

Royaume du Maroc

Agence nationale de réglementation des télécommunications



PROJET DE FIN D'ÉTUDES

pour l'obtention de diplôme d'ingénieur d'Etat

Ingénierie des Systèmes Ubiquitaires et Distribués – Cloud et IoT (SUD)

Intégration/Livraison continues pour Arkevia

AZENNOUD Khalil

Membres du jury :

M. BENNANI Ouassim (encadrant externe)

M. DAHCHOUR Mohamed (encadrant interne)

M. IBN EL HAJ El Hassan (examinateur)

M. SOUISSI Omar (examinateur)

CEGEDIM

2eme etage, Arribat Center, imm.DetE
Av. Omar Ibn Al Khattab ,Rabat 10010

Promotion : 2020/2021

Table des matières

| | | |
|----------|---|-----------|
| 1 | Introduction générale | 10 |
| 2 | Contexte général du projet | 11 |
| 2.1 | Présentation de CEGEDIM | 11 |
| 2.1.1 | Le groupe Cegedim | 11 |
| 2.1.2 | Domaines d'activités | 12 |
| 2.1.3 | Les clients du groupe Cegedim | 13 |
| 2.1.4 | Cegedim en chiffres | 13 |
| 2.2 | CEGEDIM MAROC | 15 |
| 2.2.1 | Présentation de Cegedim SRH | 15 |
| 2.2.2 | Missions CEGEDIM SRH | 16 |
| 2.2.3 | Organisation du Groupe | 16 |
| 3 | Cadre du Projet | 19 |
| 3.1 | Etude de l'existant | 19 |
| 3.2 | Problématique | 22 |
| 3.3 | Objectif du Projet | 22 |
| 3.4 | Comment démarrer une transformation DevOps? | 24 |
| 3.4.1 | Communication, processus et méthodes agiles : | 25 |
| 3.4.2 | Industrialisation des processus de livraison | 26 |
| 4 | Analyse et spécifications des besoins | 28 |
| 4.1 | Démarche de l'étude comparative | 28 |
| 4.2 | Outils de gestion de code source | 28 |
| 4.2.1 | Choix des critères | 29 |
| 4.2.2 | Analyse | 29 |
| 4.2.3 | Comparaison et synthèse | 31 |
| 4.3 | Outils d'intégration continue | 32 |
| 4.3.1 | Choix des critères | 32 |
| 4.3.2 | Analyse | 32 |
| 4.3.3 | Comparaison et synthèse | 33 |
| 4.4 | outils de déploiement continu | 34 |
| 4.5 | outils d'analyse des logs | 34 |
| 4.5.1 | Elasticsearch | 35 |
| 4.5.2 | Fluent Bit | 35 |
| 4.5.3 | Kibana | 36 |
| 4.6 | Autres Outils | 36 |
| 5 | Réalisation | 38 |
| 5.1 | Environnement du travail | 38 |
| 5.1.1 | IDE : IntelliJ IDEA | 38 |
| 5.1.2 | WildFly | 38 |
| 5.1.3 | Git | 39 |

| | | |
|----------|---|-----------|
| 5.1.4 | SonarQube | 39 |
| 5.2 | Implémentation de la chaine CI/CD | 40 |
| 5.2.1 | Configuration de Jenkins | 40 |
| 5.2.2 | Jenkinsfile | 42 |
| 5.3 | Implémentation du système de gestion des logs | 49 |
| 5.4 | Conclusion | 50 |
| 6 | Conclusion Générale | 51 |
| 7 | Annexes | 52 |
| 7.1 | Implémentation du Jenkinsfile | 52 |
| 7.2 | Structure des Helm Charts | 52 |
| 8 | Bibliographie Webographie | 54 |

Table des figures

| | | |
|----|--|----|
| 1 | Répartition de Cegedim | 11 |
| 2 | Répartition du domaine d'activité de Cegedim | 12 |
| 3 | Clients du groupe Cegedim | 13 |
| 4 | Evolution de chiffre d'affaires de Cegedim | 14 |
| 5 | Détails sur le chiffre d'affaires | 14 |
| 6 | Logo CEGEDIM SRH | 15 |
| 7 | Organigramme du Groupe Cegedim SRH | 16 |
| 8 | Département RD - Cegedim SRH | 17 |
| 9 | Vue globale du système | 20 |
| 10 | Les dépendances entre les modules | 21 |
| 11 | Existant | 22 |
| 12 | Pipeline Intégration / Déploiement Continu | 23 |
| 13 | Dashboard des Logs | 23 |
| 14 | Collaboration Dev et Ops | 24 |
| 15 | Communication agile | 25 |
| 16 | Logo GitLab | 29 |
| 17 | Logo BitBucket | 30 |
| 18 | Logo Github | 30 |
| 19 | Récapitulatif de comparaison des outils de gestion de code source | 31 |
| 20 | Récapitulatif de comparaison des outils d'intégration continue | 33 |
| 21 | Logo Kubernetes et Docker | 34 |
| 22 | Logo Elasticsearch | 35 |
| 23 | Logo Fluent Bit | 35 |
| 24 | Logo Kibana | 36 |
| 25 | Logo Rancher | 36 |
| 26 | Logo Helm | 37 |
| 27 | Logo IntelliJ IDEA | 38 |
| 28 | Logo WildFly | 38 |
| 29 | Logo Git | 39 |
| 30 | Logo SonarQube | 39 |
| 31 | Génération du Token sur GitLab | 41 |
| 32 | Configuration de GitLab sur Jenkins | 41 |
| 33 | Configuration du Projet sur Jenkins | 42 |
| 34 | Génération du WAR au niveau du Jenkins Agent | 43 |
| 35 | Résultats de l'analyse du Code avec SonarQube | 43 |
| 36 | Logo Junit | 44 |
| 37 | Logo Mockito | 44 |
| 38 | Depot du WAR sur Jfrog Artifactory | 45 |
| 39 | Depot de l'image Docker sur Jfrog Artifactory | 45 |
| 40 | Notification par mail au git committer pour le succès du job Jenkins | 46 |
| 41 | Notification par mail au git committer pour le fail du job Jenkins | 46 |
| 42 | Chaine Intégration Continue | 47 |
| 43 | Déploiement Continu (Helm/Kubernetes) | 48 |

| | | |
|----|--|----|
| 44 | Logging (EFK stack) | 49 |
| 45 | Dashboard des logs capturés par Fluent-bit | 49 |
| 46 | Architecture Finale | 50 |
| 47 | Exécution des différents Jobs cités au niveau du Jenkinsfile | 52 |
| 48 | Exemple structure Helm Chart de Wordpress | 53 |

Remerciements

Au terme de ce travail, je tiens à remercier chaleureusement M.Bennani Ouassim , Ingénieur Etudes et Développement pour m'avoir offert l'opportunité de bénéficier d'une expérience professionnelle enrichissante au sein d'une équipe dynamique.

Je tiens aussi à présenter mes sincères remerciements à M. Lotfi WAHBI ,chef de projet RD , qui m'a apporté son aide et a fait mon suivi durant toute la période du stage, je le remercie pour tous les conseils et informations qu'il a partagés avec moi, et de sa sympathie sans égale.

Merci également à toute l'équipe de SRH pour avoir fait preuve de disponibilité et d'attention à mon égard tout au long de ma période de stage.

Je tiens aussi à exprimer mes reconnaissances à M. DAHCHOUR Mohamed pour son encadrement de mon projet de fin d'études, pour sa disponibilité et tous ses conseils avisés concernant les missions évoquées dans ce rapport.

Je remercie également les membres du jury qui ont accepté d'évaluer ce travail. Par la même occasion je tiens à exprimer ma gratitude à tout le corps professoral de l'Institut National des postes et télécommunications (INPT) pour l'intérêt qu'ils manifestent dans la formation des futurs ingénieurs. Que tous ceux et celles qui ont contribués de près ou de loin à l'accomplissement de ce travail trouvent l'expression de mes remerciements les plus chaleureux.

Résumé

Afin de réduire le délai de commercialisation et produire des logiciels de qualité, l'entreprise Cegedim a opté pour l'intégration des pratiques DevOps dans les projets de ses clients.

Dans ce cadre vient ce projet pour améliorer et optimiser un pipeline de livraison continue, qui gère les changements depuis le code source jusqu'à la livraison en production.

Après une étude des principales valeurs et pratiques de DevOps notamment la livraison continue et la gestion de configuration, nous avons effectué une étude comparative des outils d'automatisation de ces pratiques. Suite à l'étude comparative, une analyse des besoins et une conception de l'architecture du projet se sont avérées nécessaires, avant la mise en place du pipeline de livraison continue optimisé.

Ce projet a permis de minimiser le temps de déploiement dans les différents environnements et d'avoir une résilience contre les pannes d'infrastructures.

Mots-clés : DevOps, Intégration Continue , Livraison Continue , Logging .

Abstract

In order to reduce time to market and produce quality software, Cegedim opts to integrate DevOps practices into its clients' projects.

In this context, this project aims to improve and optimize a continuous delivery pipeline, which manages changes from source code to delivery of a release in production. Following the comparative study, an analysis of the different needs and a design of the project architecture appeared necessary, before the implementation of the optimized continuous delivery pipeline.

This project allows to minimize the time of deployments in the different environments, and to have resilience against infrastructure failures.

Keywords : DevOps, Continuous Integration , Continuous Delivery, Logging .

ملخص

من أجل تقليل الوقت اللازم للتسويق وإنتاج برامج عالية الجودة ، اختارت Cegedim دمج ممارسات DevOps في مشاريعها.

ضمن هذا الإطار ، يأتي هذا المشروع لتحسين وتحسين خط أنابيب التسليم المستمر ، والذي يدير التغييرات من كود المصدر إلى تسليم الإنتاج.

بعد دراسة القيم والممارسات الرئيسية لـ DevOps بما في ذلك التسليم الإدارة وإدارة التكوين ، أجرينا دراسة مقارنة أدوات التشغيل الآلي لهذه الممارسات. بعد الدراسة المقارنة وتحليل الاحتياجات والتصميم المعماري من المشروع بدا ضروريا ، قبل تنفيذ خط أنابيب التسليم الأمثل المستمر.

ساعد هذا المشروع في تقليل وقت النشر في بيئات مختلفة ولديك مرونة ضد أعطال البنية التحتية. .
الكلمات الرئيسية: DevOps ، التكامل المستمر ، التسليم المستمر ، التسجيل.

1 INTRODUCTION GÉNÉRALE

L'avènement de l'entreprise numérique implique la révision profonde des modes de création d'application. Il n'est plus possible de patienter 6 mois pour les livrables de développement. Avec ces exigences de temps au marché et l'évolution des projets dans des configurations logicielles et d'infrastructure de plus en plus complexes, générant des risques opérationnels et de planification, les besoins d'industrialiser les tests et de fluidifier les déploiements en production se sont progressivement affirmés. En rapprochant les équipes de développement, de test et d'exploitation, le DevOps répond précisément à ce défi du digital. Les entreprises en ont désormais bien conscience.

Parmi les pratiques les plus répandues de DevOps, on trouve la livraison continue et la gestion de configuration. La livraison continue est une stratégie logicielle qui permet aux organisations d'offrir de nouvelles fonctionnalités aux utilisateurs rapidement et efficacement. L'idée de base de livraison continue est de créer un processus reproductible et fiable d'amélioration progressive pour amener le logiciel du concept au client. L'objectif de la livraison continue est de permettre un flux constant de changements vers la production via une ligne de production automatisée de logiciels. Le pipeline de livraison continue est ce qui rend tout cela possible.

C'est dans ce cadre que s'inscrit le présent projet, qui vise à implémenter un pipeline d'intégration et de livraison continue. Ce projet s'est déroulé au sein de l'entreprise Cegedim.

Le présent rapport décrit l'essentiel du travail réalisé lors de ce projet, il est organisé en cinq chapitres :

Dans le premier chapitre, nous présentons l'entreprise d'accueil Cegedim, après nous explorons les problèmes que l'organisation trouve dans l'absence d'une transformation DevOps, ensuite nous expliquons la relation de la CI/CD avec DevOps.

Dans le deuxième chapitre, nous présentons une étude comparative des outils DevOps les plus répandus sur le marché, ainsi que l'outil choisi dans chaque catégorie.

Le troisième chapitre aborde la phase d'analyse et des spécifications des besoins, dans lequel nous allons analyser l'existant et spécifier les besoins, après nous exposons notre solution, finalement nous exposerons les acteurs et leurs cas d'utilisations.

Dans le quatrième chapitre, analyse et conception, nous exposerons le processus du livraison continue, l'architecture physique.

Le cinquième chapitre est consacré à la mise en oeuvre de notre projet .

2 CONTEXTE GÉNÉRAL DU PROJET

Intorudction

afin d'avoir une meilleure compréhension du projet, il est nécessaire d'avoir une idée générale sur l'entreprise Cegedim et ses activités . Nous allons aussi présenter une description générale du projet puis nos objectifs et notre démarche.

2.1 PRÉSENTATION DE CEGEDIM

2.1.1 LE GROUPE CEGEDIM



FIGURE 1 – Répartition de Cegedim

Spécialisée dans la gestion des flux numériques de l'écosystème santé et BtoB, Cegedim intervient également dans la conception de logiciels métiers destinés aux professionnels de santé et de l'assurance. Fondée en 1969 à Paris, l'entreprise compte 4 900 collaborateurs. Elle a réalisé, en 2018, un chiffre d'affaires de 467,7 millions d'euros. Cegedim propose une large gamme de solutions et de services innovants à destination des professionnels de santé, des entreprises de santé (laboratoires pharmaceutiques, compagnies d'assurance) et des entreprises de tous secteurs intéressés par les problématiques d'externalisation, d'hébergement sécurisé et d'échanges dématérialisés. Cegedim est un leader dans chacun de ses secteurs d'activité.(CEGEDIM, 2020).

