TP DevOps

Membres du groupe:

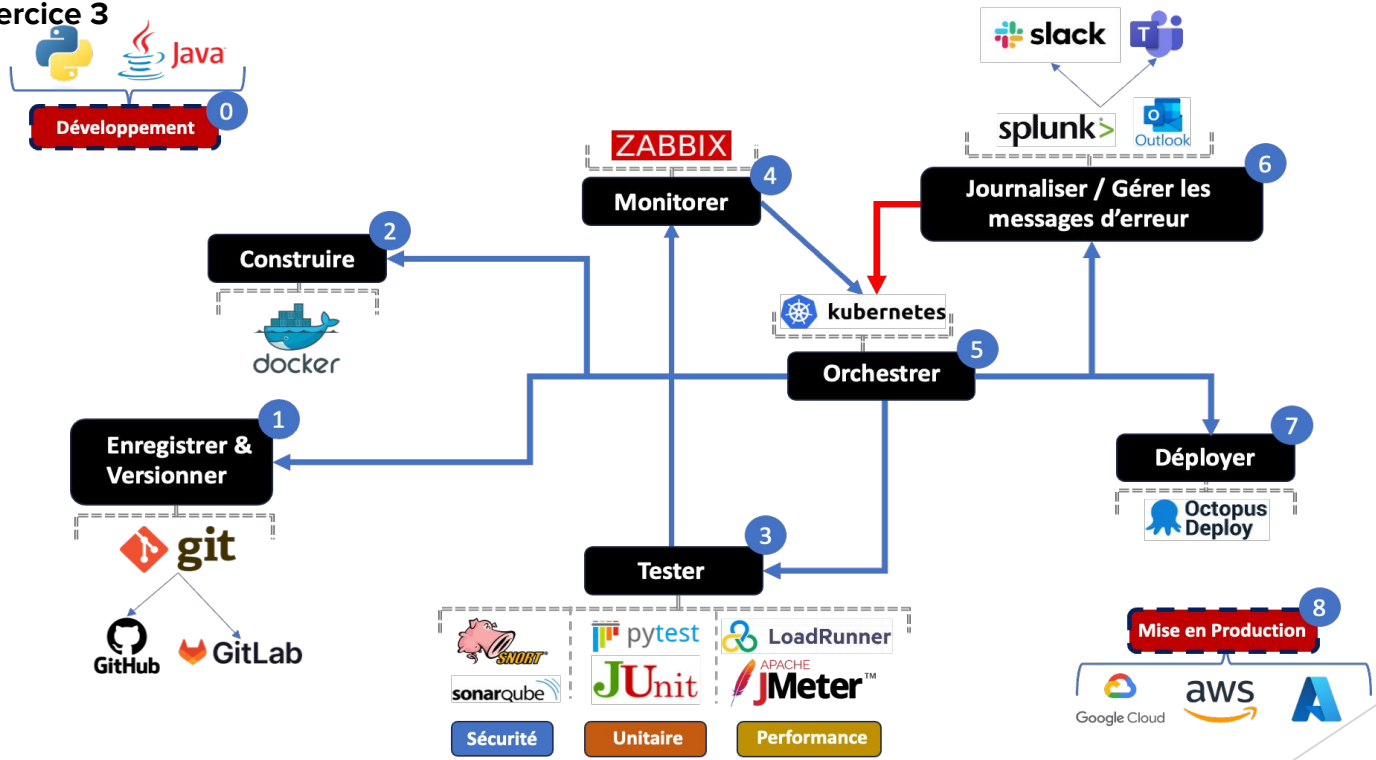
- Charles-Albert Kotto
- Ehian Christian
- François Collu
- Abdelmoutakabir Oumaima
- Franck Mevengue
- Job Daryl Massande

Sous la supervision de M. Xavier Van Lindt

Architecturer une toolchain de Transition :

Pipeline de déploiement : Cas de développement d'application

Exercice 3



Légende :

- Chaîne courante
- Rétropropagation (Cas d'erreur)

Expliquer le rôle de chaque outil :
Outils de déploiement

Exercice 2

0. **Développement** :

C'est à ce stade que l'essentiel du travail de codage, de test et d'optimisation est effectué pour créer par exemple une application fonctionnelle et robuste avec des langages comme **Java** et **Python**.

