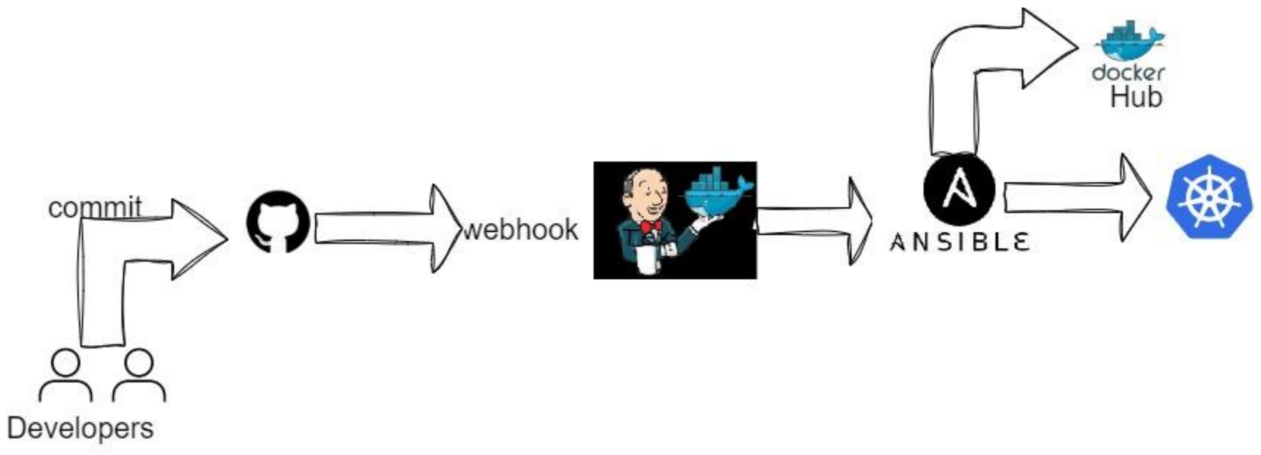
Deployment on Kubernetes cluster using Jenkins CI/CD

In This article I will show you how to deploy Kubernetes cluster using Jenkins CI/CD pipeline. In this demo project we are taking help of various DevOps tools like GitHub, Jenkins, Ansible, Docker Hub and Kubernetes Minikube cluster.



Let's go through the contents of DevOps tools uses:

Developer: Developer will write source code and docker file and push into the code to GitHub repository.

GitHub: GitHub is a code hosting platform for version control and collaboration. And it will store our source code into a repository.

Webhook: Webbook informs Jenkins whenever new code is available and tells Jenkins to build the new code.

Jenkins: Jenkins will pull out the code from GitHub repository, then execute the CI/CD pipeline.

Ansible: Ansible server will execute the Kubernetes deployment and service yaml files to deploy the application.

Kubernetes: Pods and application will run on Kubernetes cluster.

Prerequisite:

GitHub, Jenkins, Docker, Jenkins, Docker Hub Account, Ansible, Kubernetes (deployment & services yaml files).

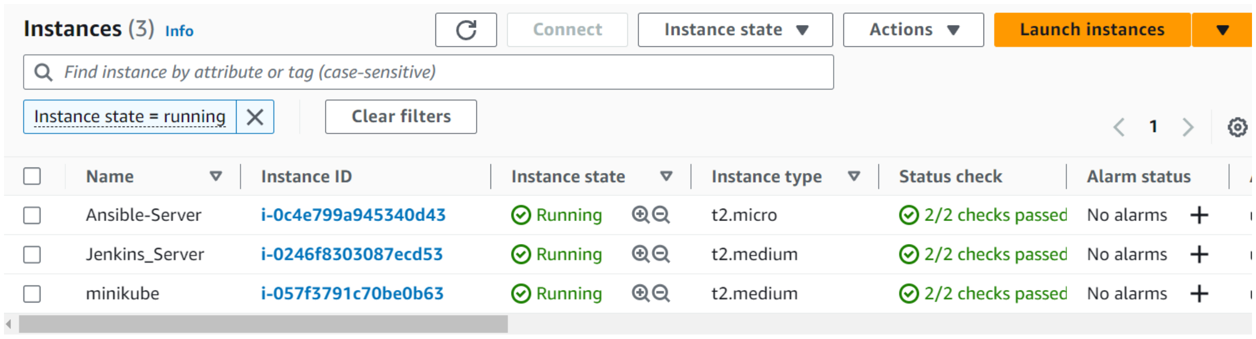
3 EC2 Ubuntu Instances required:

