**Project – 2**

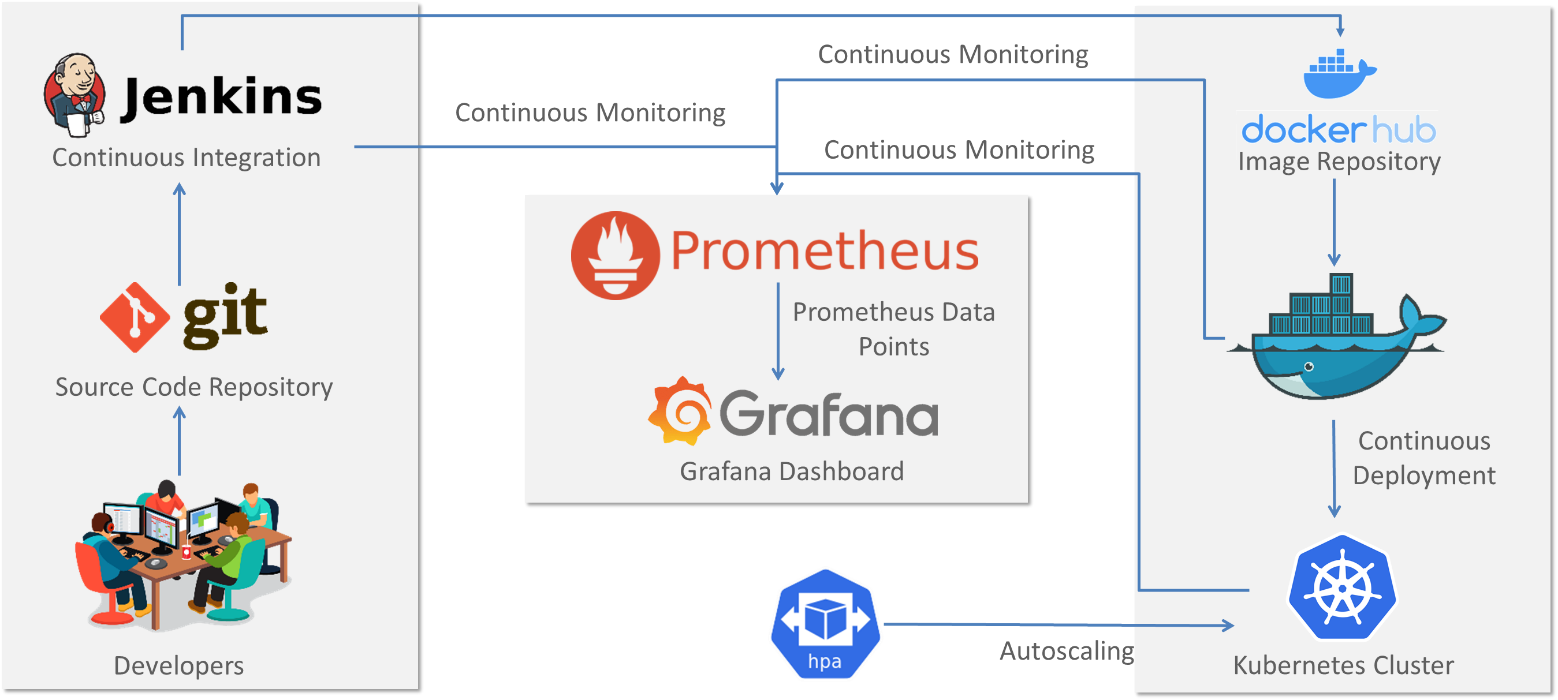
**Done By Bhavani Raju**

## Problem Statement

A retail company Abstergo Corp. has recently setup an online shopping portal(website) to sell their products. Due to fierce competition, the company wants a solution that can reduce the time and effort it needs to enhance the functionality of their website on a regular basis. They are looking for an automated way to deploy the new code (for new features) to production website whenever they want.

## Business Requirements

* The team of developers working on new features will merge their code to a GitHub repo.
* As soon as the code reaches GitHub, using a CI (Continuous Integration) pipeline, setup in Jenkins, automated builds will be triggered.
* The automated builds will frequently deploy new features to the production website.
* Every build will prepare a Dockerfile and push docker images to docker-hub.
* Every docker image will be deployed (Continuous Deployment) to a kubernetes-cluster.

