# Réalisation d’une plateforme d’Intégration continue

Ci-dessous est décrite la plateforme realisé en sein du cet projét.

Cette plateforme utilise Jenkins pour la création d’une pipeline d’Intégration continue complète pour gérer les phases de développement et des test (tests unitaires et tests d’acceptance) d’une simple application web développée en Java; et Ansible pour la partie de déployement.

Ce permet d’avoir un outil à utiliser soit pour l’intégration d' applications plus complexes, soit pour se pratiquer dans les cadres du Continuous Delivery et du Cloud (Ansible, Jenkins, Docker, Kubernetes...).

## Web application

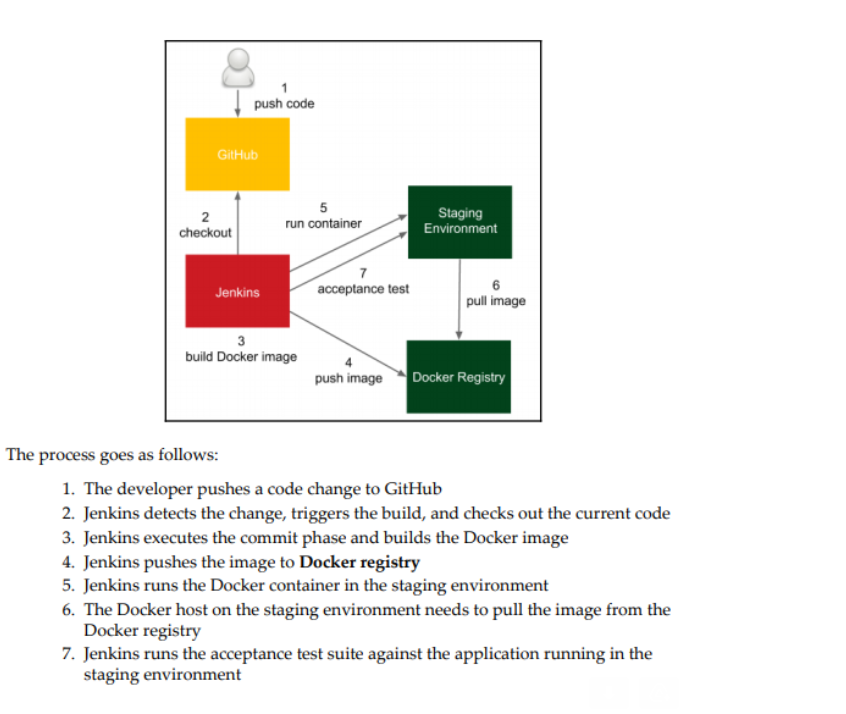
L'application Java ici utilisée est un simple web app réalisé en Java 8 utilisant Spring et Gradle.

Comme son nom le suggere, “Calculator-Hazelcast” permet de calculer le somme de 2 nombres. Hazelcast est l’outil utilisé pour offrir un service de caching pour le calculateur. Au moyen de Hazelcast, chaque fois que la méthode pour calculer la somme est appellé, Spring enregistre dans la cache le resultat. Un délai de 10 secondes permet de vérifier l'utilisation de la cache quand la même somme est appelée plusieurs fois.

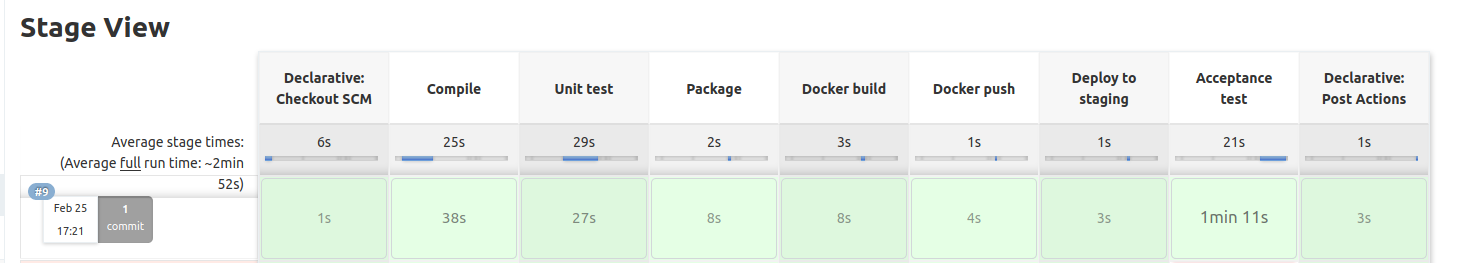
Pour une application aussi simple, un service de caching n’est pas d’une grande utilité. Le but est de déployer Hazelcast sur un serveur separé, par exemple pour être partagé avec autres applications et environnements (test/preprod/prod) et rendre de cet façon le plus intéressant la partie de déploiement avec Ansible.