1. Jenkins (default-jre+Jenkins), t2.Medium+20GB

2. Ansible (python+ansible+docker), t2.micro + 8GB

3. Webapp, Kubernetes Cluster (docker+minikube), t2.Medium+20GB

Signed In AWS Console and created three Ubuntu Instances.



Name	Instance ID	Instance state	Instance type	Status check	Alarm status
Ansible-Server	i-0c4e799a945340d43	Running	t2.micro	2/2 checks passed	No alarms
Jenkins_Server	i-0246f8303087ecd53	Running	t2.medium	2/2 checks passed	No alarms
minikube	i-057f3791c70be0b63	Running	t2.medium	2/2 checks passed	No alarms

How to install Jenkins on Ubuntu: t2.Medium+20GB

# sudo apt update

# sudo apt install openjdk-11-jdk

# java --version

Jenkins can easily be installed on Ubuntu by importing and adding the GPG keys to the system.

Now you got to add GPG keys:

# wget -p -O - https://pkg.jenkins.io/debian/jenkins.io.key | sudo apt-key add –

After adding GPG keys, add the Jenkins package address to the sources list by typing the command given below:

# sudo apt update

Let's move forward and do the real work of installing Jenkins.

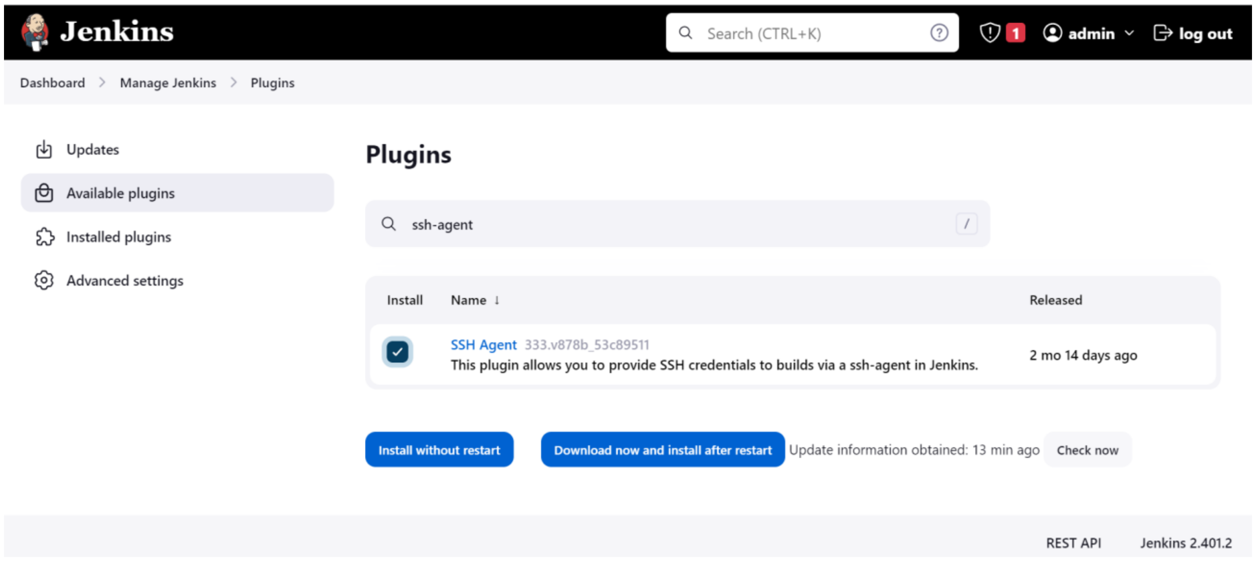
# sudo apt install Jenkins

# sudo systemctl status Jenkins

# sudo systemctl start Jenkins

After started Jenkins install ssh-agent plugin using below steps:

Jenkins → Dashboard→ Manage Jenkins → Plugins→ click on available plugins→ search(ssh-agent)→click on Install without restart.