- **Outils** : Java, Python

1. **Enregistrer & Versionner** :

Les développeurs enregistrent leur code source dans un système de gestion de versions tel que **Git**. Cela permet de suivre les modifications, de gérer les branches de développement et de collaborer efficacement.

- **Outils** : Git, Github, Gitlab

2. **Construire** :

Cette étape garantit que l'application est prête pour les tests et le déploiement ultérieurs.Le code source enregistré est récupéré et utilisé pour construire des images de conteneurs **Docker**. Ces images contiennent l'application, ses dépendances et la configuration nécessaire pour s'exécuter de manière cohérente sur différentes plates-formes.

- **Outil** : Docker

Expliquer le rôle de chaque outil :
Outils de déploiement

Exercice 2

3. **Tester** :

L'étape de test garantit la qualité, la fiabilité et la performance de l'application avant son déploiement, minimisant ainsi les erreurs et les problèmes potentiels dans l'environnement de production.

L'application est soumise à divers tests :

- **Tests de sécurité :** Effectuer une analyse statique du code pour détecter les violations de règles de codage et les vulnérabilités.

- **Outil** : SonarQube

- **Tests unitaires :** Effectuer des tests unitaires pour valider le comportement des composants individuels de l'application.

- **Outils** : Pytest, JUnit

- **Tests de performance :** Effectuer des tests de charge et de performance pour évaluer la réactivité et la stabilité de l'application sous différentes charges.

- **Outils** : LoadRunner, JMeter

Expliquer le rôle de chaque outil :
Outils de déploiement

Exercice 2

4. **Monitorer** :

L'étape de monitoring consiste à surveiller en temps réel les performances et la disponibilité de l'application. On utilise **Zabbix** pour surveiller les performances de l'application en temps réel. Il collecte des données sur l'utilisation des ressources, les temps de réponse et d'autres métriques importantes, permettant aux équipes d'opérations informatiques d'identifier et de résoudre rapidement les problèmes.

- **Outil** : Zabbix

5. **Orchestrer** :

L'étape d'orchestration consiste à coordonner le déploiement, la gestion et l'évolutivité des conteneurs de manière automatisée et efficace. **Kubernetes** assure la mise à l'échelle automatique, la gestion des versions, et garantit que l'application fonctionne de manière fiable, indépendamment de l'environnement.

- **Outil** : Kubernetes

Expliquer le rôle de chaque outil :
Outils de déploiement

Exercice 2

6. **Journaliser / Gérer les messages d'erreur** :

Implique l'enregistrement et la gestion des journaux et des messages d'erreur générés par l'application.Cela inclut l'utilisation d'outils tels qu'**Outlook** et **Splunk** pour recevoir des notifications d'erreurs par e-mail, ainsi que l'utilisation de Splunk pour analyser et gérer les journaux d'application et d'infrastructure.

- **Outils** : Splunk, Outlook

Si des erreurs sont détectées pendant la surveillance des journaux , elles sont analysées dans **Splunk** et les développeurs sont notifiés à travers des outils comme **Slack** ou **Teams**. Si une solution est trouvée, les instructions de résolution sont envoyées à **Kubernetes** (Étape 5). Une fois que tous les tests, y compris la gestion des erreurs, sont réussis, l'application est déployée sur les plates-formes de cloud computing à l'aide d'**Octopus Deploy** (Étape 7). Ce processus garantit que l'application est surveillée en continu, que les erreurs sont résolues efficacement et que l'application est déployée en toute confiance.

- **Outils** : Kubernetes, Slack, Teams

Expliquer le rôle de chaque outil :
Outils de déploiement

Exercice 2

7. **Déployer** :

L'étape de déploiement consiste à mettre en place l'application dans l'environnement de production après avoir passé avec succès les tests. **Octopus Deploy** automatise le déploiement de l'application dans divers environnements (développement, test, production). Il gère également les configurations spécifiques à chaque environnement pour assurer la cohérence entre les déploiements.