2.1.2 DOMAINES D'ACTIVITÉS

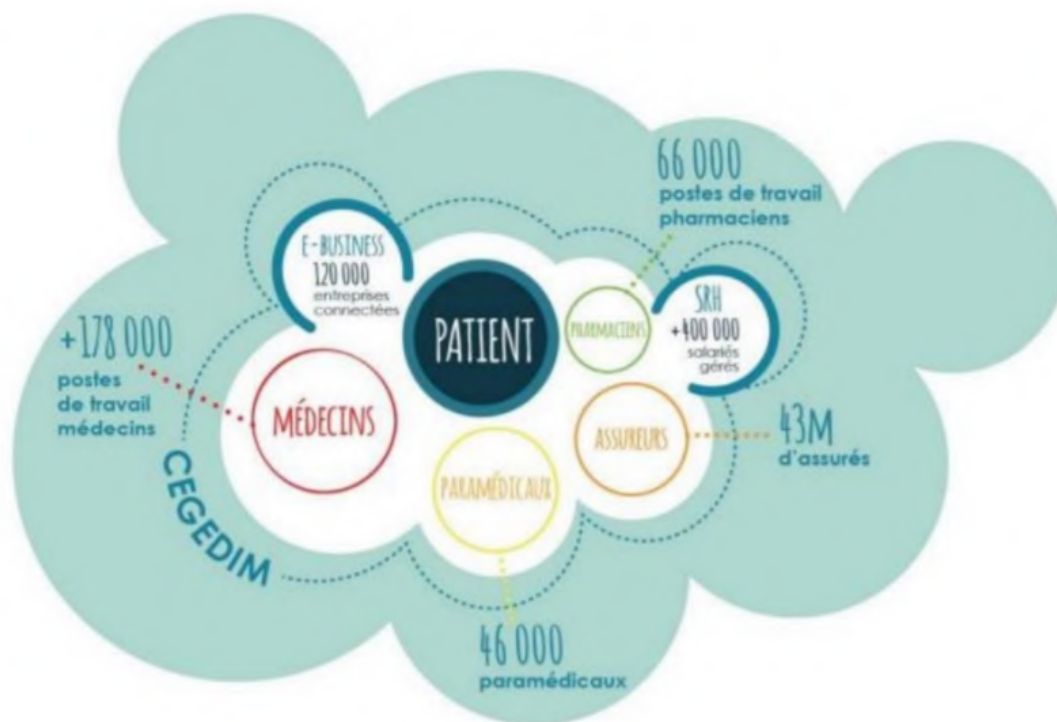


FIGURE 2 – Répartition du domaine d'activité de Cegedim

Comme le montre la figure, CEGEDIM se positionne sur plusieurs activités. Dont :

- Solutions pour les professionnels de santé : Spécifiques à chaque pays et à Chaque spécialité, ces offres ont la particularité d'associer à l'offre technologique des contenus et des bases de données médicales. Elles témoignent de la volonté du Groupe d'être un acteur reconnu dans l'optimisation de la qualité des soins.
- Solutions pour l'industrie pharmaceutique : dans les opérations commerciales et médicales, à travers des solutions innovantes et performantes.
- Solutions pour les mutuelles et assureurs de santé : Dans un contexte concurrentiel réglementaire et économique en pleine évolution, il accompagne assureurs, mutuelles et institutions de prévoyance dans leurs projets technologiques.
- Solutions pour les entreprises de tous secteurs : Cegedim s'est imposé sur les secteurs de l'hébergement des données de santé et sur les problématiques d'externalisation et d'échanges dématérialisés.

2.1.3 LES CLIENTS DU GROUPE CEGEDIM

Les clients du groupe sont généralement les assureurs, les organismes de protection sociale, les caisses de prévoyances et les caisses de sécurité. Ces dernières ont, en général, plusieurs systèmes d'information pour les différents métiers de l'assurance personne comme l'allocation famille, vieillesse, retraite, AMO, AMC.



FIGURE 3 – Clients du groupe Cegedim

2.1.4 CEGEDIM EN CHIFFRES

Si on compare le chiffre d'affaires durant le premier trimestre (du 1er janvier au 31 mars) de l'année 2020 avec le chiffre d'affaires du premier trimestre de l'année 2021, on constate une augmentation de 1 de chiffre d'affaires de Cegedim. Les figures suivantes illustrent les données financières de Cegedim en million d'euros :

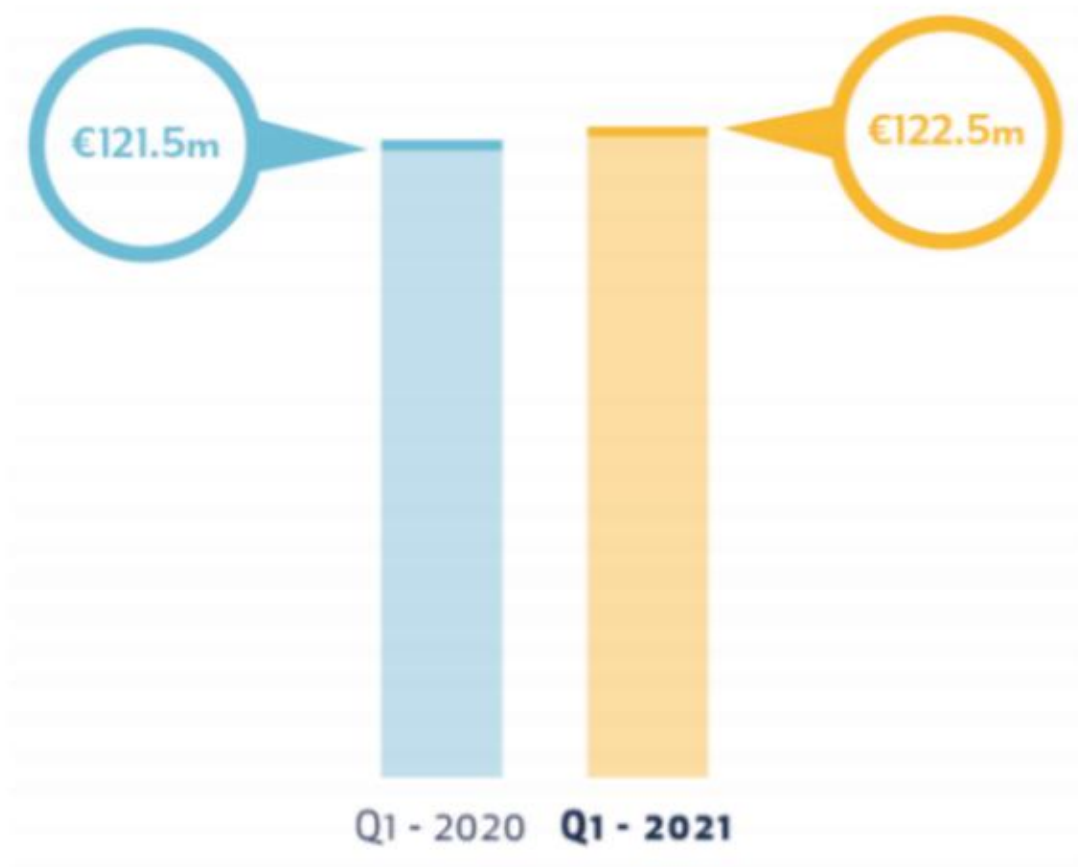


FIGURE 4 – Evolution de chiffre d'affaires de Cegedim



FIGURE 5 – Détails sur le chiffre d'affaires

2.2 CEGEDIM MAROC

CEGEDIM Maroc est une filiale du groupe CEGEDIM Insurance Solutions, située sur Arribat center de Rabat, est un centre de Recherche et Développement en édition de progiciels. Le centre travaille en coordination avec les autres centres de RD de CEGEDIM en France pour maintenir et faire évoluer le parc de progiciels.

2.2.1 PRÉSENTATION DE CEGEDIM SRH



FIGURE 6 – Logo CEGEDIM SRH

Cegedim SRH est une filiale du groupe Cegedim spécialisée dans les solutions et services pour la gestion de la paie et des Ressources Humaines. Acteur innovant du cloud RH et des services externalisés, Cegedim SRH dispose d'une expertise depuis plus de 25 ans dans le domaine.

Cegedim SRH se positionne comme le n°2 du marché de l'externalisation de la paie et des Ressources Humaines en France. Dans ce cadre, elle s'appuie sur la plateforme SIRH nommée TEAMSIRH pour proposer des solutions RH adaptées aux besoins, au contexte et à la taille de ses clients (de 200 à > 25.000 salariés).

Cette offre va de la mise à disposition des outils en mode SaaS jusqu'à la prise en charge de processus RH en mode Business Process Outsourcing (BPO). Cegedim SRH connaît également une forte croissance internationale. Ainsi, elle est installée au Maroc, en Suisse et en Angleterre.

2.2.2 MISSIONS CEGEDIM SRH

Cegedim SRH opère auprès des plus grands acteurs du marché public et privé, et accompagne ses clients par des prestations de conseil à forte valeur ajoutée, matérialisées à travers une offre complète et globale de prestations articulées autour des axes suivants :

- SaaS : abonnement aux services hébergés de TEAMS RH incluant la maintenance corrective et les mises à jour légales et conventionnelles de l'application.
- Processing : externalisation partielle avec pilotage de la relation client, le suivi du Traitement de la paie, des opérations d'exploitation, de production et d'édition.
- BPO (Business Process Outsourcing) : externalisation complète avec prise en charge de l'ensemble des opérations de traitement de la paie (accréditation ISAE 3402).
- BPO on demand : choix dans un catalogue de services des processus RH à externaliser (soldes de tout compte, déclaratifs sociaux, gestion des arrêts maladie, suivi des visites médicales...).

La mission principale du groupe est la conduite des transformations RH créatrices de valeur pour les clients et faire du système d'information RH un véritable levier de performance au service de leur stratégie.

2.2.3 ORGANISATION DU GROUPE

Cegedim SRH est organisé suivant l'organigramme suivant :



FIGURE 7 – Organigramme du Groupe Cegedim SRH

L'équipe que j'ai intégré est répartie entre l'agence 7 « Rabat » et l'agence 3 « Boulogne » sous contrôle de la direction des opérations.

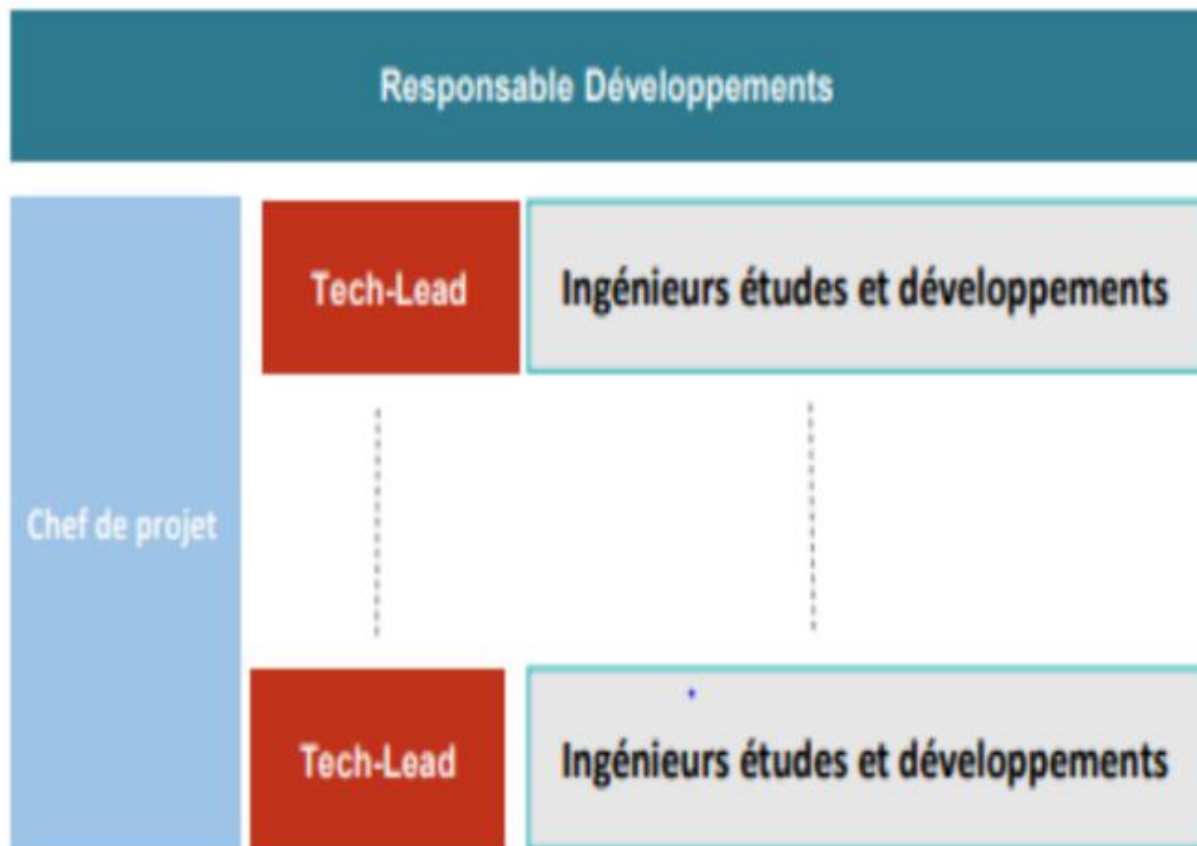


FIGURE 8 – Département RD - Cegedim SRH

Responsable Développement :

- Accompagner le chef de projet dans la gestion des projets.
- Produire les indicateurs sur l'activité de développement.
- Suivi des risques et gestion des alertes.
- Travailler en collaboration avec les autres responsables de pôles.

Chef de projet :

- Valider les spécifications fonctionnelles.
- Garant du suivi du processus de delivery et des indicateurs clés.
- Elaboration du plan de charge et la planification des équipes.
- Garant des engagements (qualité, délai, coût).

Tech-Lead :

- Responsable de la qualité technique (audit, conception, ...).
- Responsable des bonnes pratiques de développement.
- Encadrement et assistance technique.

- Participer aux développements.
- Assister le chef de projet pour les estimations de charge

Ingénieur étude et développement :

- Corriger les anomalies et développer de nouveaux Modules.
- Respecter les bonnes pratiques de développement.
- Participer à la définition de la couverture des tests techniques.
- Rédiger les documents techniques.

Organisation QA Cible

Responsable QA :

- Etablir et piloter une stratégie de test.
- Accompagner le Team Lead dans la gestion des projets.
- Contribuer à améliorer les processus de test.
- Produire les indicateurs sur l'activité de testing.
- Suivi des risques et gestion des alertes.
- Travailler en collaboration avec les autres responsables de pôles.

TL fonctionnel/technique :

- Garant du suivi du processus de Delivery et des indicateurs clés.
- Elaboration du plan de charge et la planification des équipes.
- Accompagner et suivre les testeurs dans la mise en place des bonnes pratiques et des outils.
- Accompagner les ingénieurs QA sur l'élaboration des plans de test.
- Réaliser le reporting de ces activités.
- Accompagner l'intégration des nouveaux arrivants et veiller à la montée en compétence.

