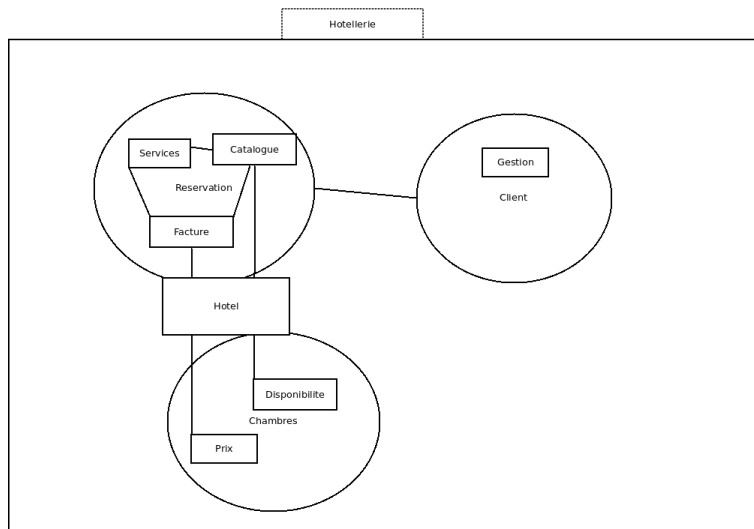


Fonctionnalités Projet

Kevin Alary & Nicolas Yon

DDD(Domain driven design)



Langage

Étant basé sur le web, avec des machines distantes, nous avons opté pour le langage **PHP**. En effet, ce langage semble être tout à fait adapté pour ce type de situation. En ce qui concerne les bases de données, le NoSQL semble être une solution adaptée, puisque moins lourde que le SQL.

Communication

Afin que les services puissent communiquer entre eux, nous avons opté pour l'API REST, qui est plutôt simple et semble bien remplir nos demandes. Aussi nous ne voyons pas l'intérêt de crypter les informations entre les microservices, donc du simple *HTTP* suffira, et non pas du *HTTPS*.

1 Ajout/Suppression d'un client

Cette fonctionnalité est bien évidemment indispensable.

2 Modification du prix

Il serait également intéressant d'avoir la possibilité de pouvoir modifier les prix pour les mêmes raisons que citées plus.

3 État des chambres

Il faudrait savoir quelles chambres sont disponibles. Il faudrait également pouvoir modifier l'état des chambres (libre ou occupée). Il faudrait aussi savoir combien de chambres sont disponibles (par hôtel).

4 Réservation

Il faut bien sûr pouvoir modifier (ajout/suppression) les réservations. On pourrait donc bien sûr pouvoir avoir accès au calendrier.

5 Ajout/suppression de chambres

Il serait intéressant de pouvoir ajouter ou supprimer des chambres si l'hôtel subit des modifications physiques.

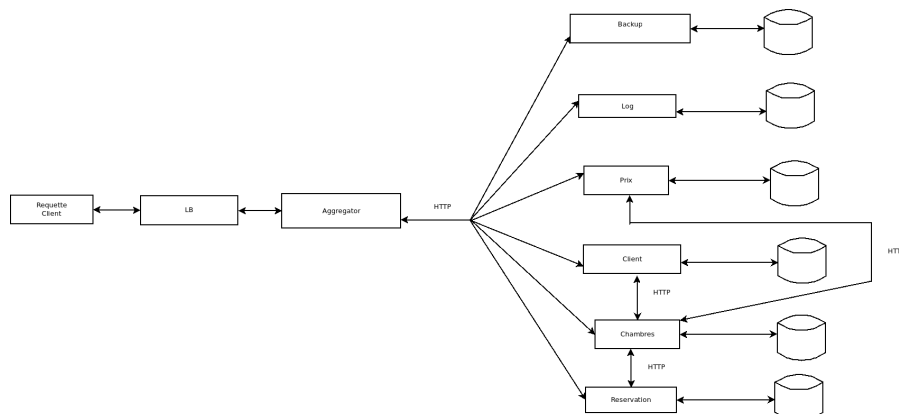
6 Ajout/suppression d'hôtels

Il serait, comme pour les chambres, également intéressant de pouvoir supprimer ou ajouter des hôtels si nécessaire.

