **Dockerisés**
- Middleware **Traefik** pour **networking** et **reverse-proxy**
- **Stockage** limités à son composant / domaine
- **Multi-serveur** ready (hosts, ports et URL en variables d’environnements)

Architecture - applications

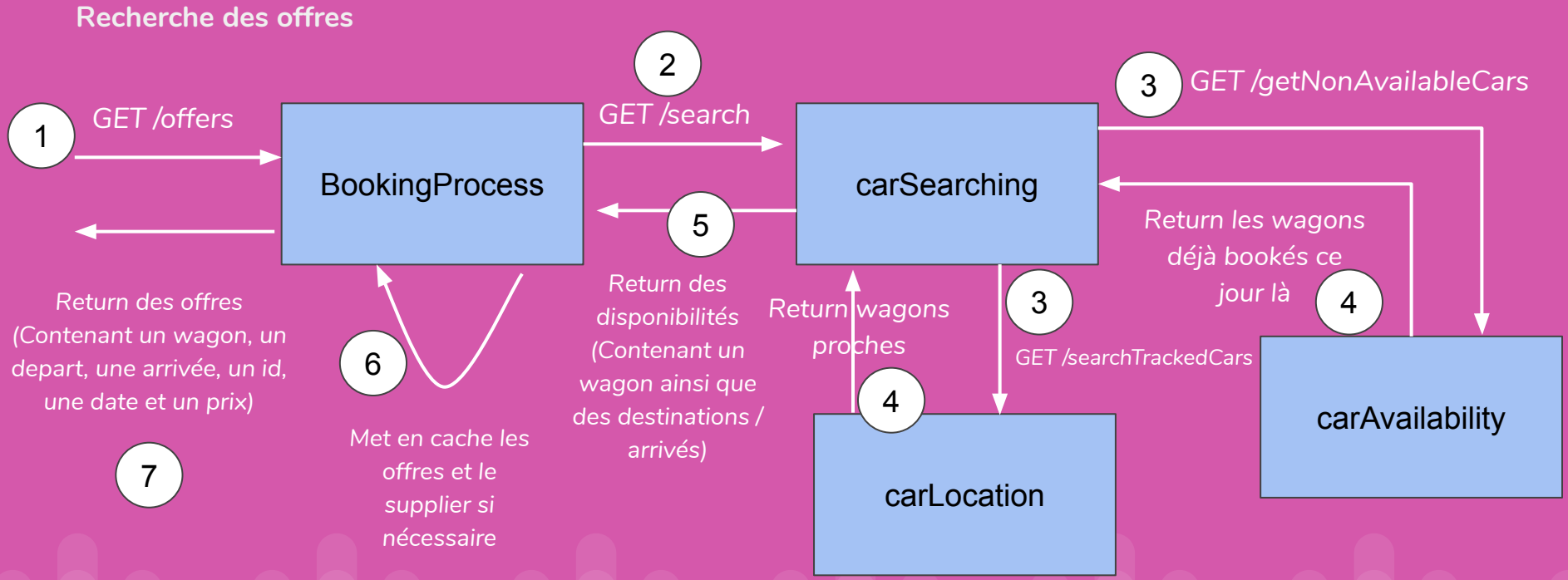
- **Golang** pour la **performance** algorithmique, la gestion de la concurrence, le coût d'implémentation, et la scalabilité
- **Javascript** pour le coût et la **compatibilité** avec **Neo4j** (driver déjà existant et fonctionnel).
- **Base de données** orientée graphe **Neo4j** pour la représentation des Nodes, leurs liens et leurs distances.
- Base de données **PostgreSQL** pour les **réservations**
  - car structure de données fixes (pouvoir profiter des propriétés ACID).