Ingénieur QA :

- formaliser les scénarios de test fonctionnels et automatisés.
- Valider et vérifier le développement de l'application.
- Respecter les bonnes pratiques de testing.
- Rédiger les documents fonctionnels.

3 CADRE DU PROJET

3.1 ETUDE DE L'EXISTANT

ARKEVIA ou Arkevia Refonte est un coffre-fort électronique vous permettant de recevoir et de conserver les bulletins de paie déposés par votre employeur dans un espace sécurisé.

- Un accès exclusif, confidentiel et hautement sécurisé à vos documents importants
- Un site accessible 7j/24h depuis n'importe quel endroit via une connexion Internet
- Un espace de stockage gratuit pour vos documents personnels
- La conservation de vos documents durant 50 ans quelle que soit votre situation professionnelle

Services offerts

Bénéfices pour le salarié :

- Accès illimité (24h/7j) depuis n'importe quel endroit grâce à une connexion Internet
- Espace de stockage dédié pour archiver les bulletins de paie et documents RH en version électronique
- Archivage à valeur probante : tous les bulletins de salaire déposés dans le coffre-fort ont la même valeur juridique que leur équivalent papier, et ceci grâce à la signature numérique qui y est apposée
- Durée de conservation de 50 ans des bulletins de salaire
- Espace personnel sécurisé de 1 Gb
- Protection des documents importants contre le vol et la perte
- Accès au coffre-fort sécurisé et garanti, même en cas de départ de l'entreprise
- Données stockées en France

Bénéfices pour l'entreprise :

- Optimisation des processus RH d'édition et de distribution documentaire
- Réduction des coûts de distribution
- Image moderne et innovante de la DRH
- Enrichissement de l'expérience du salarié

Vue Globale

Au global, on a une répartition des tâches sous la forme schématique suivante :

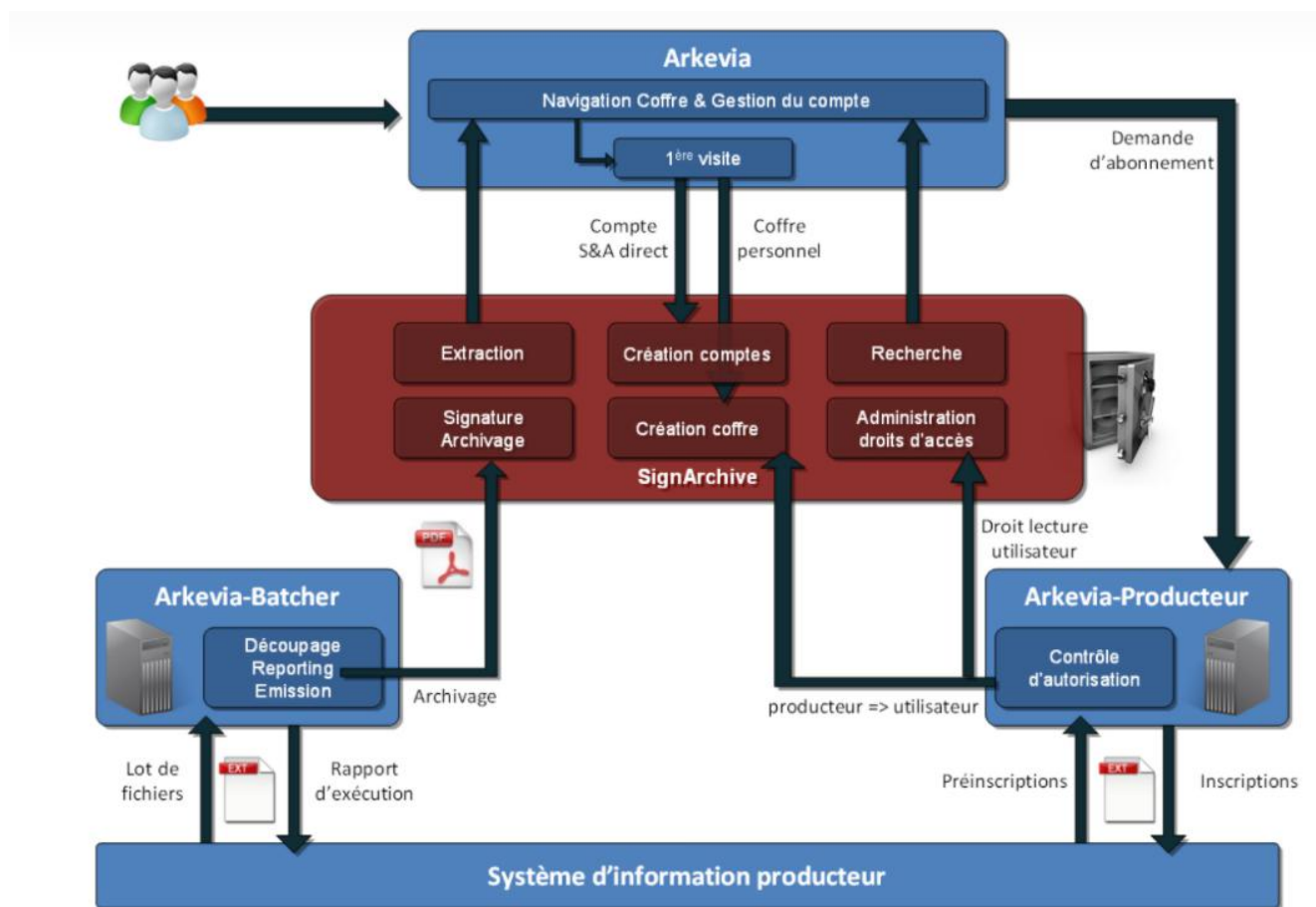


FIGURE 9 – Vue globale du système

- **Arkevia :**
Arkevia se connecte à SignArchive pour en afficher le contenu, indépendamment du fait que SRH publie ou non des documents pour cet utilisateur. Cependant, si l'utilisateur souhaite s'abonner au service de production SRH, alors, Arkevia pousse la demande via un web service au module approprié, Arkevia-SRH.
- **Arkevia-Producteur :**
Arkevia-Producteur (Arkevia-SRH) peut prendre en compte ou non la demande, selon les autorisations qui lui ont été publiées par SRH directement, via le fichier des pré-inscriptions. Chaque inscription ou résiliation est remontée à SRH via le même canal que le fichier des préinscriptions.
- **Archive-Batcher :**
lors de la fabrication des bulletins de paie, le module de gestion interne à SRH peut constituer une archive des bulletins à archiver conforme aux utilisateurs Arkevia inscrits. Ce fichier est poussé au module SignArchive-Batcher qui pousse les fichiers dans SignArchive.
- **Arkevia Legacy :**
C'est l'ancienne version du portail Arkevia, les clients utilisaient cette version sont migrés au fur et à mesure vers la refonte

Module applicatifs

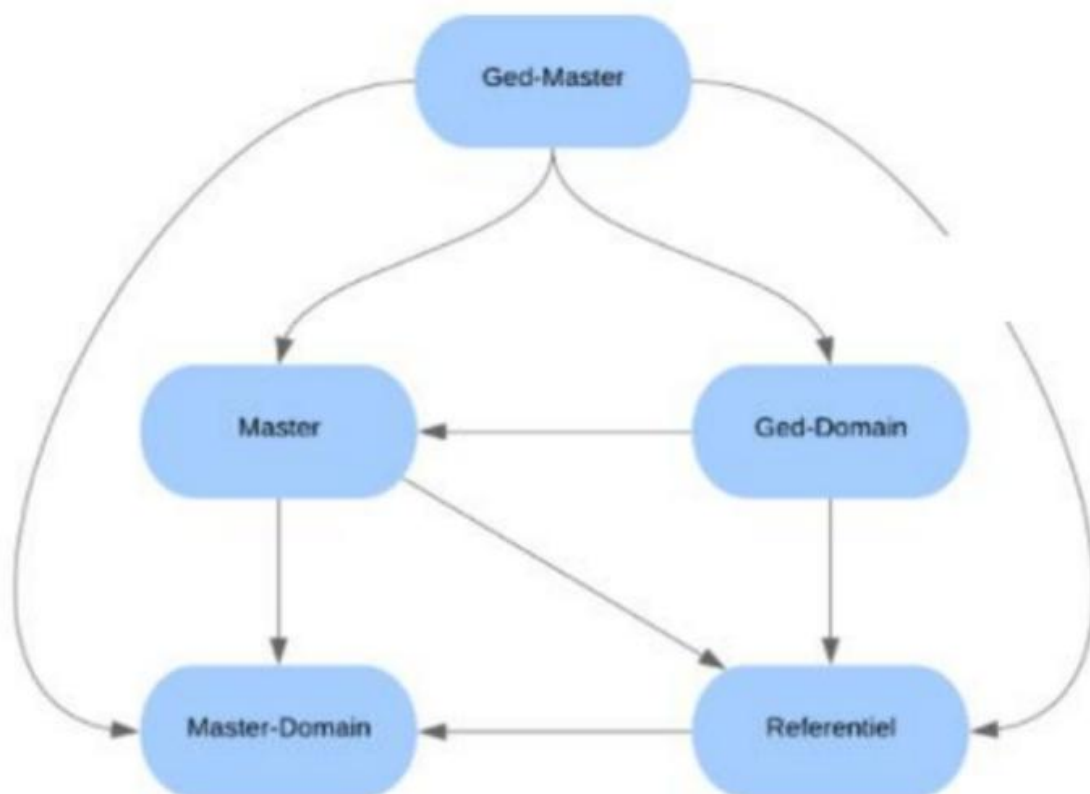


FIGURE 10 – Les dépendances entre les modules

- **Master :**
Ce module regroupe les services de configuration, d'authentification, de gestion de compte.
- **Master-Domain :**
Ce module gère les interactions avec la base de données, il regroupe tous les entités qui servent à faire le mapping avec les tables. On trouve aussi l'ensemble des objets d'accès aux données.
- **Ged-Domain :** Ce module regroupe tous les entités spécifiques pour la gestion des documents, qui servent à faire le mapping avec les tables.
- **Ged-Master :**
Regroupe les services de la gestion des répertoires et documents, responsable de la consommation de l'API SignArchive.
- **HS-SY-Client-Arkevia :**
Dans ce module on a regroupé tous les services spécifiques au client Arkevia, l'inscription, désactivation de compte, exportation des documents...
- **HS-SY-Client-Standard :**
Dans ce module on trouve tous les services standards et les classes des écrans utilisés dans l'application.
- **HS-SY-Portal;**
Contient les composants, les scripts, images et feuille de style utilisé par Arkevia refonte.

3.2 PROBLÉMATIQUE

CI - CD

Sur la version actuelle d'Arkevia, la procédure de livraison se fait d'une façon manuelle :

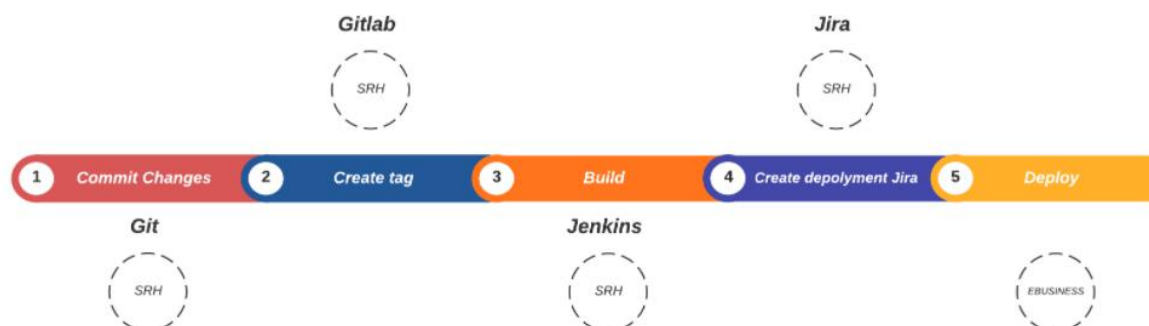


FIGURE 11 – Existant

Logging

Dans toutes les applications informatiques, le risque de bugs et d'anomalies persistent toujours. Les fichiers de logs ont été créés afin de tracer l'exécution du code et de fournir les détails d'événements. Cela permet au développeur de détecter l'origine des anomalies, et ainsi, pouvoir les corriger.

Et c'est aussi le cas pour Arkevia, les informations d'exécution sont stockées dans des fichiers de logs (des fichiers texte), qui sont consultés par la suite, dans le cas d'une anomalie, par exemple. Mais, bien que ces fichiers de logs contiennent des détails d'exécution (Type, Date, heure, ...), il reste difficile d'identifier la cause du bug et le temps nécessaire pour ce fait n'est pas optimale.

3.3 OBJECTIF DU PROJET

Le but de ce projet est de pouvoir choisir, puis implémenter, une architecture CI/CD permettant de réduire au maximum les possibilités de régression et le temps de livraison (Time to Market), et ainsi assurer une transformation réussie vers une structure Devops, tout en respectant les bonnes pratiques associées. Aussi pouvoir exploiter au maximum ces données de logs d'une manière optimale et conviviale. Et en parallèle, implémenter un système de monitoring, permettant de suivre l'impact des exécutions sur la performance et les ressources.