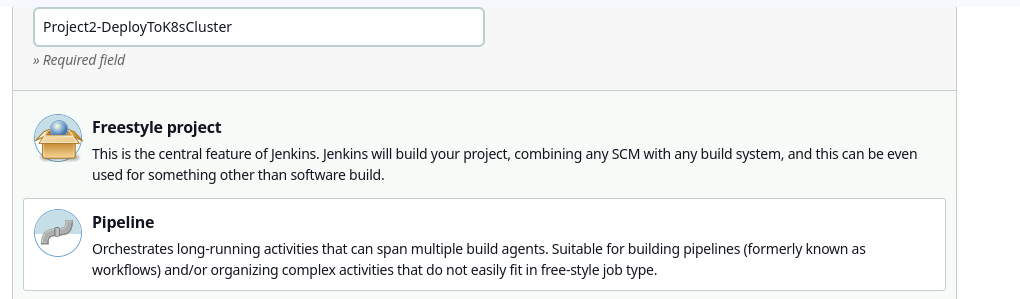


Fork the given repository to your own account and use it as the application for your pipeline project

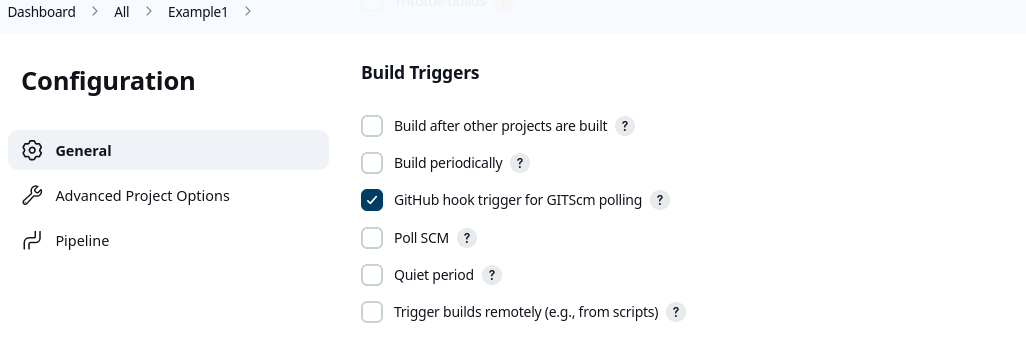
**GitHub:** <https://github.com/bhavukm/cicd-pipeline-train-schedule-autodeploy>

**Steps for executing the solution:**

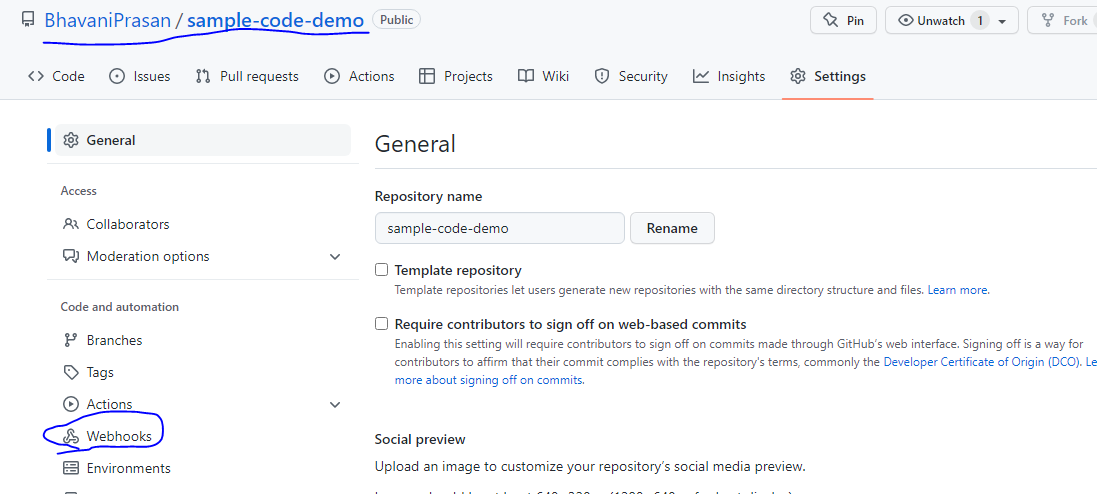
1. Create a New **pipeline** project.