## Pipeline d’intégration continue

Ci dessous un schéma décrivant la pipeline realisé:



Le 7 phases détaillées sont exécuté par 7 stages de la pipeline:



Ci-dessous la pipeline (Jenkinsfile):

pipeline {

agent any

stages {

stage("Compile") {

steps {

sh "./gradlew compileJava"

}

}

stage("Unit test") {

steps {

sh "./gradlew test"

}

}

stage("Package") {

steps {

sh "./gradlew build"

}

}

stage("Docker build") {

steps {

sh "docker build -t localhost:5000/calculator ."

}

}

stage("Docker push") {

steps {

sh "docker push localhost:5000/calculator"

}

}

stage("Deploy to staging") {

steps {

sh "docker run -d --rm -p 8765:8080 --name calculator localhost:5000/calculator"

}

}

stage("Acceptance test") {

steps {

sleep 60

sh "chmod +x acceptance\_test.sh && ./acceptance\_test.sh"

}

}

}

post {

always {

sh "docker stop calculator"

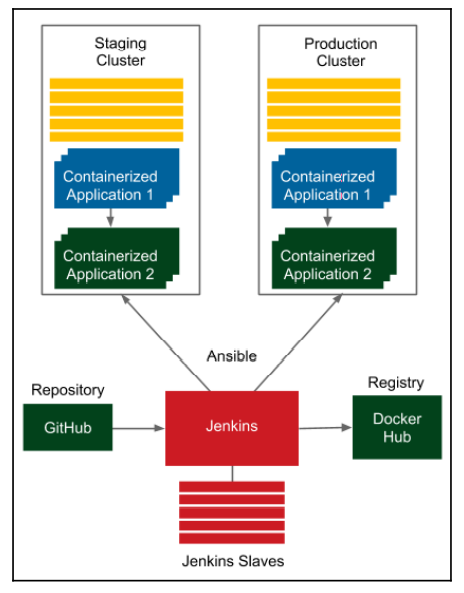
}

}

}

## Déploiement avec Ansible

Ici le schéma qui illustre l’utilisation de Ansible pour déployer l’application et le service Hazelcast:



Ci-dessous les détails des playbooks realisés:

* **Playbook-calc:**
  + Le premier tâche du playbook permet de préparer l'environnement en vérifiant que Java Runtime Environment soit installé (et de l’installer dans le cas que il ne le soit pas)
  + Après le dossier “calculator” est créé sur le hçote cible
  + Dans le dossier crée, le fichier “calculator.sh” est copié. Cet script serve pour lancer l’application
  + Par moyen du script “calculator.sh”, l’application est configurée comme une service Unix
  + Enfin le fichier .jar correspondant à l’application est copié dans le dossier cible, et le service est re-démarré par moyen d’un trigger (handler)
* **Playbook-hazel:**
  + Même que pour l’app calculateur, le premier tâche du playbook permet de préparer l'environnement en vérifiant que Java Runtime Environment soit installé (et de l’installer dans le cas que il ne le soit pas)
  + Hazelcast est téléchargé (une fichier JAR) pour être installé
  + Aprés Hazelcast est configuré comme une service Unix, de façon qu’il puisse être géré de la façon standard
  + Enfin le service Hazelcast est démarré