Atelier 02

Activités de recherche :

Tâche 1 : Faire une recherche internet sur la différence entre les applications monolithiques et les applications micro services. Déterminer pour chacun les avantages et les inconvénients.

Applications monolithiques :
Les applications monolithiques sont des systèmes logiciels dont toutes les fonctionnalités sont développées dans une seule base de code, souvent regroupées dans une seule unité déployable.

Applications microservices:
Les microservices décomposent une application en une série de petits services indépendants qui sont développés, déployés et mis à l'échelle séparément.

	Les avantages	**Les inconvénients**
Les applications monolithiques	Simplicité, Moins de complexité, Communication interne rapide	Scalabilité limitée, Difficulté de mise à jour, Dépendance
Les applications micro-services	Scalabilité, Flexibilité, Développement parallèle, Résilience	Complexité de gestion, Latence réseau, Surcoût

Tâche 2 : Faire une recherche internet sur les entreprises qui ont des applications qui se base sur l'architecture microservice.

Les entreprises qui utilisent l'architecture microservices incluent :

- **Netflix** : Utilise des microservices pour ses applications de diffusion vidéo, permettant une haute disponibilité et la scalabilité.
- **Uber** : Implémente des microservices pour gérer des transactions complexes et de haute fréquence dans un environnement à grande échelle.
- **Amazon** : L'architecture microservices est utilisée pour gérer les différentes fonctionnalités de son site web, ses services de cloud computing (AWS), et ses plateformes logistiques.
- **Spotify** : Utilise des microservices pour gérer sa plateforme musicale, facilitant l'extensibilité et la mise à jour continue.

Tâche 3 : Quel est la différence entre la virtualisation et la conteneurisation ?

- **Virtualisation** :
    - Permet de créer plusieurs machines virtuelles (VM) sur un seul serveur physique, chaque VM ayant son propre système d'exploitation (OS).
    - Plus lourd en termes de ressources et de gestion, car chaque VM doit fonctionner avec un OS complet.
- **Conteneurisation** :

- Permet d'exécuter des applications dans des "conteneurs" isolés, partageant le même noyau OS.
- Moins lourd que la virtualisation car il n'y a pas de besoin de gérer des OS distincts pour chaque conteneur. Les conteneurs sont plus rapides à démarrer et plus efficaces en termes de ressources.

Tâche 4 : Faire une recherche internet sur les conteneurs des applications informatiques qui existent dans le marché. Spécifiez le conteneur le plus utilise dans le marché.

Les conteneurs les plus utilisés dans le marché sont :

- **Docker** : Le conteneur le plus populaire et largement adopté. Docker permet de packager des applications avec toutes leurs dépendances dans des conteneurs pour faciliter le déploiement et la gestion.
- **Kubernetes** : Bien que Kubernetes soit un outil d'orchestration, il est étroitement associé à Docker pour la gestion des conteneurs à grande échelle.

Tache 5 : Quel est le rôle de l'orchestration des conteneurs ?

L'orchestration des conteneurs permet de gérer et d'automatiser le déploiement, la mise à l'échelle et la gestion de plusieurs conteneurs. L'orchestration est utilisée pour :

- **Automatiser le déploiement** et la gestion des applications dans des environnements distribués.
- **Mettre à l'échelle dynamiquement** les applications en fonction des demandes de ressources.
- **Assurer la résilience** en réorientant les conteneurs échoués vers d'autres hôtes disponibles.
- **Gérer les dépendances** entre les conteneurs.

Tache 6 : Citer quelques outils qui vont permettre l'orchestration des conteneurs.

Quelques outils d'orchestration de conteneurs sont :

- **Kubernetes** : L'outil d'orchestration de conteneurs le plus populaire, permettant la gestion et l'automatisation du déploiement des applications conteneurisées.
- **Docker Swarm** : Outil d'orchestration intégré à Docker, plus simple mais avec des fonctionnalités limitées par rapport à Kubernetes.
- **Apache Mesos** : Un autre système de gestion de clusters, plus flexible mais également plus complexe que Kubernetes.
- **Amazon ECS (Elastic Container Service)** : Service géré d'AWS pour l'orchestration des conteneurs.

Tache 7 : Quels sont les phases du DevOps ?

Les phases du DevOps incluent :

1. **Planification** : Définir les exigences et planifier le développement.
2. **Développement** : Écrire le code, implémenter les fonctionnalités.
3. **Intégration continue (CI)** : Tester et intégrer régulièrement les changements de code dans un dépôt partagé.
4. **Déploiement continu (CD)** : Automatiser le déploiement des applications en production.
5. **Exploitation** : Surveillance des systèmes en production, résolution des incidents.
6. **Rétroaction** : Analyser les performances, recueillir les commentaires pour améliorer les processus.

Tache 8 : Quels sont les problèmes que DevOps essaye de résoudre ?

DevOps vise à résoudre plusieurs problèmes :

- **Manque de collaboration** entre les équipes de développement et d'opérations.
- **Lenteur de livraison des fonctionnalités** à cause de processus manuels.
- **Problèmes de qualité** du code qui entrent en production.
- **Difficulté à maintenir la cohérence** des environnements de développement et de production.

Tache 9 : Quels sont les fondements du DevOps ?

Les fondements du DevOps sont :

- **Collaboration** entre les équipes de développement et d'opérations.
- **Automatisation** des processus de tests, de déploiement et de gestion des infrastructures.
- **Amélioration continue** des pratiques de développement et d'exploitation.
- **Responsabilisation** des équipes pour la qualité et la performance des applications en production.

Tache 10 : Citer les outils utilisés dans chaque phase du DevOps ?

Voici des exemples d'outils pour chaque phase du DevOps :

- **Planification** : Jira, Trello
- **Développement** : GitHub, GitLab, Bitbucket
- **CI/CD** : Jenkins, GitLab CI, Travis CI
- **Déploiement** : Docker, Kubernetes, Terraform

- **Exploitation** : Nagios, Prometheus, Grafana
- **Rétroaction** : ELK Stack (Elasticsearch, Logstash, Kibana), New Relic

Tache 11 : Quels sont les avantages des applications cloud-native ?

Les avantages des applications cloud-native incluent :

- **Scalabilité** : Les applications peuvent être mises à l'échelle facilement en fonction des besoins.
- **Résilience** : Elles sont conçues pour être tolérantes aux pannes et pour se rétablir rapidement.
- **Flexibilité** : Utilisation de microservices et d'architectures décentralisées.
- **Déploiement rapide** : Automatisation des déploiements grâce aux pratiques de CI/CD.
- **Optimisation des ressources** : Utilisation efficace des ressources du cloud avec une gestion fine des conteneurs et des services.