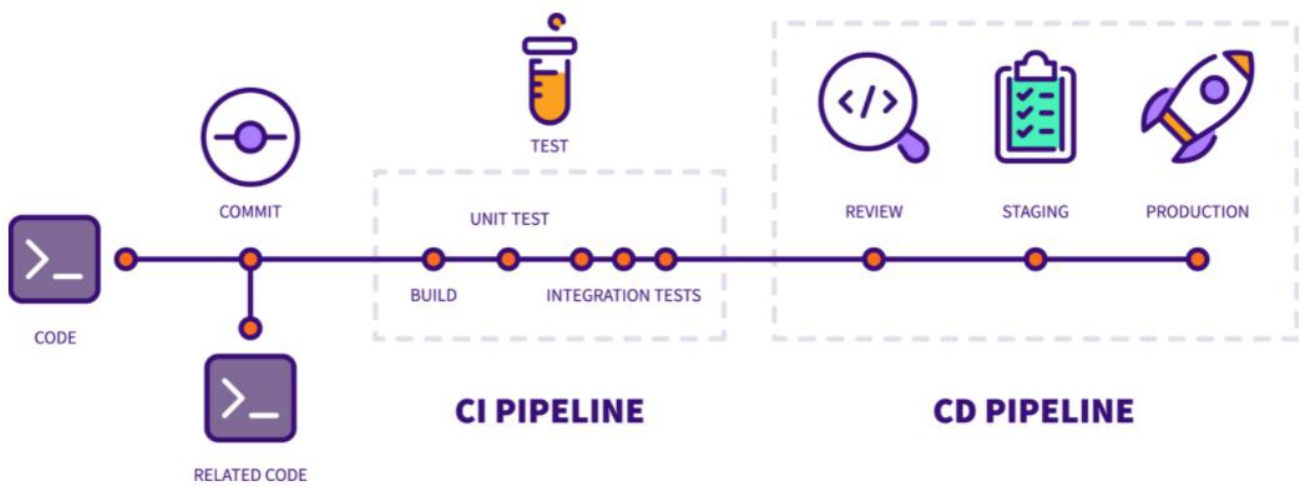


FIGURE 12 – Pipeline Intégration / Déploiement Continu

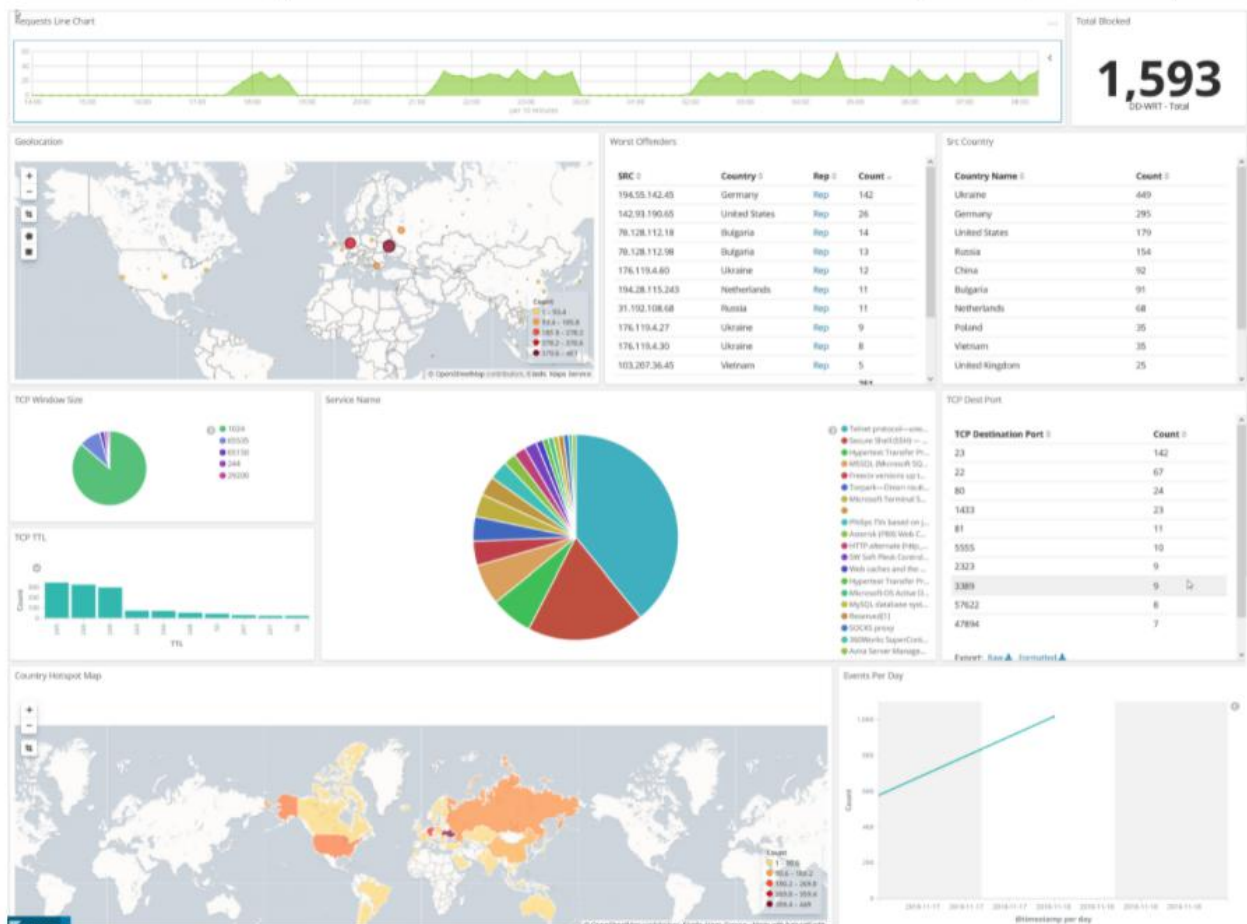


FIGURE 13 – Dashboard des Logs

3.4 COMMENT DÉMARRER UNE TRANSFORMATION DEVOPS?

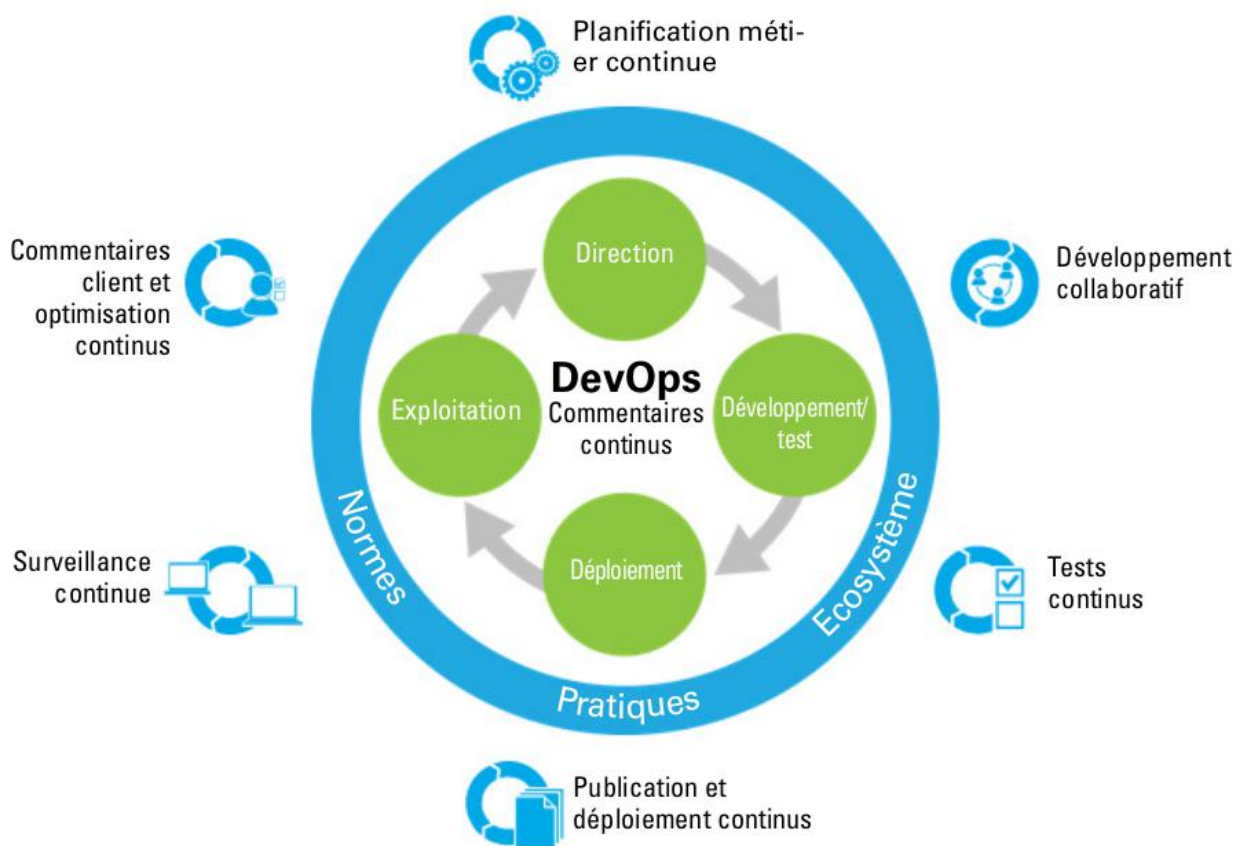


FIGURE 14 – Collaboration Dev et Ops

DevOps est un ensemble de bonnes pratiques pour l'industrialisation du système d'information, plus une stratégie pour réduire le Time To Market. Les méthodes agiles sont des formes efficaces de développement et gestion de projet qui offrent plusieurs avantages :

- Une haute qualité du produit ou service.
- La prise en compte des besoins Métier, utilisateurs de manière continue.
- Des livraisons régulières des nouvelles fonctionnalités (développements réguliers).
- Les mises en production (donc mises à disposition du produit aux utilisateurs) dépendent de plans de déploiement, de releases planifiées tous les mois, 3 mois, 6 mois, etc.
- Une fois en production, le produit n'est que peu étudié afin de s'assurer qu'il correspond réellement aux attentes des utilisateurs finaux. On note une faible réactivité face aux retours.

Pour ces raisons et d'autres, nous sommes confronté à un besoin d'étendre les principes agiles sur toute la chaîne de création d'un produit ou service informatique. Ce qui permettra de :

- Réduire le délai de mise en production, de mise en ligne, de mise sur le marché du produit.
- Pouvoir livrer et démontrer à tout moment (livraison continue) l'état des produits.
- Rapprocher les métiers du développement et des opérations (exploitation) pour une meilleure performance.
- Être plus efficient tout au long du cycle de vie du produit.

- Gérer efficacement et de façon plus simple les environnements via l'automatisation.

En vertu de ce qui précède, une comparaison des méthodes agile (sans DevOps) et méthode agile avec DevOps se résume dans la possibilité de livrer à n'importe quel moment tant que les tests sont valides, au lieu de la fin de chaque sprint .

3.4.1 COMMUNICATION, PROCESSUS ET MÉTHODES AGILES :



FIGURE 15 – Communication agile

Les équipes de développement ont besoin de communiquer avec les équipes opérationnelles. Elles doivent pouvoir adapter rapidement leur environnement de développement et les changements d'architecture sur leur code. De la même manière, lorsqu'une nouvelle fonctionnalité demande un changement d'architecture ou l'installation d'un nouvel environnement et/ou serveur, les équipes opérationnelles se doivent d'être de bon conseil sur la configuration et l'installation. Elles auront souvent une vision plus juste que les développeurs.

Les équipes opérationnelles ont besoin de communiquer avec les équipes de développement qui ont des connaissances sur des outils, des processus qui peuvent aider à rendre les environnements plus facile à gérer, plus efficace et plus propre. Elles ont également besoin de comprendre le besoin applicatif et fonctionnel des développeurs pour permettre une meilleure optimisation des ressources, un meilleur monitoring et une meilleure remontée d'erreur.

Les méthodes agiles ne sont pas réservées uniquement aux équipes de développement mais peuvent aussi être appliquées sur toute la chaîne IT. Les méthodologies de développement Agile comme Kanban et Scrum peuvent révolutionner les pratiques opérationnelles de la même ma-

nière qu'elles ont changé la façon dont les développements s'opèrent. Prenons l'exemple de l'analyse des problèmes en production qui nécessite une coopération entre les équipes d'exploitation et celles de développement comme la gestion des logs et des exceptions, des outils de concentration et d'analyse de logs. DevOps permet avec les méthodes agiles de gérer sous forme de points les problèmes rencontrés et le partage des informations, des solutions facilitent la résolution des dits problèmes.

L'organisation, bien qu'étant le point le plus complexe à aborder pour l'adoption d'une culture DevOps, n'est pas impossible à mettre en place, même pour de grandes entreprises : Amazon a adopté ce concept, et une devise en est alors ressortie : «You build it, you run it». Le point essentiel pour la communication est de s'assurer que les objectifs soient communs aux différentes parties et se dirigent vers une amélioration de la disponibilité, de la performance et de l'évolutivité des applications.

3.4.2 INDUSTRIALISATION DES PROCESSUS DE LIVRAISON

Beaucoup trop d'organisations restent encore sur le cycle :

Développement long → Recette longue → Livraison douloureuse → Bugs → Recettes correctives → Redéploiement de patch .

Soit par méconnaissance, soit par peur du changement ou par le coût de celui-ci. Ce type de projet avec un planning de livraison à plusieurs mois regroupe souvent

un lot de fonctionnalités pour éviter une multitude de déploiement (souvent due à la frilosité de la mise en production). On se retrouve alors avec des allers retours dans tous les sens et des attentes dans chaque équipe, ce qui peut créer des groupes (eux / nous) et certaines frustrations (exemple : ça fait deux mois qu'on a fini de développer cette fonctionnalité mais elle n'est toujours pas testée ni livrée en production).

Prenons le postulat de base : « une portion de code qui n'est pas livrée n'apporte aucune valeur ajoutée à l'entreprise ou à l'application ». Plus les déploiements sont fréquents, plus le processus est maîtrisé. Il y a moins de code à livrer, donc moins de risque d'erreur, moins d'impact lors d'un roll back (retour-arrière). De plus, une application avec laquelle on peut rapidement livrer des nouvelles fonctionnalités ou des correctifs de bugs aura un meilleur retour des utilisateurs (plus de réactivité) et donc une meilleure image. C'est là qu'intervient la méthodologie du déploiement continu permettant le déploiement du code de façon rapide et efficace. Pour arriver à une automatisation complète ou quasi-complète de la chaîne de production, plusieurs étapes sont nécessaires :

«Continuous Integration» est le processus d'intégration (génération de code, compilation, exécution des tests unitaires, fonctionnels, ..) à chaque changement d'environnement (source code, os...).

C'est une très bonne pratique de développement qui consiste à utiliser le «développement piloté par les tests (Test driven Development, TDD) » ou à aller encore plus loin en utilisant le

»développement piloté par les comportements » (Behavior Driven Development, BDD) pour permettre une couverture améliorée du code et du fonctionnement de l'application. Ces tests sont ensuite exécutés par le serveur d'intégration à chaque évolution de code ou changement de périmètre fonctionnel. Le contrôle et la gestion des résultats (tests, fiabilité, analyse du code) sont à la charge du serveur d'intégration.

«Continuous Delivery» est l'étape suivante et caractérise la possibilité de déployer ses applications à tout moment.

Cette étape est très importante et est dépendante des résultats de l'intégration, si toutes les contraintes sont au vert, le déploiement peut se faire de manière automatisée sur l'environnement cible (recette, pré-production, production).

«Continuous Deployment» est l'ensemble de ces processus et méthodologies permettant le déploiement en continu en production.

Malheureusement, la démarche de déploiement continu est loin d'être simple à mettre en œuvre, car dépendante de tous les acteurs du système d'information de l'entreprise.