Plugins

ssh-agent

Install	Name ↓	Released
✓	SSH Agent 333.v878b_53c89511 This plugin allows you to provide SSH credentials to builds via a ssh-agent in Jenkins.	2 mo 14 days ago

Update information obtained: 13 min ago

Install Ansible on Ubuntu server: t2.micro + 8GB

Connect to ansible ubuntu instance and execute below commands as normal user.

```
# sudo apt-add-repository ppa:ansible/ansible -y

# sudo apt update -y

# sudo apt install ansible -y

# ansible –version
```

Docker Installation steps.

```
apt install docker.io -y
usermod -aG docker ubuntu
systemctl restart docker
systemctl enable docker.service
```

Install minikube cluster with docker: t2.Medium+20GB

Run the following commands to update all system packages to the latest release:

```
sudo apt update

sudo apt upgrade
```

If a reboot is required after the upgrade then perform the process

```
[ -f /var/run/reboot-required ] && sudo reboot -f
```

Docker Installation steps.

```
apt install docker.io -y
usermod -aG docker ubuntu
systemctl restart docker
systemctl enable docker.service
```

Download Minikube on Ubuntu

You need to download the minikube binary. I will put the binary under /usr/local/bin directory since it is inside $PATH.

```
# wget https://storage.googleapis.com/minikube/releases/latest/minikube-linux-amd64

# chmod +x minikube-linux-amd64

# sudo mv minikube-linux-amd64 /usr/local/bin/minikube
```

Check the minikube version using below command

```
# minikube version
```

Install kubectl on Ubuntu

We need kubectl which is a command line tool used to deploy and manage applications on Kubernetes.

```
curl -LO https://storage.googleapis.com/kubernetes-release/release/`curl -s https://storage.googleapis.com/kubernetes-release/release/stable.txt`/bin/linux/amd64/kubectl
```

Make the kubectl binary executable

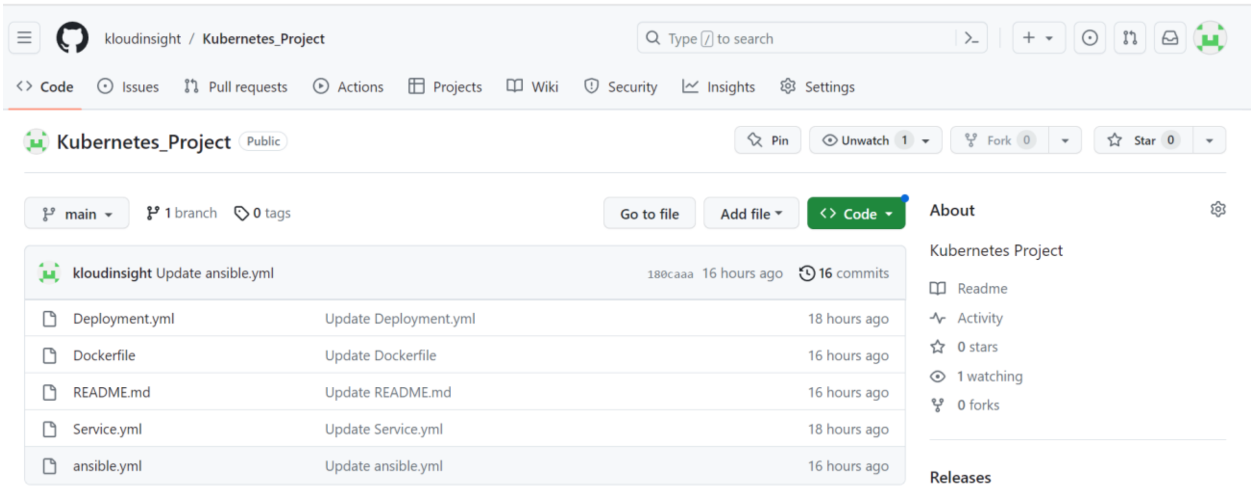
```
# chmod +x ./kubectl

# Move the binary in to your PATH:

# sudo mv ./kubectl /usr/local/bin/kubectl

# sudo usermod -aG docker $USER && newgrp docker
```

Starting minikube on Ubuntu

Now that components are installed, you can start minikube. VM image will be downloaded and configured for Kubernetes single node cluster.

