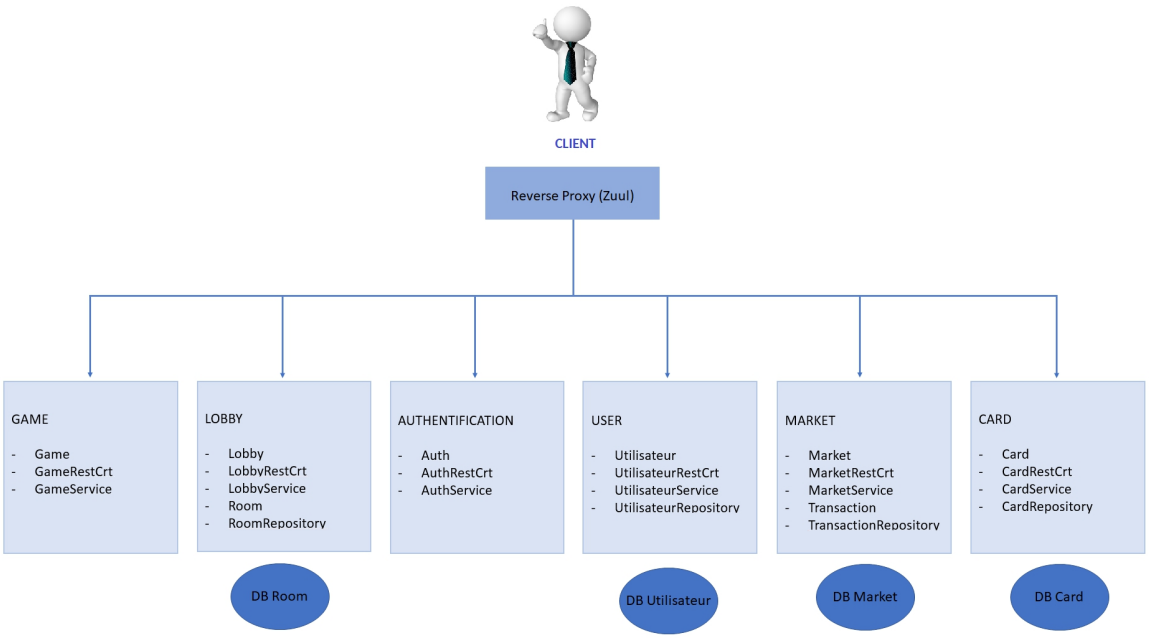
ATELIER III

Architecture du projet:



Avantages/Différences entre les architectures SOA et Microservice :

Les principales différences entre ces deux architectures sont présentées dans le tableau ci-dessous :

	Architecture SOA	Architecture Microservice
Principe de conception	Partage de ressources entre les services	Autonomie de chacun des services
Granularité	Services réutilisables mais peu affinés, faisant office parfois de sous-ensemble complet	Services plus affinés, réalisant une tâche unique et simple, avec une complète indépendance de chacun d'eux
Stockage de données	Partage des données entre différents services	Données isolées pour chacun des microservices
Communication	Partage du mécanisme de communication ESB, susceptible de ralentir le système.	Système de requête-réponse avec appels d'API REST fondés sur le protocole HTTP
Taille et portée	Pertinent à l'échelle d'une entreprise, du fait de la scalabilité permise par la réutilisabilité des services	Pertinent à l'échelle d'une application
Couplage et cohésion	Faible cohésion et couplage assez fort du fait du partage de	Forte cohésion par le caractère minimal d'un microservice, et

	données	faible couplage par l'indépendance de chacun d'eux
Déploiement	Simple mais fastidieux car nécessite de redéployer toute l'application lors d'un ajout de service, car ceux-ci sont fortement couplés entre eux	Facile et rapide du fait de l'indépendance des microservices car chacun d'entre eux peut être déployé seul. Cependant, plus complexe lorsqu'il s'agit de déployer l'ensemble de l'application.

L'architecture Microservice présente de nombreux avantages, parmi lesquels :

- **Haute évolutivité**, du fait que chaque microservice peut être échelonné indépendamment
- **Résilience,** ce qui signifie qu'un microservice qui tombe en panne n'entraîne pas la panne de l'ensemble de l'application, car il est indépendant des autres
- **Accessibilité en terme de développement,** car chaque microservice réalise une tâche métier simple, et donc facile à comprendre. Cela a pour conséquence des cycles de développement plus rapides
- **Ouverture,** car chaque microservice peut-être développé dans le langage le plus approprié.

Couverture des tests effectués :

Nous avons effectué pour chaque microservice un test via l'outil Postman pour tester la validité de nos URLs et du résultat des requêtes individuelles. Nous avons ensuite effectué des tests mettant en jeu plusieurs microservices à la fois, afin de s'assurer de la bonne communication au sein de notre application. Nous avons pour cela utilisé le reverse proxy Zuul. Nous avons ensuite reproduit ces tests via navigateur.

En parallèle, des tests unitaires et des tests d'intégration ont été effectués pour la plupart des microservices afin de s'assurer du bon fonctionnement des méthodes et classes et de la non-régression du système.

Nous avons ensuite fait un test global mettant en jeu tous les microservices et le front-end de notre application.

L'analyse Sonar n'a pas pu être effectuée.

Sources :

- https://www.talend.com/fr/resources/microservices-vs-soa/

- https://www.ibm.com/cloud/blog/soa-vs-microservices

- https://nirmata.com/2015/02/02/microservices-five-architectural-constraints/

- https://docs.oracle.com/fr/solutions/learn-architect-microservice/index.html#GUID-BDCEFE30-C883-45D5-B2E6-325C241388A5

- https://www.redhat.com/fr/topics/microservices/what-are-microservices