4 ANALYSE ET SPÉCIFICATIONS DES BESOINS

Introduction

Pour mettre en place un pipeline de livraison continue, nous avons besoin d'un ensemble d'outils. Choisir la combinaison la plus convenable reste une tâche laborieuse. A cette fin, nous avons fait une étude comparative afin de choisir les meilleurs outils répondant aux besoins techniques de notre projet. L'outil approprié doit avoir des fonctionnalités adéquates d'une part, d'autre part il doit respecter les critères étudiés. La démarche suivie ainsi que les détails l'étude effectuée sont présentés dans la suite .

Etude comparative des outils DevOps

4.1 DÉMARCHE DE L'ÉTUDE COMPARATIVE

Savoir choisir une combinaison des outils DevOps homogènes et ayant un interfaçage de communication automatique entre eux est un facteur majeur du succès de la pipeline. Plusieurs outils DevOps existent dans le marché, et d'autres apparaissent chaque jour. Le choix, dépend ainsi des critères à prendre en considération. Pour bien organiser cette étude, qui constitue une phase préalable à la phase de mise en œuvre du pipeline, nous suivons les étapes suivantes :

- Lister les outils les plus connus aux marchés et sélectionner les plus intéressants pour les analyser et les étudier,
- Élaborer des critères de comparaison pertinents et rigoureux. De plus, nous ne tiendrons pas compte des critères non différenciés malgré leur importance,
- Comparer les outils selon les critères établis pour sortir avec des tableaux comparatifs, avant d'en faire la synthèse et identifier l'outil approprié.

Dans ce qui suit, nous présenterons l'étude comparative complète pour chaque catégorie d'outils qui reflète l'approche que nous venons de décrire.

4.2 OUTILS DE GESTION DE CODE SOURCE

La gestion du code source (SCM) est la manière dont les modifications de logiciels sont effectuées. Le SCM a un certain nombre d'objectifs qui visent essentiellement à faire en sorte que les équipes de développement puissent livrer plus rapidement des modifications de code de qualité[3]. En améliorant le suivi, la visibilité, la collaboration et le contrôle tout au long du cycle de livraison, les outils de SCM permettent aux développeurs travaillant sur des projets complexes de bénéficier de plus de créativité, de liberté et d'options. En outre, SCM peut protéger les fichiers sources de tout type d'anomalie, et permet à toutes les équipes de savoir qui a fait quelle modification et à quel stade. Cette étude sur le gestionnaire de code source a pour but de consolider le choix de l'outil GitLab, déjà utilisé en interne, face à ses concurrents BitBucket et GitHub.

4.2.1 CHOIX DES CRITÈRES

Les critères qui guideront notre choix d'un outil de gestion du code source sont :

- Open Source.
- Gratuit ou payant.
- Les gestionnaires de version supportés.
- Suivi des bugs.
- Gestion d'équipe.
- Accessibilité.

4.2.2 ANALYSE

GitLab



FIGURE 16 – Logo GitLab

GitLab est un gestionnaire de dépôt git basé sur le Web, comprenant un wiki et des fonctions de suivi des problèmes. fonctions de suivi des problèmes. GitLab fournit une gestion centralisée des dépôts Git, permettant aux utilisateurs d'avoir un contrôle total sur leurs dépôts ou projets.

Écrit en Ruby (avec des modules complémentaires en Go), le gestionnaire comprend des contrôles d'accès granulaires d'accès granulaires, des revues de code, le suivi des problèmes, des flux d'activité, des wikis et l'intégration continue. intégration. En décembre 2016, il compte 1400 contributeurs open source et est utilisé par des des entreprises comme Sony, IBM, le CERN, la NASA, etc.

BitBucket



FIGURE 17 – Logo BitBucket

BitBucket est un service web hébergé pour les projets logiciels utilisant le système de contrôle de version Mercurial ou git. Mercurial ou git, et propose des plans commerciaux ou gratuits. Ces derniers incluent un nombre illimité de répertoires privés et jusqu'à 5 utilisateurs. Son élasticité simplifie la collaboration en équipe. L'extraction de requêtes, les autorisations de branches et les discussions en ligne et les discussions en ligne sont des fonctionnalités clés. Écrit en Python, utilisant le cadre web Django, BitBucket permet aux équipes de fournir et de partager un code de meilleure qualité et de partager du code plus rapidement. Utilisé pour son extrême élasticité, notamment dans un environnement commercial, il garantit aux développeurs vitesse et fiabilité quel que soit le lieu du projet où que se situe GitHubt

GitHub



FIGURE 18 – Logo Github

GitHub est un service de dépôt web hébergé offrant toutes les fonctionnalités de gestion de code source fonctionnalités de gestion, tout en garantissant aux développeurs un espace pour stocker leurs projets et et construire des logiciels en parallèle. GitHub fournit la collaboration, le contrôle d'accès contrôle d'accès, des wikis et des outils simples de gestion des tâches pour les projets Conçu par des développeurs pour des développeurs, GitHub offre une interface graphique et une interface web et une interface web ainsi qu'une intégration

mobile. Il ne se limite pas développement de logiciels : son aspect ouvert et "réseau social" est fondamental. Il vous permet de de faire une copie du projet public d'une autre personne et d'en modifier les caractéristiques tout en visualisant l'œuvre tout en visualisant le travail et les profils de chaque personne.

4.2.3 COMPARAISON ET SYNTHÈSE

Le tableau résume la comparaison des gestionnaires de code source présentés précédemment, selon les critères établis et expliqués auparavant.

| <div> <div>Outil</div> <div>Critère</div> </div> |  <div>GitLab</div> |  <div>GitHub</div> |  <div>Bitbucket</div> |
|--|---|---|--|
| Open Source | Oui | Non | Non |
| Licence | Gratuit pour la version communauté | Payant | Payant |
| Gestionnaire de version | Git | Git | Git, Mercurial |
| Fonctionnalité intégration continue | Intégré ou à travers d'autres outils | À travers d'autres outils seulement | À travers d'autres outils seulement |
| Suivi de problèmes | Oui | Oui | À travers un autre outil Jira |

FIGURE 19 – Récapitulatif de comparaison des outils de gestion de code source

L'outil GitLab est open source, de plus gratuit pour la version communauté. Cette raison nous a poussées de le choisir sans hésitation. D'une autre, son intégration de fonctionnalité d'intégration continue et suivi de problèmes permettra dans le futur d'abandonner l'utilisation des outils à part.

4.3 OUTILS D'INTÉGRATION CONTINUE

Les serveurs d'intégration continue (CI) sont le point d'entrée de nombreuses nouvelles organisations en DevOps. En tant qu'extension des méthodes agile, les serveurs CI permettent la construction et le test automatisés, ainsi que divers niveaux de notifications fondamentaux pour maintenir les efforts agiles sur la bonne voie.

Enfin, le CI facilite le processus de livraison des logiciels, en raccourcissant les cycles de livraison et en donnant aux développeurs plus de liberté pour se concentrer sur l'innovation. Il permet à différents développeurs ou équipes de travailler en parallèle sur différents aspects d'un même projet.

Parmi les logiciels d'intégration continue existants sur le marché, nous avons choisi de comparer Jenkins le plus connu, Travis CI et CircleCI.

4.3.1 CHOIX DES CRITÈRES

Les critères qui vont orienter notre choix d'outil d'intégration continue sont :

- Open Source : le code source du logiciel est accessible.
- Licence : gratuit ou payant.
- Les gestionnaires de version supportés.
- Installation : Complexité de mettre en place l'outil pour un environnement de production.
- Gestionnaire de code source : supporté par défaut ou avec des plug-ins.
- Système d'exploitation : supporté pour lancer des compilations et des tests de code.

4.3.2 ANALYSE

Jenkins

Jenkins est un outil d'intégration continue Open Source écrit en Java. Jenkins est un successeur de Hudson. Il supporte les outils SCM tels que Subversion, git, etc. Jenkins peut également exécuter des scripts Shell et des projets Ant ou Maven [6].

Jenkins dispose en outre de nombreux plug-ins qui le rendent compatible avec tous les langages de programmation et une grande majorité de systèmes de contrôle de version et de référentiels. Jenkins permet aux utilisateurs de concevoir et livrer des applications à grande échelle rapidement et supporte conception, déploiement et automatisation dans la plupart des projets.

TravisCI

TravisCI est un service hébergé d'intégration continue (IC) en Open Source pour concevoir et tester des projets hébergés sur GitHub. Les exécutions de builds et de tests sont déclenchées automatiquement toutes les fois qu'un commit est réalisé et poussées vers un référentiel GitHub [6].

TravisCI se configure en plaçant un fichier `travis.yml` dans le répertoire racine de votre référentiel. TravisCI a été conçu pour exécuter tests et déploiements en laissant les développeurs se concentrer sur le code. Cette automatisation facilite le déploiement simple, rapide et agile pour les équipes logicielles.

TravisCI est gratuit pour les projets Open Source, payant pour les projets commerciaux ou privés.

CircleCI

CircleCI est une plateforme d'intégration et de déploiement continu qui automatise les processus de build, de test et de déploiement. Il permet aux équipes de développement de déployer des projets logiciels rapidement, tout en facilitant l'élasticité. CircleCI est un serveur Cloud hébergé qui réduit considérablement l'effort de test. Il supporte un grand nombre de technologies, comme Ruby on Rails, Sinatra, Node, Python, PHP, Java et Clojure[6].

Il dispose de quatre fonctionnalités : une configuration rapide, une intégration avec un grand nombre d'outils (conférant de la souplesse dans l'environnement de travail utilisateur), il supporte tous les tests et permet de configurer facilement un enchaînement. L'association de ces fonctionnalités permet à l'utilisateur de livrer des projets mieux testés plus rapidement.

4.3.3 COMPARAISON ET SYNTHÈSE

Le tableau synthétise la comparaison des serveurs d'intégration continue présentés préalablement, selon les critères établis et exposés auparavant




| Outil Critère |  Jenkins |  Travis CI |  circleci |
|-----------------------------|--|--|---|
| Open Source | Oui | Oui | Non |
| Licence | Gratuit | Gratuit, il existe une version payante | Payant |
| Installation | Facile | Difficile | Aucune installation, SaaS |
| Gestionnaire de code source | GitLab, GitHub, Bitbucket, TFS, CVS, Preforce | GitHub, Bitbucket | GitHub, Bitbucket |
| Système d'exploitation | Linux, MacOS, Windows, Unix | Linux, MacOS | Linux, MacOS |

FIGURE 20 – Récapitulatif de comparaison des outils d'intégration continue

D'après le tableau de cette étude et les différentes informations collectées, nous sommes arrivés à la conclusion suivante : Jenkins est l'outil le plus adapté pour notre cas, il est populaire et bénéficie d'une vaste communauté qui délivrent des plugins ouvrant un large

champ de possibilité, contrairement à TravisCI, sans oublier son aspect open source et licence gratuit qui lui donne un pas d'avance par rapport à CircleCI.

4.4 OUTILS DE DÉPLOIEMENT CONTINU

Afin d'avoir un déploiement simple et rapide , Docker reste le maitre en matière , nous avons opté pour deployer Arkevia dans des conteuneurs Docker , puis Kubernetes se chargera de l'orchestration de ces derniers .



FIGURE 21 – Logo Kubernetes et Docker

4.5 OUTILS D'ANALYSE DES LOGS

- EFK stack la stack EFK est choisie comme la solution open source de référence pour la gestion des logs avec l'orchestrateur Kubernetes. La collecte des logs des différents conteneurs se fait par la couche Fluentd, puis acheminés vers Elasticsearch pour le stockage. La lecture et l'analyse s'effectuent grâce à la couche de représentation Kibana. C'est ainsi que la tentation d'exploiter le cluster Elasticsearch sous Kubernetes devient évidente.

4.5.1 ELASTICSEARCH



FIGURE 22 – Logo Elasticsearch

Elasticsearch est au cœur de la Suite Elastic. Il stocke toutes vos données et fournit des capacités de recherche et d'analyse de manière évolutive. Elasticsearch peut être utilisé sans utiliser d'autres composants pour alimenter votre application en termes de recherche et d'analyse.

4.5.2 FLUENT BIT



FIGURE 23 – Logo Fluent Bit

Fluent Bit permet aux clients d'acheminer leurs journaux de conteneurs vers différentes solutions de surveillance notamment Elasticsearch

4.5.3 KIBANA



FIGURE 24 – Logo Kibana

Kibana est l'outil de visualisation pour la Suite Elastic et peut vous aider à obtenir des informations puissantes sur vos données dans Elasticsearch. On l'appelle souvent une fenêtre sur la Suite Elastic. Il offre de nombreuses visualisations, notamment des histogrammes, des cartes, des graphiques linéaires, des séries chronologiques, etc. Vous pouvez créer des visualisations en quelques clics et explorer les données de manière interactive. Il vous permet de créer de superbes tableaux de bord en combinant différentes visualisations, en partageant avec d'autres et en exportant des rapports de haute qualité.

Kibana dispose également d'outils de gestion et de développement. Vous pouvez gérer les paramètres et configurer les fonctionnalités de sécurité X-Pack pour Elastic Stack. Kibana dispose également d'outils de développement qui permettent aux développeurs de créer et de tester des requêtes d'API REST.

4.6 AUTRES OUTILS

- Rancher



FIGURE 25 – Logo Rancher

Rancher, projet open-source créé par la société Rancher Labs est un outil gratuit d'orchestration de conteneurs Docker. Il permet de facilement déployer des conteneurs Docker sur des machines possédant Docker. Grâce à une configuration simple et complète, il permet de lier ses conteneurs afin de composer des architectures de services

aisément. Il peut déployer des conteneurs sur des services cloud comme AWS, Azure, DigitalOcean mais aussi sur des machines personnalisées possédant Docker tout en s'appuyant sur docker-machine.

- Helm



FIGURE 26 – Logo Helm

Dans Kubernetes la configuration de nos services / applications se fait généralement via des fichiers yaml. Quand on a une seule application en ligne, cela reste assez simple mais dès qu'on a plusieurs environnements, applications et services, on se retrouve très vite submergé de fichiers plus ou moins semblables.

C'est là que Helm intervient !

Helm est le package manager soutenu et recommandé par Kubernetes, il est aussi un des seuls sur le marché, son unique concurrent, KPM de CoreOS, n'est plus maintenu depuis juillet 2017.

Helm permet donc de déployer des applications / stacks complètes en utilisant un système de templating et de dépendances afin d'éviter la duplication et avoir ainsi une arborescence cohérente pour nos fichiers de configurations.