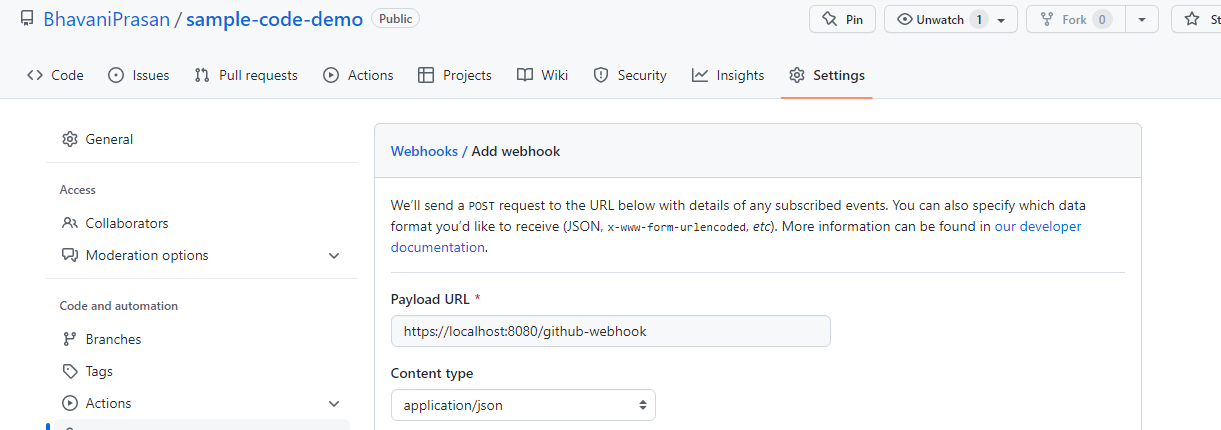


1. This build has to triggered when any push event occurs in master, so select this “GitHub hook trigger for GITSCM polling”



In Git hub, go to your repo, select web hook,





1. The problem here is that your localhost Jenkins server (without any public IP address) is sitting behind a firewall and/or NAT, so the GitHub webhook cannot be delivered to your localhost Jeninks server.
2. So I triggered the job manually.
3. Under pipeline section, write this code,



pipeline {

agent any

environment{

DOCKER\_IMAGE\_NAME = "bhavaniprasan/addressimg:latest"

}

stages{

stage('Clone Repo') {

steps{

git 'https://github.com/BhavaniPrasan/sample-code-demo.git'

}

}

stage('Compile code') {

steps{

sh 'mvn compile'

}

}

stage('Package the code') {

steps{

sh 'mvn package'

}

}

stage('Build Docker image') {

steps {

script {

app = docker.build("bhavaniprasan/addressimg")

app.inside {

sh 'echo Hello, World!'

}

}

}

}

stage('Push Docker Image') {

steps {

script {

docker.withRegistry('https://registry.hub.docker.com', 'docker\_hub\_login') {

app.push("${env.BUILD\_NUMBER}")

app.push("latest")

}

}

}

}

stage('DeployToProduction') {

steps {

kubernetesDeploy(

kubeconfigId: 'kubeconfig',

configs: 'replicaset.yml',

enableConfigSubstitution: true

)