Scénario du POC

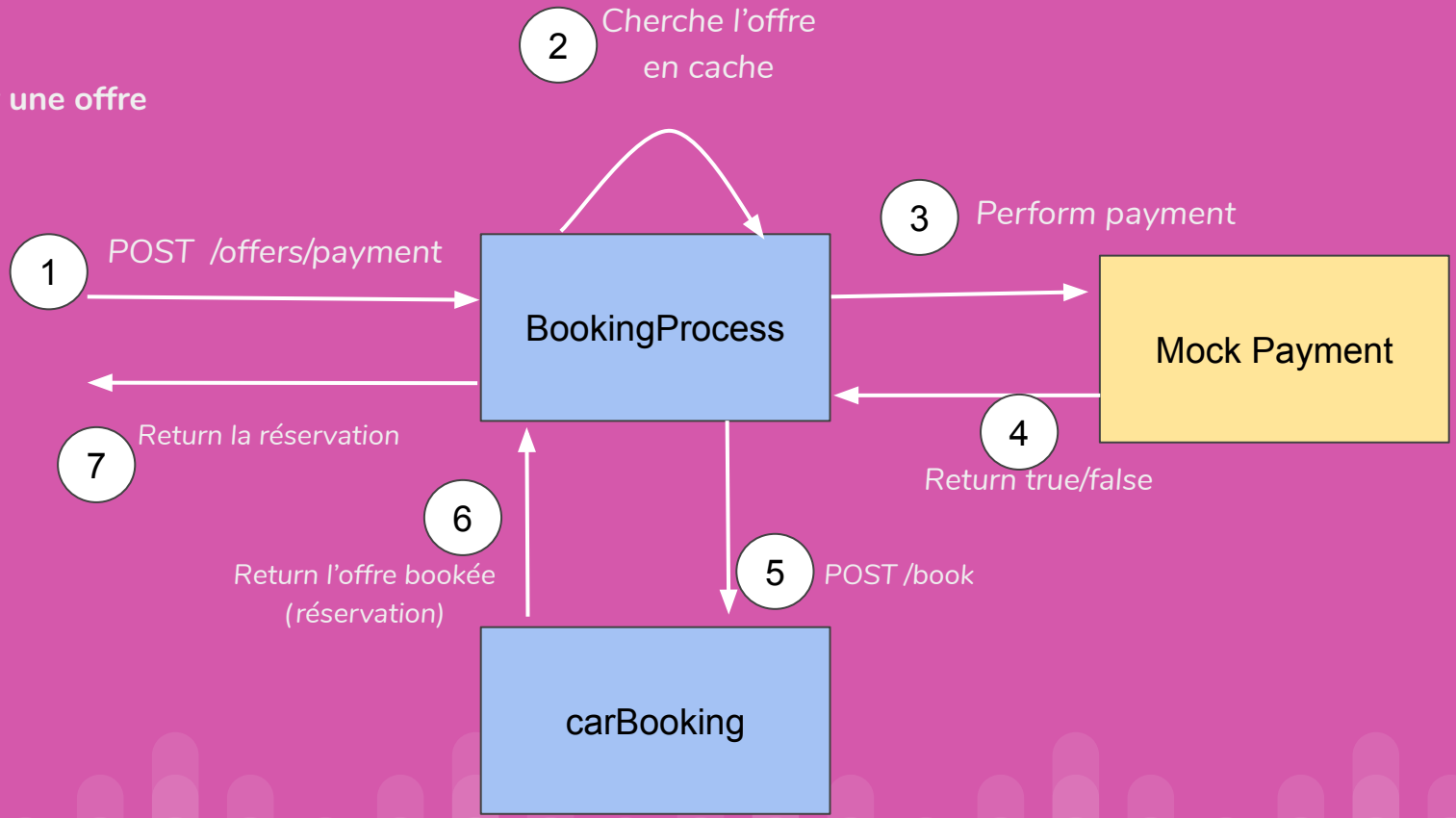
- Un fournisseur
- Une recherche sans résultat
- Seconde recherche avec changement de date
- Réservation

Démonstration

Architecture - système : Workflows



Payer une offre



Communications via HTTP avec un marshalling JSON

Organisation de l'équipe

MASIA Sylvain

- Service BookingProcess

MONTOYA Damien

- Service CarBooking
- Frontend

RIGAUT François

- Service CarSearching
- Service CarLocation

PERES Richard

- CarAvailability
- Dockerization & CircleCI

- 1~ service / personne, sur 1 branche et intégration dev
- Communication régulière pour intégration en équipe
- Intégration rapide dès la première semaine

Perspectives d'amélioration : Architecture

- Archi **modulable** = ajout de composants facile (ex: **Garage**)
- Mise en place d'un **bus** de **communication** (usecases et archi adéquats)
  => et/ou **synchro** de **BD** / **Caching** pour éviter les **components** "**CRUD**"
- **Batch CRON** pour **synchronisation** de toute les **entités** "statics"
  => Node, carType, etc. utilisées en **readonly** par les services
- **Caching** avancé dans **bookingProcess** pour éviter les **calculs inutiles**
- **Kubernetization** possible et facile (mais pas "forcément" utile)

Perspectives d'amélioration : Fonctionnalités

- Ajouter un **coef** de **pertinence** de résultat (fait par les services de recherches)
- Calcul du **prix** => en fonction de la **pertinence** (+ d'autres facteurs)
- Gestion de la **concurrence** par **bookingProcess**
- **Planification** de la **position** et **disponibilité** des cars dans le **temps**
- **Monitoring** des performances et statistiques avec Prometheus par exemple

Questions