- **Outils** : Octopus Deploy

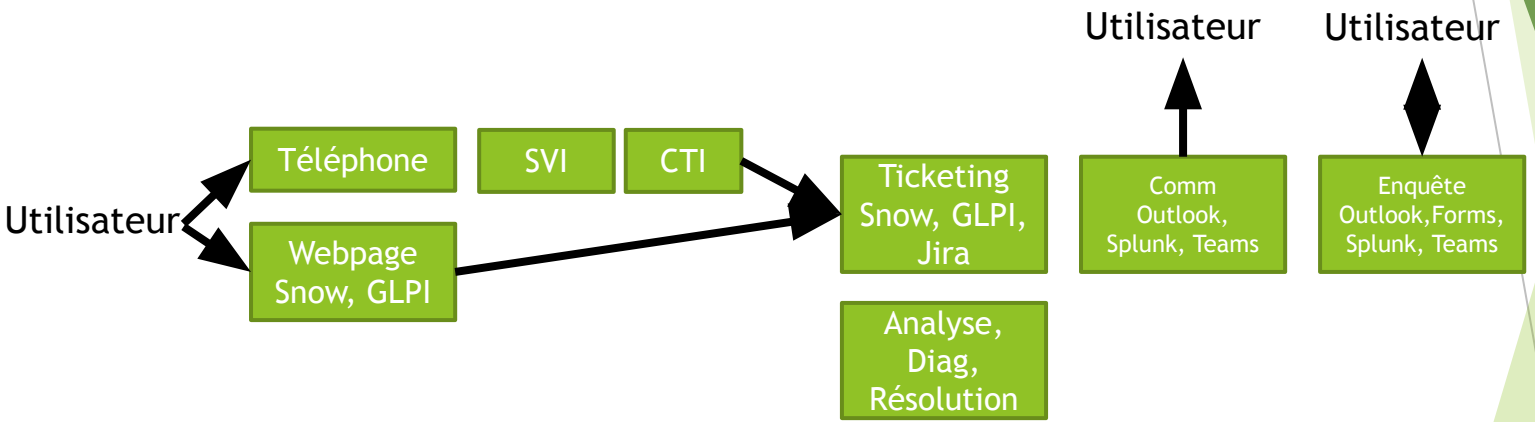
8. **Mise en production** :

L'application est finalement déployée dans l'environnement de production, qui peut être un cloud public tel que **Google Cloud**, **AWS (Amazon Web Services)** ou **Azure (Microsoft Azure)**. Ces plates-formes permettant aux utilisateurs finaux d'accéder à celle-ci via Internet. La mise en production garantit que l'application est accessible au public ou aux utilisateurs autorisés, offrant ainsi une expérience utilisateur optimale et stable.

- **Outils** : GCP, AWS, Microsoft Azure

Architecturer une toolchain de RUN :
Outils d'exploitation : incident utilisateur

Exercice 3



Scrum en mode DevOps

Personnes : rôles

Exercice 5

En DevOps, trois rôles principaux se distinguent:

- Le **Product Owner**: Par sa connaissance et sa compréhension très approfondie du produit, il gère les besoins des parties prenantes, tout en optimisant le travail de l'équipe *Dev*. En faisant ainsi, il agit sur la valeur du produit.

- Le **Scrum Master:** À l'aide de ses qualités humaines, il accompagne chaque membre de l'équipe Scrum pour les aider à affronter les obstacles pouvant être rencontrés. Il sert également d'intermédiaire entre le Product Owner et l'équipe de développement. Garant de la méthode Scrum, il faut le voir comme un *coach*, et non un *manager*.

- **L'Équipe de développement:** Lors de chaque sprint, cette équipe regroupe l'ensemble du personnel chargé de la réalisation du produit. Les itérations nécessaires à la réalisation du produit sont décidées par ses membres.

