ECOLE MAROCAINE DES SCIENCES DE L'INGENIEUR
Membre de HONORIS UNITED UNIVERSITIES

Projet de class

Conception et réalisation d'une application de Bloc d'opération

IIR5 – Group 4

Ingénierie Informatique et Réseaux

Réalisé par :

Nom de l'étudiant : **Anass Hajjouj – Hamza Maataoui**

Encadré par : **M. Mohamed Lachgar**

Remerciements

Nous tenons à exprimer nos sincères remerciements à notre professeur **M.LCHGER** pour son précieux soutien et ses conseils éclairés tout au long de la réalisation de notre Projet de class. Sa disponibilité, son expertise et son dévouement ont grandement contribué à la réussite de ce projet.

Ses retours constructifs et ses orientations ont été d'une importance capitale pour affiner nos idées, surmonter les défis et atteindre les objectifs fixés. Nous sommes reconnaissants pour l'inspiration qu'il a apportée à notre travail et pour avoir partagé son savoir d'une manière qui a profondément enrichi notre expérience académique collective.

Merci infiniment, Professeur **M.LCHGER**, pour avoir été un guide exceptionnel et pour avoir joué un rôle crucial dans la concrétisation de ce projet. Votre mentorat a été une source d'inspiration et a grandement contribué à notre développement professionnel en tant qu'équipe.

Table de matière

Partie I : Contexte générale du projet

Introduction :

Dans un contexte où les avancées technologiques transforment la façon dont les services sont proposés et consommés, notre projet prend place avec pour objectif de répondre à un besoin croissant dans le secteur de la réservation d'hôtels. Cette introduction pose les bases du contexte général de notre initiative, mettant en lumière les défis et les opportunités inhérents à ce domaine dynamique.

I. Micro-services

1.1Introduction :

Les micro-services sont une approche architecturale dans le développement logiciel où une application monolithique traditionnelle est décomposée en un ensemble de services autonomes, indépendants et spécialisés. Chaque micro-services représente une unité fonctionnelle distincte, déployée et gérée de manière indépendante. L'architecture micro-services permet une meilleure résilience, une évolutivité aisée, ainsi qu'une agilité accrue dans le développement, le déploiement et la maintenance des applications.

1.2L’importance :

L'importance de l'architecture micro-services repose sur plusieurs avantages clés qui répondent aux défis modernes du développement logiciel et de la gestion des systèmes informatiques. Voici quelques points cruciaux qui soulignent l'importance de l'architecture micro-services:

- **Scalabilité et Flexibilité** : Les micro-services permettent une scalabilité granulaire, où chaque service peut être développé, déployé, et évolué indépendamment des autres. Cela offre une flexibilité accrue pour adapter les composants du système en fonction des besoins spécifiques.

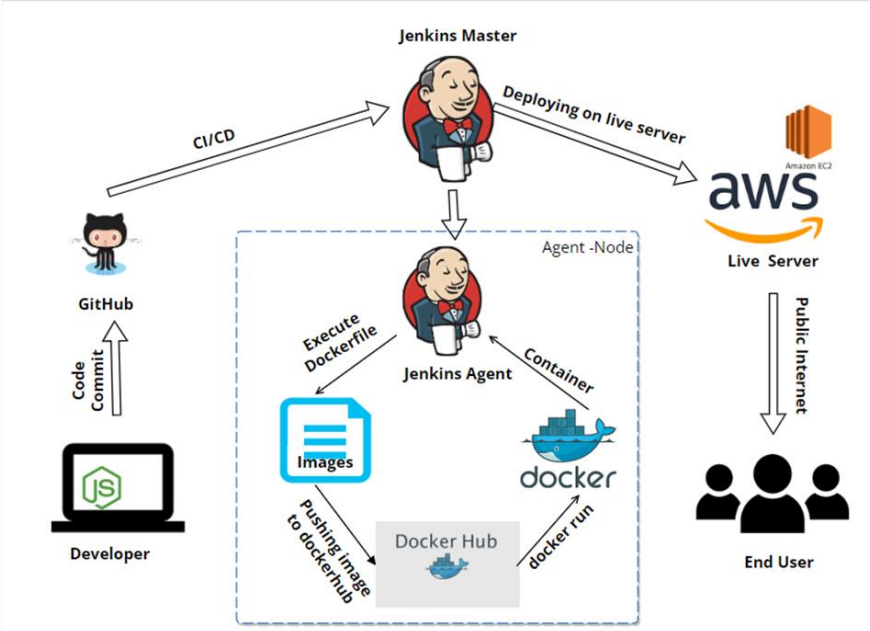
- **Facilité de Maintenance :** En découpant une application en petits services, la maintenance devient plus aisée. Les mises à jour, les correctifs, et les modifications peuvent être effectués

sur un service sans impacter l'ensemble de l'application. Cela facilite également la localisation et la résolution des problèmes.