7 Mail

Permettrait de pouvoir écrire des mails.

Microservices



Le pattern utilisé serait avec simplement un gateway/aggregator qui permettrait de pouvoir faire appel aux différents services, soit de pouvoir modifier/récupérer des données. Si un service a besoin de faire appel à un autre, il passerait par la gateway. La communication se fera grâce à l'API REST.

1 Backup

Ce service permettrait de pouvoir créer un backup. Le backup servirait avant tout ce qui est donnée (hormis les logs). En effet les logs serviraient plus à pouvoir identifier la source d'un problème, ce qui ne semble pas indispensable en cas de problèmes matériels.

2 Log

Ce service permet d'avoir un système de logs. Pensant faire tourner les serveurs sous linux, les logs seraient sauvegardés sous

`/var/log/<type de service>`

Avec, par exemple, *<type de service>* :

- système
- communication entre les microservices

3 Client

Ce service permettrait d'écrire et de lire dans la base de données des clients.

4 Réservation

Ce service permettrait d'écrire et de lire dans la base de données des réservations.

5 Prix

Puisque les prix fluctuent en fonction du jour, ce service permettrait de récupérer le prix actuel.

6 Récupération de log

Ce service permettrait de pouvoir lire les logs.

7 Chambres

Ce service permettrait de pouvoir lire et écrire dans la base de données des chambres.

Routes

- GET

- /client/lname/{lname}/fname/{fname}/mail/{mail}
Récupère l'id d'un client
lname: string
fname: string
mail: string
retourne: int
- /hotels
Récupère tous les hôtels
retourne: int[]
- /client/id/{id}
Récupère un client spécifique
id: int
retourne: dictionnaire
- /room/id/{id}
Récupère une chambre
id: int
retourne: dictionnaire
- /hotel/id/{id}/rooms
Récupère toutes les chambres d'un hôtel
id: int
retourne: int[]
- /hotel/id/{hotel_id}/room/id/{room_id}
Récupère la chambre d'un hôtel
hotel_id: int
room_id: int
retourne: dictionnaire
- /calendar
Récupère le calendrier
retourne: dictionnaire
- /calendar/year/{year}
Récupère le calendrier mais pour une seule année
retourne: dictionnaire
- /booking/id/{booking_id}/client/{client_id}
Récupère une réservation pour un client
booking_id: int
client_id: int
retourne: dictionnaire
- /booking/day/{day}/month/{month}/year/{year}/nights/{nights}
Récupère les chambres libres pour la date et le nombre de nuits indiqués
day: int
month: int
year: int
nights: int
retourne: int[]

- UPDATE

- /client/id/{id}/fname/{fname}
Met à jour le prénom d'un client
id: int
fname: string
- /client/id/{id}/lname/{lname}
Met à jour le nom de famille d'un client
id: int
lname: string
- /client/id/{id}/mail/{mail}
Met à jour le mail d'un client
id: int
mail: string
- /booking/id/{id}/state/{state}
Met à jour la réservation
id: int
state: int
- /room/id/{id}/roomtype/{rtype_id}
Met à jour une chambre
id: int
rtype_id: int

- POST

- /client/lname/{lname}/fname/{fname}/mail/{mail}
Ajoute un client
lname: string
fname: string
mail: string
- /room/roomtype/{rtype_id}
Ajoute une nouvelle chambre
rtype_id: int
- /booking/day/{day}/month/{month}/year/{year}/nights/{nights}/room/id/{room_id}-
/client/id/{client_id}
Ajoute une réservation
day: int
month: int
year: int
nights: int
room_id: int
client_id: int
- /hotel/id/{id}
ajoute un nouvel hôtel
id: int

- DELETE

- /client/id/{id}
Supprime un client
id: int
- /booking/id/{id}
Supprime une réservation
id: int
- /hotel/id/{id}
Supprime un hôtel
id: int
- /room/id/{id}
Supprime une chambre
id: int