Mais Helm ce n'est pas que ça, il propose également la possibilité de gérer vos Charts avec la possibilité de les compresser et de les mettre dans un répertoire distant (Cdn, Git, disque local ou partagé...). Il intègre aussi un système facilitant les Updates et Roll-backs de vos applications.

5 RÉALISATION

- Introduction Dans ce chapitre, nous mettons en place le pipeline d'intégration et déploiement continu , ainsi que le système de gestion des logs après avoir déployé Arkevia .

5.1 ENVIRONNEMENT DU TRAVAIL

5.1.1 IDE : INTELLIJ IDEA



FIGURE 27 – Logo IntelliJ IDEA

IntelliJ IDEA est un environnement de développement intégré (IDE) écrit en Java pour le développement de logiciels informatiques. Il est développé par JetBrains (anciennement IntelliJ), et est disponible en tant qu'édition communautaire sous licence Apache 2 et dans une édition commerciale propriétaire. Les deux peuvent être utilisés pour le développement commercial.

5.1.2 WILDFLY



FIGURE 28 – Logo WildFly

WildFly, anciennement JBoss Application Server ou JBoss, est un serveur d'applications Java EE Libre écrit en Java, publié sous licence GNU LGPL. Étant écrit en Java, WildFly peut être utilisé sur tout système d'exploitation fournissant une machine virtuelle Java (JVM). Le nom JBoss est aujourd'hui utilisé pour JBoss EAP, produit dérivé WildFly et faisant l'objet d'un support commercial [9].

5.1.3 GIT



FIGURE 29 – Logo Git

Git est un logiciel de gestion de versions décentralisé. C'est un logiciel libre créé par Linus Torvalds, auteur du noyau Linux, et distribué selon les termes de la licence publique générale GNU version 2. Le principal contributeur actuel de git et depuis plus de 16 ans est Junio Hamano. En 2016, il s'agit du logiciel de gestion de versions le plus populaire qui est utilisé par plus de douze millions de personnes.

5.1.4 SONARQUBE



FIGURE 30 – Logo SonarQube

Développé par SonarSource, SonarQube est un logiciel open source de gestion de la qualité du code. Il est utilisé pour inspecter le code source des logiciels et applications en développement et détecter des bugs, vulnérabilités de sécurités, instances de code dupliqué et autres anomalies pouvant nuire à la qualité du code source, et ainsi au fonctionnement de l'application qui en résulte. L'objectif est d'aider les développeurs à créer un code d'une meilleure qualité, tout en simplifiant le processus de développement.

5.2 IMPLÉMENTATION DE LA CHAÎNE CI/CD

L'intégration, la livraison et le déploiement continus sont des pratiques conçues pour aider à augmenter la vitesse de développement et la sortie de produits bien testés. L'intégration continue encourage les développeurs à intégrer fréquemment leur code à une base de code partagée de manière précoce et chaque intégration est vérifiée par un build pour minimiser les erreurs d'intégration. La livraison continue supprime les obstacles sur le chemin du déploiement ou de la publication. Le déploiement continu va encore plus loin en déployant chaque build réussi qui réussit automatiquement la suite des tests.

La livraison continue est un processus plutôt que des outils et nécessite un état d'esprit et une culture qui doivent percoler du haut vers le bas au sein d'une organisation. Une fois que l'organisation a adopté la philosophie, la partie suivante et la plus difficile consiste à cartographier le flux des logiciels au fur et à mesure qu'ils passent du développement à la production.

Dans notre cas, nous avons réalisé une étude comparative des outils DevOps pour choisir les meilleurs et les plus adaptés aux besoins de Cegedim.

5.2.1 CONFIGURATION DE JENKINS

Plugins nécessaires

pipeline – git – gitlab – gitlab api – gitlab hook – credentials – email extension – Maven integration – Sonarqube scanner for jenkins – ssh credentials plugin – config file provider.

Integration GitLab / Jenkins

On crée un token d'accès personnel pour autoriser l'accès de Jenkins à GitLab :

- Dans le coin supérieur droit, sélectionnez votre avatar.
- Edit profile
- Dans la barre latérale gauche, sélectionnez Access tokens.
- Créez un personal access token avec la case API cochée.

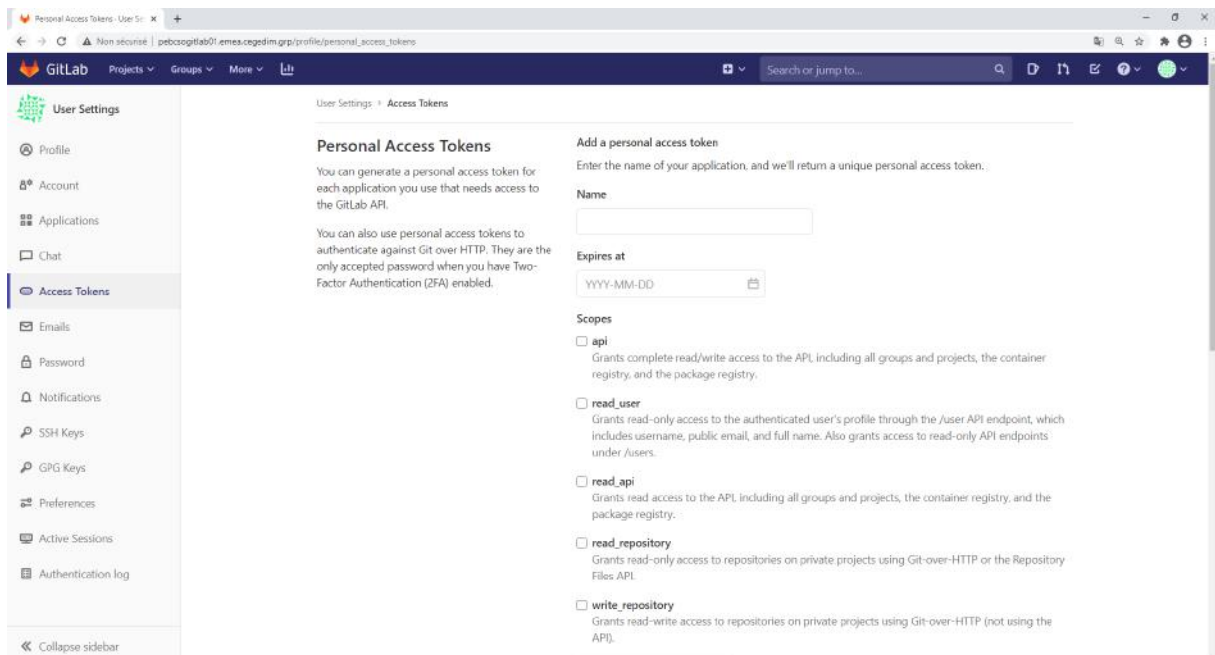


FIGURE 31 – Génération du Token sur GitLab

Configuration de Jenkins -Accédez à Manage Jenkins > Configure

- Dans la section GitLab, cochez la case : Enable authentication for '/project' end-point checkbox.

- Cliquez sur add puis choisissez Jenkins Credential Provider

- Choisissez GitLab API token comme type

- Saisir la valeur du GitLab API token et puis cliquer sur add

- Saisir GitLab host URL :

[http ://pebcogitlab01.emea.cegedim.grp/arkevia-legacy/sy-portal.git](http://pebcogitlab01.emea.cegedim.grp/arkevia-legacy/sy-portal.git) - Cliquez sur test connection pour vous assurer que la connexion est réussie avant de continuer.

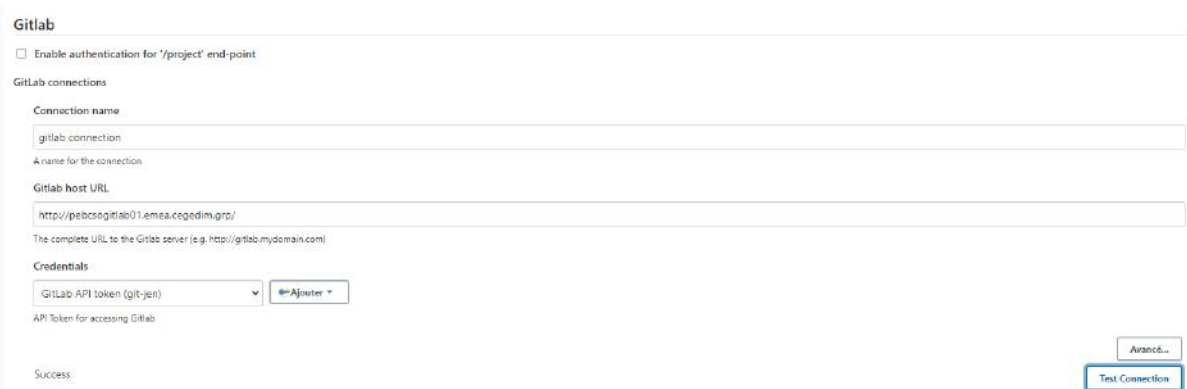


FIGURE 32 – Configuration de GitLab sur Jenkins

Configuration du Projet Arkevia sur Jenkins - Sur votre instance Jenkins, accédez à new item

- Saisir le nom du projet (Arkevia) et choisir Pipeline comme type de projet

- Dans la barre latérale gauche, sélectionnez configure

- Dans la section Build triggers cocher : Build when a change is pushed to GitLab. Push

Events

- Puis cliquer sur avancé et generate pour avoir un token qu'on aura besoin après pour la configuration du webhook .
- Dans la section pipeline choisir : Pipeline script from scm comme definition
- Choisir git comme scm
- Saisir le repository url : `http://pebcsoGitlab01.emea.cegedim.grp/arkevia-legacy/sy-portal.git`
- Ajouter votre connexion gitlab
- Spécifiez la branche sur laquelle vous travaillez
- Choisir Jenkinsfile comme script path

The screenshot shows the Jenkins configuration interface for a new project. The 'Repository URL' field is populated with 'http://pebcsoGitlab01.emea.cegedim.grp/arkevia-legacy/sy-portal.git'. Under 'Credentials', a dropdown menu shows 'gitlabconnection/***** (gitlabconnection)'. The 'Branches to build' section includes a 'Branch Specifier (blank for \'any\')' field with the value '*/branch/arkevia2_cicd_master'. The 'Script Path' field is set to 'Jenkinsfile'. At the bottom, the 'Lightweight checkout' checkbox is checked. Buttons for 'Add Repository', 'Add Branch', and 'Ajouter' are visible on the right side of the form.

FIGURE 33 – Configuration du Projet sur Jenkins

5.2.2 JENKINSFILE

Un pipeline Jenkins se compose de plusieurs états ou étapes, et ils sont exécutés dans une séquence l'un après l'autre. JenkinsFile est un simple fichier texte utilisé pour créer un pipeline sous forme de code dans Jenkins. Il contient du code en langage spécifique au domaine Groovy (DSL), qui est simple à écrire et lisible par l'homme.

Intégration Continue

L'intégration Continue sur Arkevia contient les étapes suivantes :

- Build automatisé (Maven) : génération du WAR .

```
[INFO] Installing /var/lib/jenkins/workspace/ortal_brancharkevia2_cicd_master@2/HS-SY-Portal/target/HS-SY-Portal-trunk-SNAPSHOT.war to /home/jenkins/.m2/repository/com/cedim/workflow/HS-SY-Portal/trunk-SNAPSHOT/HS-SY-Portal-trunk-SNAPSHOT.war
[INFO] Installing /var/lib/jenkins/workspace/ortal_brancharkevia2_cicd_master@2/HS-SY-Portal/.flattened-pom.xml to /home/jenkins/.m2/repository/com/cedim/workflow/HS-SY-Portal/trunk-SNAPSHOT/HS-SY-Portal-trunk-SNAPSHOT.pom
[INFO] -----
[INFO] Reactor Summary for HS-SY-build trunk-SNAPSHOT:
[INFO]
[INFO] HS-SY-build ..... SUCCESS [ 1.979 s]
[INFO] HS-SY-Master-Domain ..... SUCCESS [ 4.749 s]
[INFO] HS-SY-Module-Referentiel ..... SUCCESS [ 0.898 s]
[INFO] HS-SY-Master ..... SUCCESS [ 4.183 s]
[INFO] HS-SY-Module-Ged-Domain ..... SUCCESS [ 1.055 s]
[INFO] HS-SY-Module-Ged-Master ..... SUCCESS [ 0.698 s]
[INFO] HS-SY-Client-Standard ..... SUCCESS [ 1.825 s]
[INFO] HS-SY-Client-Arkevia ..... SUCCESS [ 2.324 s]
[INFO] HS-SY-Portal ..... SUCCESS [01:46 min]
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 02:04 min
[INFO] Finished at: 2021-08-16T16:06:32+02:00
[INFO] -----
```

Activate Windows

FIGURE 34 – Génération du WAR au niveau du Jenkins Agent

- Analyse du code source (SonarQube)

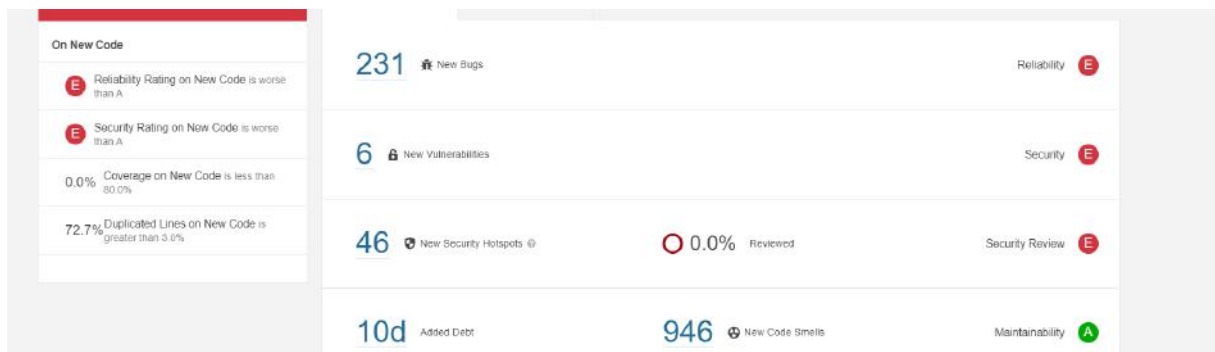


FIGURE 35 – Résultats de l'analyse du Code avec SonarQube

SonarQube permet l'inspection de code en continu, vérification de la qualité depuis la dernière version de la dernière analyse, notifications par email, ce qui permet de détecter les problèmes dès leur introduction dans le code avant que le coût de remédiation soit élevé.