Scrum en mode DevOps

Technologie : décrire l'offre de service des Ops -> Dev (IaC)

Exercice 5

Les équipes Ops (exploitation) travaillent étroitement avec les équipes Dev (développement) afin de garantir l'amélioration et la livraison continue. Dans le domaine de l'infrastructure en tant que code (IaC), l'exemple ci-dessous peut être cité:

- Dans les **modèles de déploiement pour les pipelines CI/CD**, l'équipe Ops peut fournir des infrastructures Iac à l'équipe Dev, afin d'assurer la livraison continue, ainsi que l'intégration continue. L'équipe Ops peut alors utiliser des outils comme *Ansible* ou *Terraform*, tandis que l'équipe Dev peut utiliser *Jenkins* ou *TravisCI*.
- Le schéma ci-contre illustre les tâches

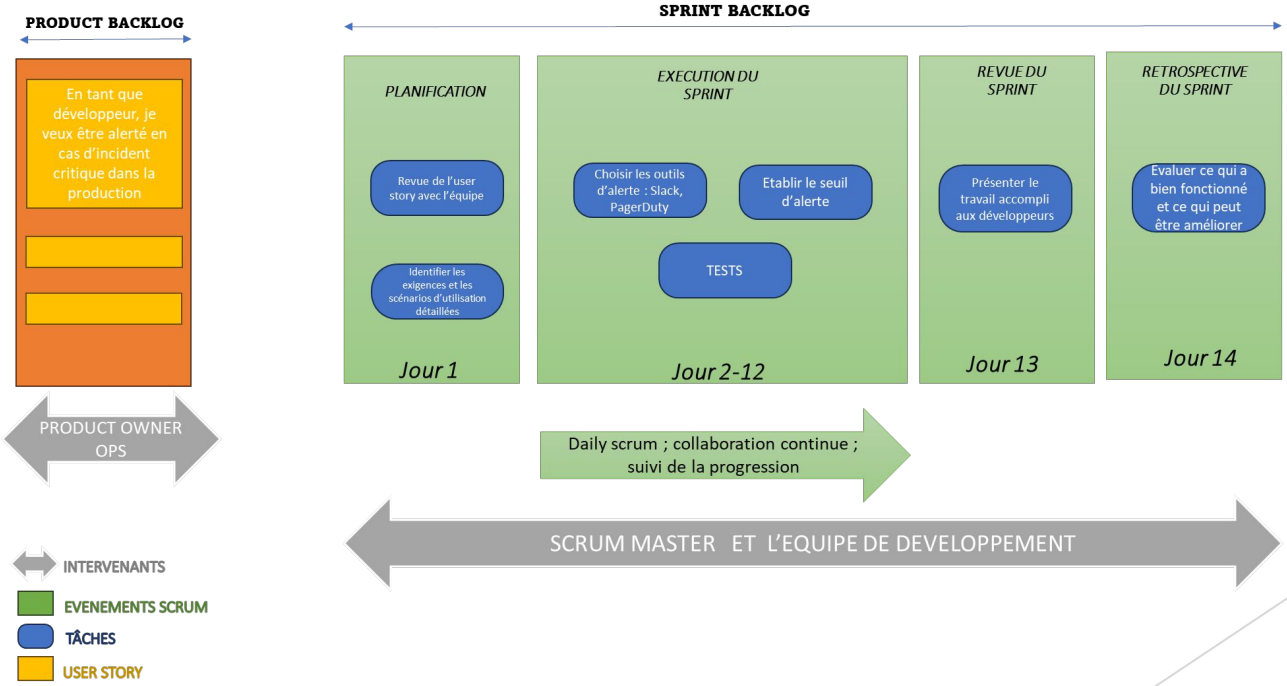
  des équipes Dev et Ops.

Scrum en mode DevOps

User stories, Product et Sprint backlogs

Exercice 5

Les user stories du Product backlog sont transformés en tâches dans le Sprint backlog(liste d'éléments du Product backlog sélectionnés pour être développés) en mode DevOps. Illustrons le par cet exemple:



FIN