# minikube start

Login to GitHub remote repository create below listed Dockerfile, deployment.yml, service,yml and ansible.yml files.


kloudinsight / Kubernetes_Project

Kubernetes_Project Public

main · 1 branch · 0 tags

kloudinsight Update ansible.yml 180caaa 16 hours ago 16 commits

Deployment.yml	Update Deployment.yml	18 hours ago
Dockerfile	Update Dockerfile	16 hours ago
README.md	Update README.md	16 hours ago
Service.yml	Update Service.yml	18 hours ago
ansible.yml	Update ansible.yml	16 hours ago

About

Kubernetes Project

Readme · Activity · 0 stars · 1 watching · 0 forks

Releases

Dockerfile

```
FROM  centos:7
MAINTAINER VenkatKumar
RUN yum install -y httpd \
 zip\
 unzip
ADD https://www.free-css.com/assets/files/free-css-
templates/download/page254/photogenic.zip /var/www/html/
WORKDIR /var/www/html/
RUN unzip photogenic.zip
RUN cp -rvf photogenic/* .
RUN rm -rf photogenic photogenic.zip
CMD ["/usr/sbin/httpd", "-D", "FOREGROUND"]
EXPOSE 80
```

Deployment.yml

```
kind: Deployment
apiVersion: apps/v1
metadata:
   name: kloudinsight
spec:
   replicas: 2
   selector:      # tells the controller which pods to watch/belong to
    matchLabels:
     app: kloudinsight
   template:
      metadata:
        labels:
          app: kloudinsight
      spec:
       containers:
        - name: kloudinsight
          image: kloudinsight/pipeline-demo
          imagePullPolicy: Always
          ports:
          - containerPort: 80
```

Service.yml

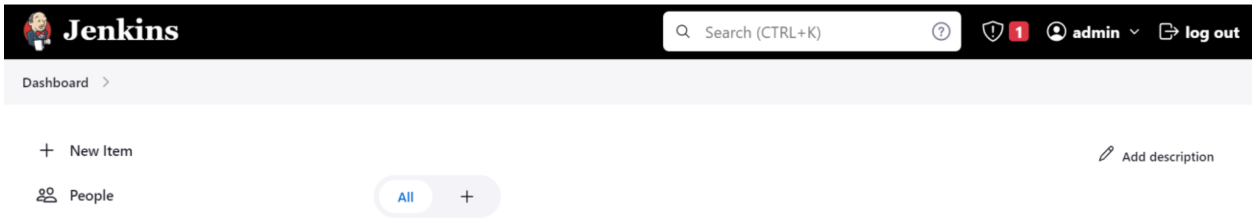
```
kind: Service
apiVersion: v1
metadata:
  name: kloudinsight
  labels:
    app: kloudinsight
spec:
  ports:
    - port: 8080
      targetPort: 80
      nodePort: 31200
  selector:
    app: kloudinsight
  type: LoadBalancer
```