- Tests unitaires et d'intégration :

Les tests Unitaires , exécutés par JUnit , donnent le premier niveau de retour aux développeurs si leur code a cassé l'un des cas de test unitaire.

Les tests d'intégration automatisés ,exécutés par Mockito , qui examinent



FIGURE 36 – Logo JUnit

J Unit est un framework de test unitaire pour le langage de programmation Java.

J Unit définit deux types de fichiers de test. Les TestCase (cas de test) sont des classes contenant un certain nombre de méthodes des tests. Un TestCase sert généralement à tester le bon fonctionnement d'une classe. Une TestSuite permet d'exécuter un certain nombre de TestCase déjà définis.

les tests d'intégrations basées sur l'API entre les modules du système et/ou l'intégration basée sur l'API du système avec les applications environnantes. Ces tests d'intégration basés sur l'API peuvent utiliser des souches de test pour les applications environnantes. Cela donne le deuxième niveau de retour aux développeurs si leur code a cassé l'un des tests d'intégration.



FIGURE 37 – Logo Mockito

L'écriture d'objets de type mock peut s'avérer longue et fastidieuse, les objets ainsi codés peuvent contenir des bugs comme n'importe quelle portion du code. Des frameworks ont donc été conçus pour rendre la création de ces objets fiable et rapide.

Mockito C'est l'un framework Java très connu permettant de générer automatiquement des objets 'mockés'. Couplé avec JUnit, il permet de tester le comportement des objets réels associés à un ou des objets 'mockés' facilitant ainsi l'écriture des tests unitaires.

- Dockérisation (Docker)
- Récupération des Artifacts (Maven / docker / Jfrog Artifactory)

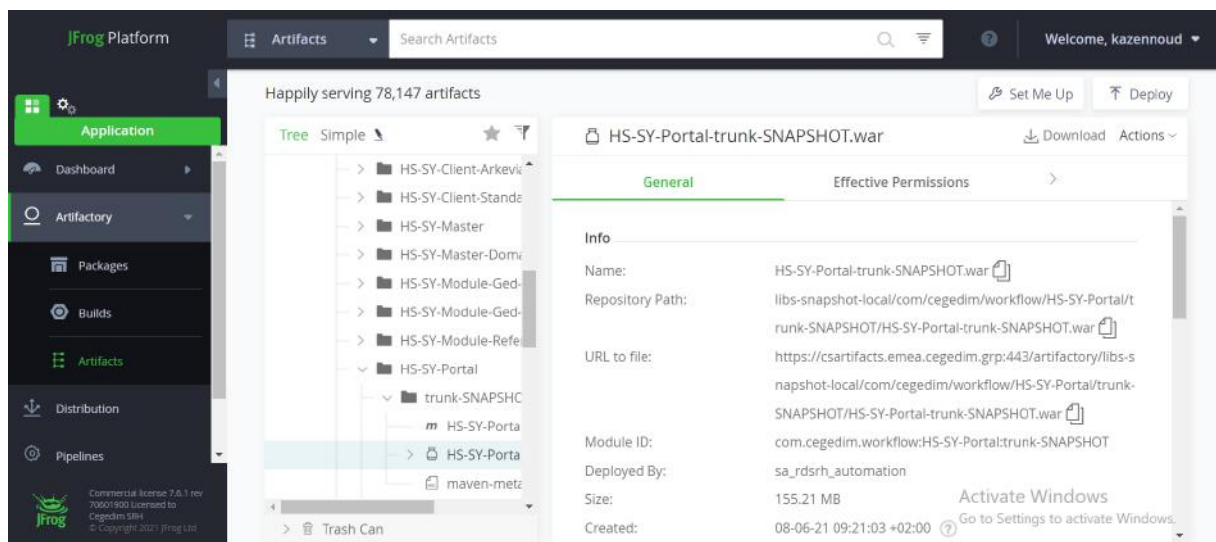


FIGURE 38 – Depot du WAR sur Jfrog Artifactory

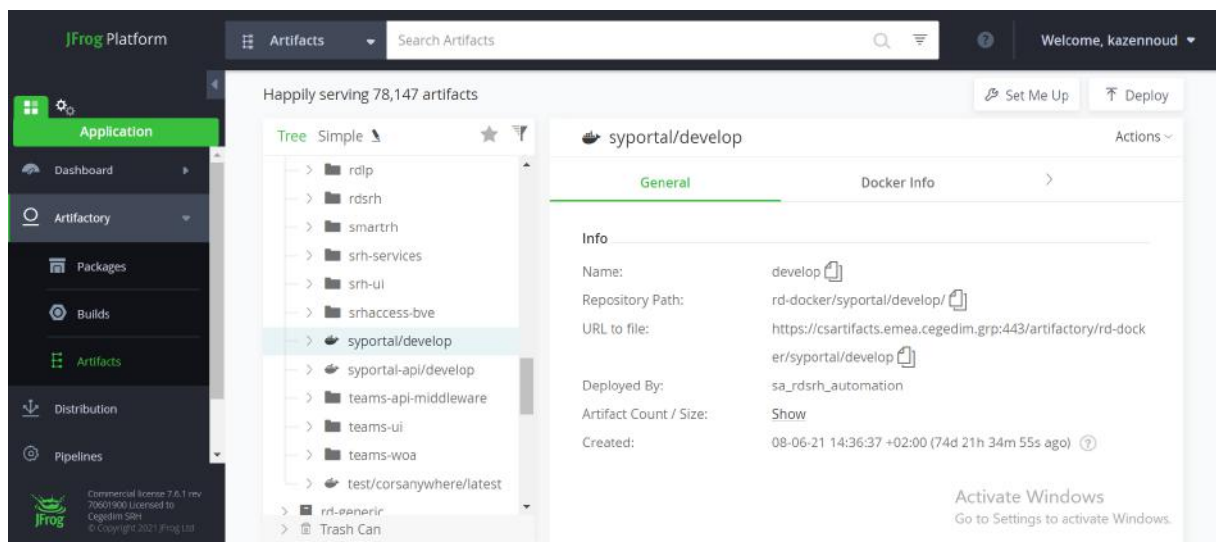


FIGURE 39 – Depot de l'image Docker sur Jfrog Artifactory

- Notifications par mail (Plugin Jenkins / SMTP)

Répondre Répondre à tous Transférer MI



mar. 03/08/2021 16:27

Jenkins-srh <noreply@cegedim-srh.com>

[SUCCESS] - [sy-portal - brancharkevia2_master] [12]

À ■ AZENNOUD Khalil

SRH R&D
Yes !! The job is done

Jenkins
sy-portal/brancharkevia2_master
successful

Build duration: 9 min 12 sec and counting

WEBAPP STATUS
Ready to access & play with

FIGURE 40 – Notification par mail au git committer pour le succès du job Jenkins

Répondre Répondre à tous Transférer MI



lun. 16/08/2021 11:31

Jenkins-srh <noreply@cegedim-srh.com>

[FAILURE] - [sy-portal - brancharkevia2_cicd_master] [53]

À ■ AZENNOUD Khalil

Message build.log (113 Ko)

Action Items

SRH R&D
Damn !! The job couldn't be done

Jenkins
sy-portal/brancharkevia2_cicd_master
failed

Build duration: 6 min 30 sec and counting

FIGURE 41 – Notification par mail au git committer pour le fail du job Jenkins

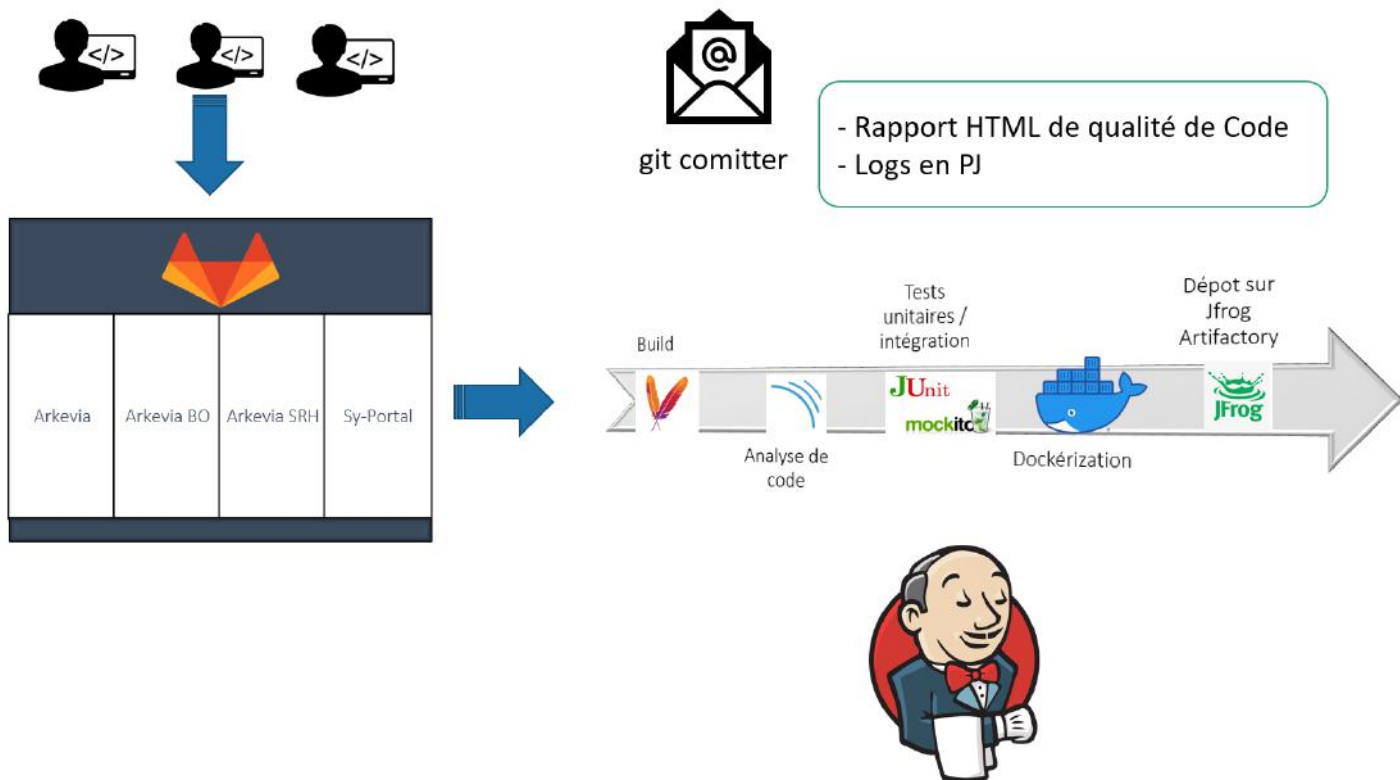


FIGURE 42 – Chaîne Intégration Continue

Déploiement Continu

Kubernetes utilise les pods comme unité d'exécution de base d'une application k8s. Un pod représente une unité de déploiement : une seule instance de l'application Arkevia dans Kubernetes, qui peut être constituée d'un seul conteneur ou d'un petit nombre de conteneurs étroitement couplés et partageant des ressources.

Un objet de déploiement (ressource) dans Kubernetes fournit une mise à jour déclarative des pods, décrite à l'aide d'un fichier manifeste écrit en yaml ou en exécutant une commande kubectl. Vous décrivez un état souhaité dans un déploiement et le contrôleur de déploiement modifie l'état réel en l'état souhaité à une vitesse contrôlée.

Pour déployer Arkevia , nous allons utilisé les Helm Charts . Les Helms Charts sont des séries de fichiers pour décrire des ressources Kubernetes liées .

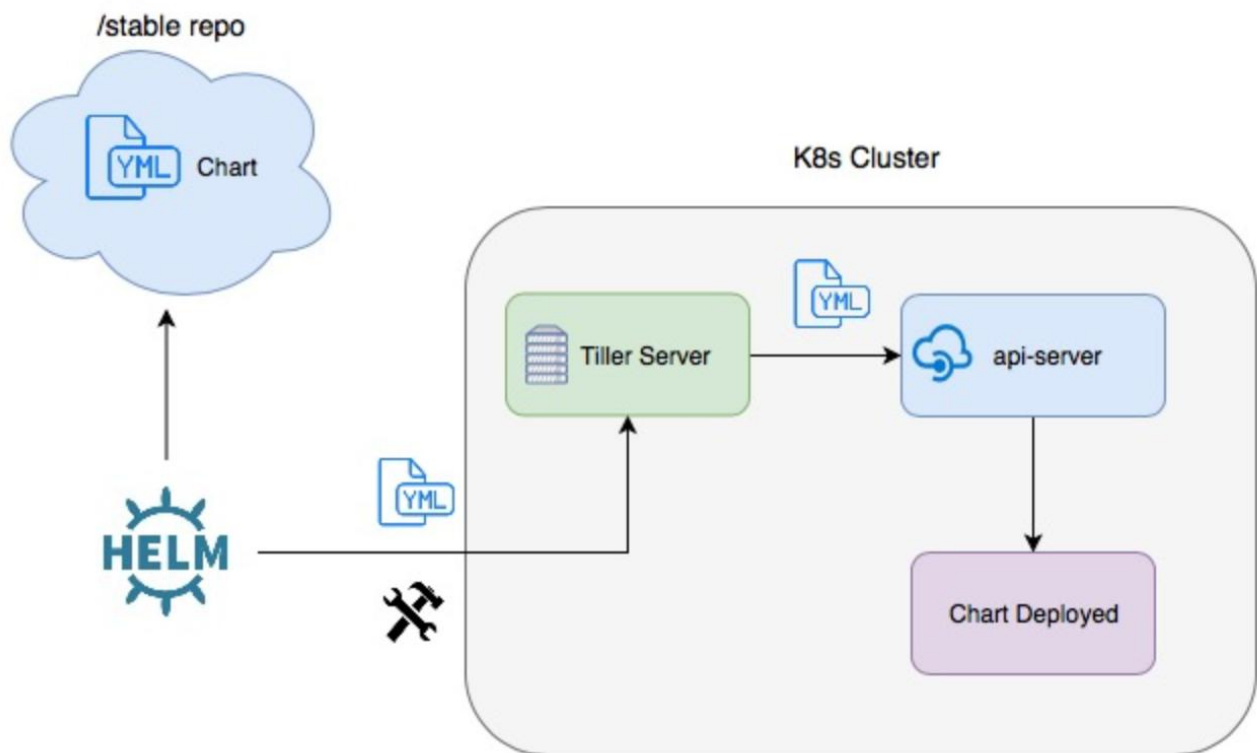


FIGURE 43 – Déploiement Continu (Helm/Kubernetes)

5.3 IMPLÉMENTATION DU SYSTÈME DE GESTION DES LOGS

Les processus de pod s'exécutant sous Kubernetes produisent fréquemment des journaux. Pour gérer efficacement ces données de journal et s'assurer qu'aucune perte de données de journal ne se produit lorsqu'un pod se termine, un outil d'agrégation de journaux doit être déployé sur le cluster Kubernetes.