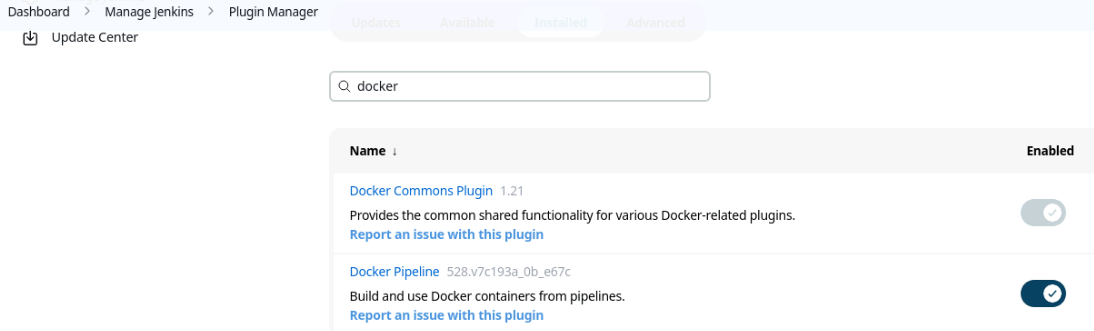
}

}

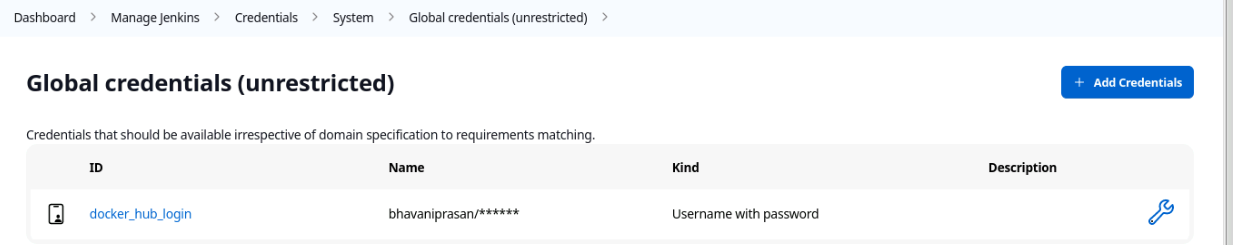
}

}

1. Install docker pipeline related plugins and kubernetes continuous deploy plugin from manage plugins.



1. Add credentials in order to communicate with Docker hub and kubernetes.



1. As kubernetes continuous deploy plugin is suspended, could not deploy through Jenkins pipeline. So doing manually in edureka lab.
2. Deployment.yml

----------------------

kind: Deployment

apiVersion: apps/v1

metadata:

name: kubeserve

spec:

replicas: 3

minReadySeconds: 10 # wait for 45 sec before going to deploy next pod

strategy:

type: RollingUpdate

rollingUpdate:

maxUnavailable: 1

maxSurge: 1 # max number of pods to run for the deployment

selector:

matchLabels:

app: kubeserve

template:

metadata:

name: kubeserve

labels:

app: kubeserve

spec:

containers:

- name: app

image: bhavaniprasan/addressimg:latest

---

kind: Service

apiVersion: v1

metadata:

name: kubeserve-svc

spec:

type: NodePort

ports:

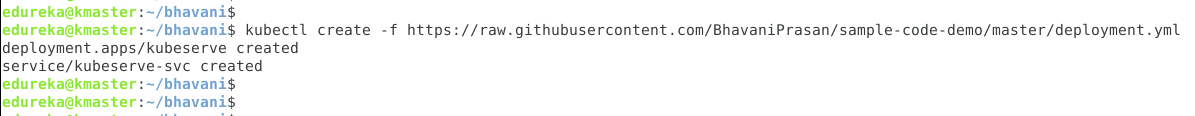
- port: 80

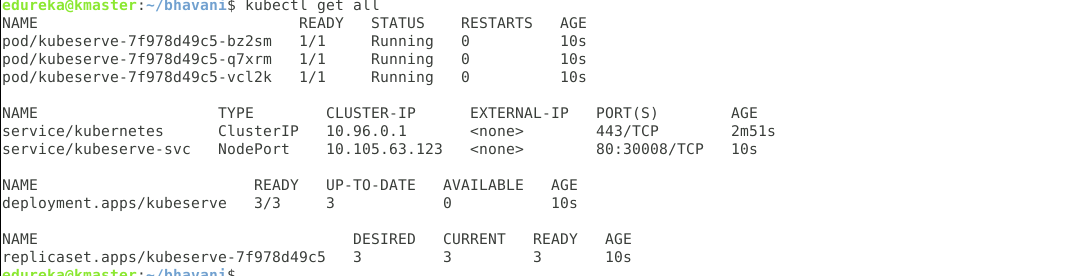
targetPort: 80

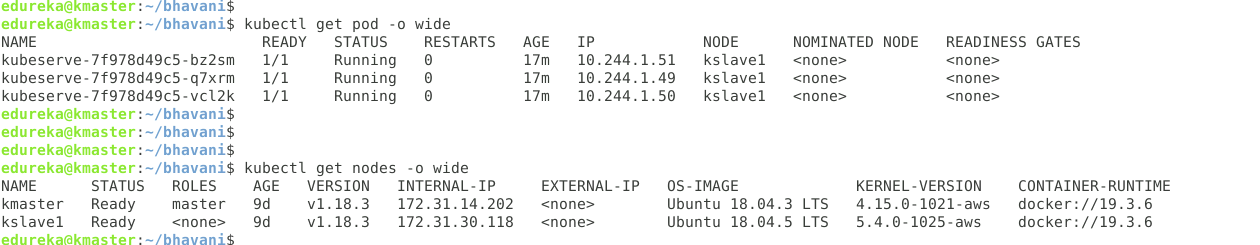
nodePort: 30008

selector:

app: kubeserve







Kubectl delete all –all (command to delete all nodes,service, replicaset)