- **Développement Agile** : L'architecture micro-services favorise les méthodologies de développement agile. Des équipes autonomes peuvent travailler sur des services spécifiques, accélérant le processus de développement, facilitant la collaboration, et permettant des itérations plus rapides.
- **Adaptabilité aux Technologies Cloud** : L'architecture microservices est bien adaptée aux environnements cloud, permettant une distribution efficace des services et une utilisation optimale des ressources cloud, favorisant ainsi la scalabilité horizontale.

Partie II : Architecture Micro-services

2.1 Architecture Micro-service :



le développeur utilise Git/GitHub pour gérer le code source, Jenkins pour automatiser les processus CI/CD, et Docker pour créer des conteneurs qui encapsulent les microservices. Ces outils sont souvent intégrés pour fournir un flux de travail efficace et reproductible, permettant le développement, les tests et le déploiement continus des microservices.

2.2 Mécanismes de communication :

Dans le cadre de notre architecture microservices, les mécanismes de communication jouent un rôle crucial dans la création d'un écosystème cohérent et interconnecté. La communication entre les différents microservices est réalisée principalement via des mécanismes basés sur les API RESTful. Cette approche a été délibérément choisie pour favoriser l'indépendance entre les services et faciliter leur intégration harmonieuse.

Les API RESTful offrent une interface standardisée et flexible pour la communication entre les microservices. Elles sont basées sur les principes du protocole HTTP, permettant une interaction stateless et simplifiant la gestion des échanges de données. Ce choix permet aux services de communiquer de manière uniforme, qu'ils soient développés en utilisant Spring, Angular, ou d'autres technologies.

Consul, en tant qu'outil central de notre architecture, joue un rôle essentiel dans la facilitation de la communication entre microservices. Il agit comme un registre de services, permettant la découverte dynamique des services disponibles dans l'écosystème. Lorsqu'un microservice doit interagir avec un autre, Consul offre un mécanisme de découverte efficace, garantissant ainsi une connectivité fluide sans dépendance rigide sur des adresses IP statiques.

La combinaison de l'approche RESTful et de l'utilisation de Consul crée un environnement où chaque microservice peut évoluer indépendamment tout en restant connecté au reste du système. Cette flexibilité dans la communication entre les microservices contribue à la résilience et à la scalabilité de notre architecture, permettant une gestion efficace des interactions complexes au sein de l'écosystème.

Partie IV: Conteneurisation avec Docker

4.1 Implémentation et avantages :

Implémentation de Docker dans notre Architecture Microservices :

La conteneurisation avec Docker a été soigneusement intégrée dans notre projet, offrant une solution efficace pour encapsuler chaque microservice. Chaque composant du système, du backend développé avec Spring aux services frontend en Angular, est maintenant encapsulé dans des conteneurs Docker autonomes.

L'implémentation a suivi un processus structuré, où chaque microservice a été dockerisé individuellement. Les fichiers Dockerfile ont été créés avec une attention particulière aux dépendances et aux configurations spécifiques à chaque service. L'utilisation de Docker Compose a été privilégiée pour orchestrer le déploiement simultané de plusieurs conteneurs, créant ainsi une infrastructure cohérente et facilement reproductible.

Avantages de la Conteneurisation avec Docker : simplification du déploiement, Docker permet le déploiement et la mise à l'échelle rapide des microservices, offrant ainsi une solution efficace pour répondre aux demandes croissantes de trafic ou de nouvelles fonctionnalités

Partie V: CI/CD avec Jenkins

5.1 Configuration et bénéfices pour la qualité du code :

La mise en place de Jenkins dans notre projet pour automatiser le processus de CI/CD a considérablement amélioré notre efficacité de développement et garantit des déploiements fiables. Voici une vue d'ensemble du processus et de la configuration que nous avons adoptés :

Processus de CI/CD avec Jenkins :

Le processus de CI/CD est déclenché automatiquement à chaque commit sur le référentiel de code source. Cela garantit une intégration continue à chaque changement, facilitant la détection rapide d'éventuels problèmes.

Partie VII : Conclusion

6.1 Résumé des accomplissements :

Notre projet a atteint avec succès ses objectifs en mettant en place une architecture de microservices moderne et innovante. En combinant des technologies de pointe telles que Consul, Spring, Angular, et

Jenkins, nous avons créé un écosystème informatique agile et hautement efficace. Les accomplissements clés de ce projet peuvent être résumés comme suit:

Intégration de Docker pour la Conteneurisation : L'utilisation de Docker a simplifié le déploiement des microservices, offrant une encapsulation efficace avec des avantages en termes de portabilité, de gestion des dépendances, et de reproductibilité de l'environnement.