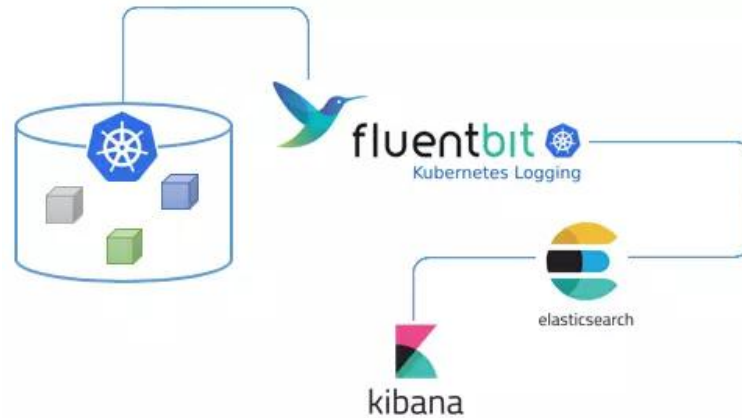


FIGURE 44 – Logging (EFK stack)

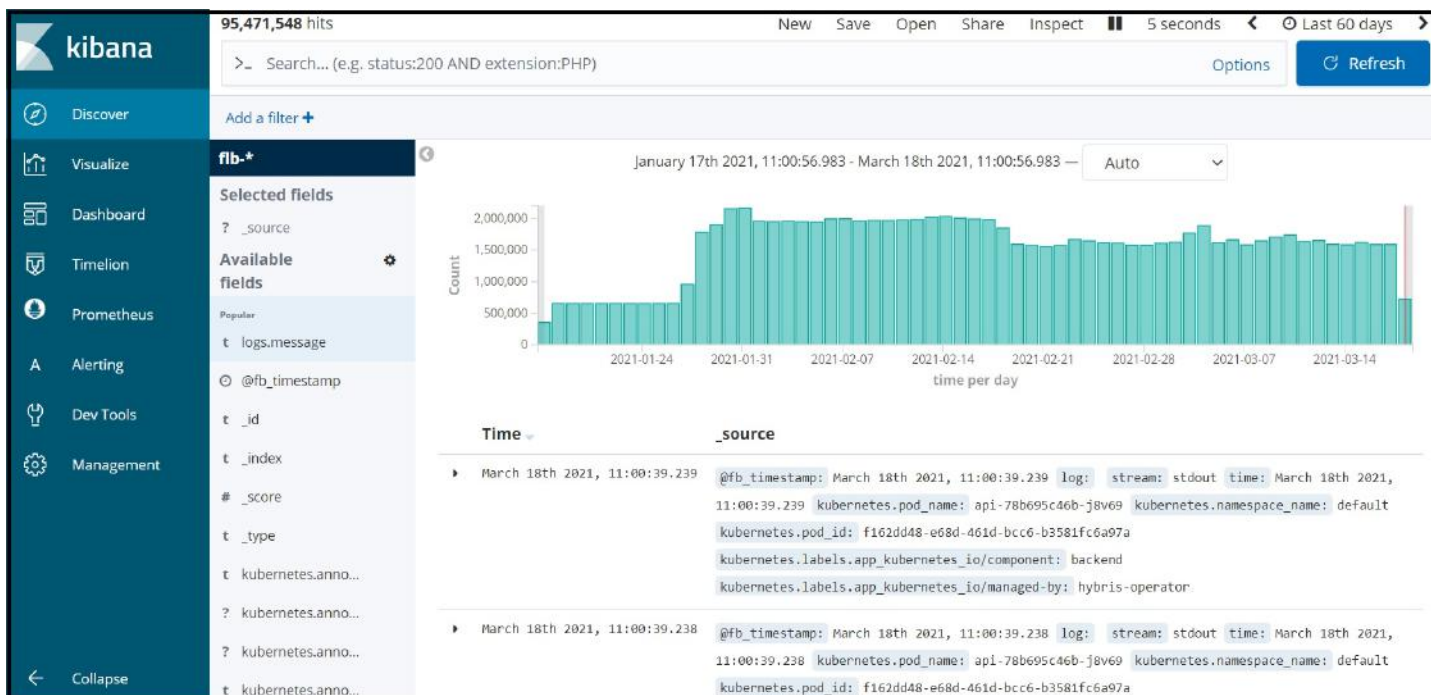


FIGURE 45 – Dashboard des logs capturés par Fluent-bit

5.4 CONCLUSION

Pour la mise en oeuvre de ce projet , nous avons eu recours à plusieurs technologies synthétisées dans la figure 46 ci dessous :

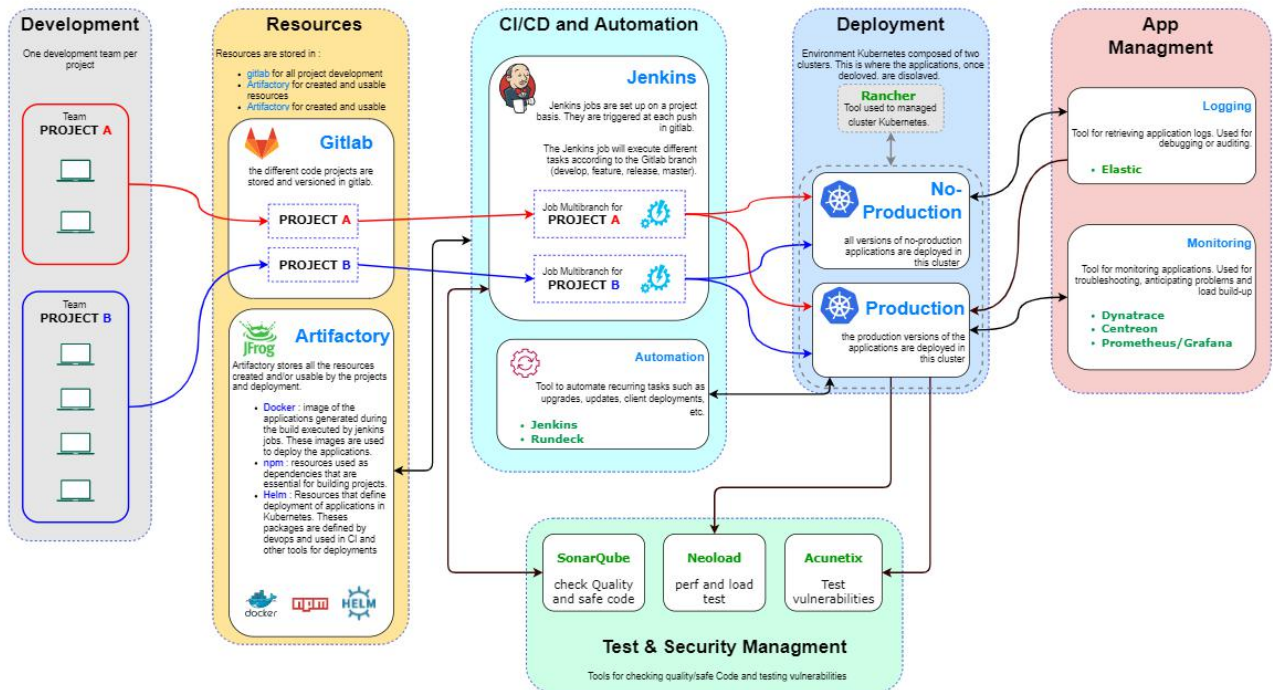


FIGURE 46 – Architecture Finale

Après une série de tests, nous avons opté pour les technologies et les programmes qui nous semblaient adéquats, car même si nous avons au final choisi Jenkins par exemple pour l'implémentation du pipeline , nous n'avons pas négligé ses concurrents, ils ont eu aussi droit à leur part de test mais ont été recalés pour une raison ou une autre.

6 CONCLUSION GÉNÉRALE

J'ai effectué le stage de fin d'études au sein de l'entreprise CEGEDIM. Le projet de fin d'études qui s'est étalé sur six mois consiste en l'intégration de la culture DevOps dans un Cloud (CEGEDIM CLOUD). L'implémentation de la chaîne Intégration/Livraison Continues et du système de gestion des logs vont permettre à CEGEDIM SRH de gagner un temps considérable en ce qui concerne la gestion de ses déploiements internes et de son travail.

Malgré la pandémie, le projet s'est bien déroulé, je suis resté en contact avec mon encadrant et j'ai pu communiquer régulièrement avec lui. SRH et son personnel étaient aussi très coopératif et toujours prêts à répondre à mes questions.

C'était une expérience très fructueuse qui m'a permis d'être dans la peau d'un administrateur système et d'un ingénieur DevOps en même temps, je remercie énormément et encore une fois toutes les personnes qui ont contribué à l'aboutissement de ce projet .

7 ANNEXES

7.1 IMPLÉMENTATION DU JENKINSFILE

Le jenkinsfile est le fichier présent dans le SCM du v projet Arkevia qui permet de décrire la pipeline CI-CD attendu . Écrits en Groovy,on décrit l'enchaînement et le parallélisme des instructions. Toutes les étapes CI/CD seront présentes sur le Jenkinsfile .

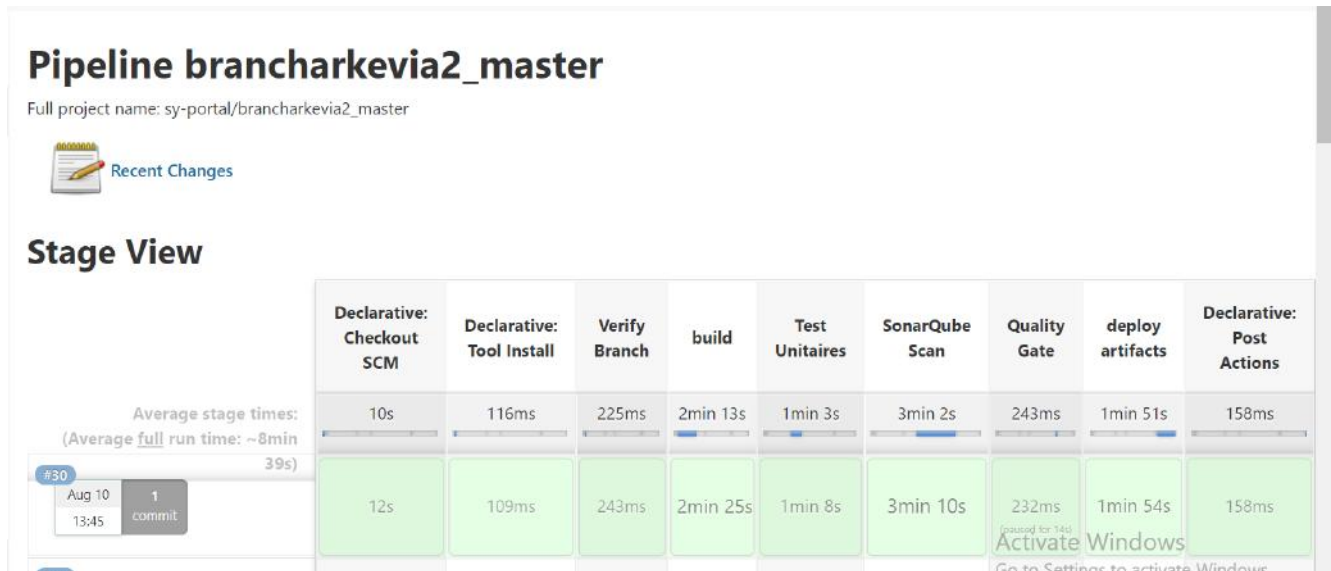


FIGURE 47 – Exécution des différents Jobs cités au niveau du Jenkinsfile

7.2 STRUCTURE DES HELM CHARTS

Les charts sont regroupés dans les répertoires avec les fichiers sous l'arborescence suivante :

- templates/ : ce répertoire contient les fichiers de modèle combinés avec les valeurs de configuration (à partir de + values.yaml + et de la ligne de commande) et restitués dans les manifestes Kubernetes. Les modèles utilisent le format de modèle du langage de programmation Go.
- Charts/ : facultative, les dépendances de diagramme gérées manuellement peuvent être placées dans ce répertoire, bien qu'il soit généralement préférable d'utiliser + Requirements.yaml + pour lier dynamiquement les dépendances. crds : facultative, Kubernetes Custom Resource Definitions
- Chart.yaml : un fichier YAML avec des métadonnées sur le chart, telles que le nom et la version du chart, des informations sur le responsable, un site Web pertinent et des mots clés de recherche.
- README.md : un fichier Lisez-moi contenant des informations pour les utilisateurs du chart.
- LICENSE : une licence en texte clair pour la carte.
- values.yaml : un fichier YAML contenant les valeurs de configuration par défaut du chart.

Helm File Structure

Wordpress Example

```
Wordpress/
├── Chart.yaml                #Required
├── LICENSE
├── values.yaml              #Required
├── values.schema.json
├── charts/                  #Required
│   └── dependent charts
├── crds/
│   └── needed crds
└── templates/              #Required
    ├── deployment.yaml
    ├── ingress.yaml
    ├── service.yaml
    ├── NOTES.txt
    └── tests/
        └── test-connection.yaml
```

FIGURE 48 – Exemple structure Helm Chart de Wordpress

8 BIBLIOGRAPHIE WEBOGRAPHIE

- 1 . Jenkins documentation : <https://www.jenkins.io/doc>
- 2 . GitLab documentation : <https://docs.gitlab.com>
- 3 . Docker documentation : <https://docs.docker.com>
- 4 . Kubernetes documentation : <https://kubernetes.io/fr/docs/home>
- 5 . Fluent-bit documentation : <https://docs.fluentbit.io>
- 6 . CI-CD documentation : <https://www.redhat.com/fr/topics/devops/what-is-ci-cd>
- 7 . Elastic Stack documentation : <https://www.elastic.co/guide/index.html>
- 8 . Jira Software : <https://fr.atlassian.com/software/jira>
- 9 . SonarQube documentation : <https://docs.sonarqube.org/latest>
- 10 . Archotecture Helm : <https://v2.helm.sh/docs/architecture>
- 11 . Journalisation d'évènements (logging) <https://kubernetes.io/fr/docs/concepts/cluster-administration/logging/>