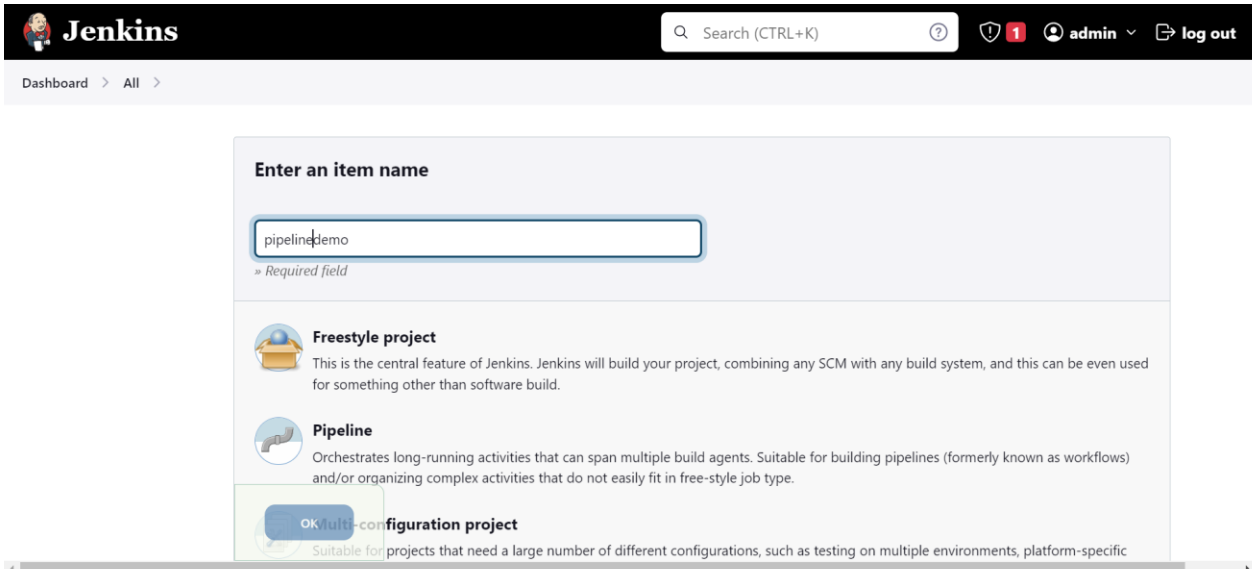
ansible.yml

```
- hosts: all

 tasks:

   - name: create new deployment

     command: kubectl apply -f /home/ubuntu/deployment.yml

   - name: create new service

     command: kubectl apply -f /home/ubuntu/service.yml
```

Pipeline Script Steps:

Login into Jenkins server and click on New Item, Enter pipeline name and select Pipeline and click on OK.





Enter an item name

pipelinedemo

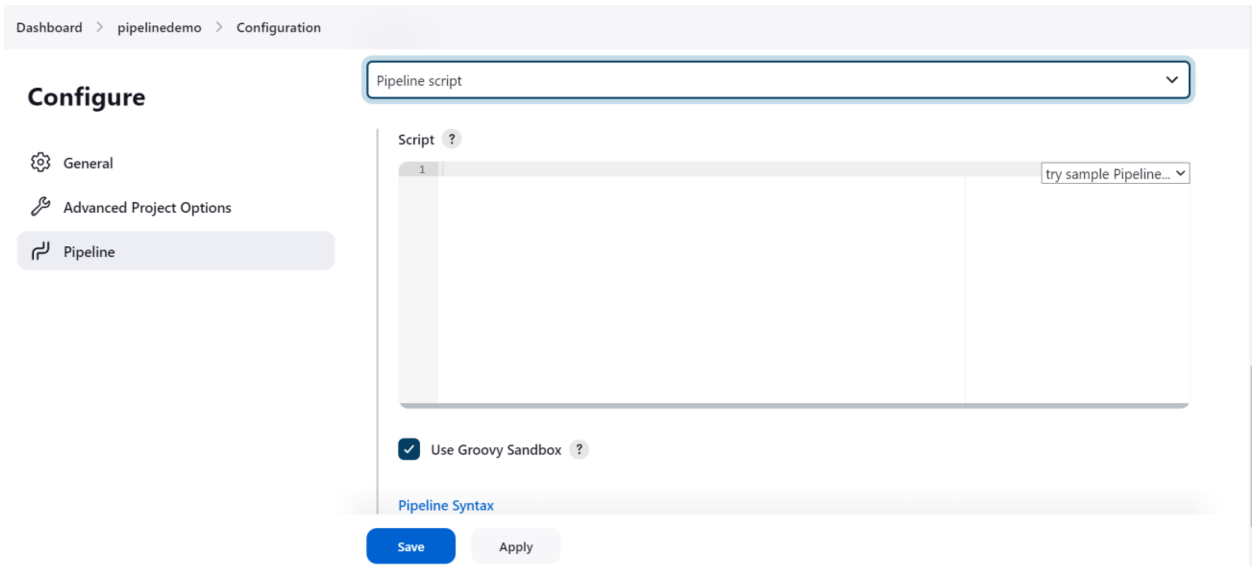
» Required field

Freestyle project
This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.

Pipeline
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

Multi-configuration project
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific

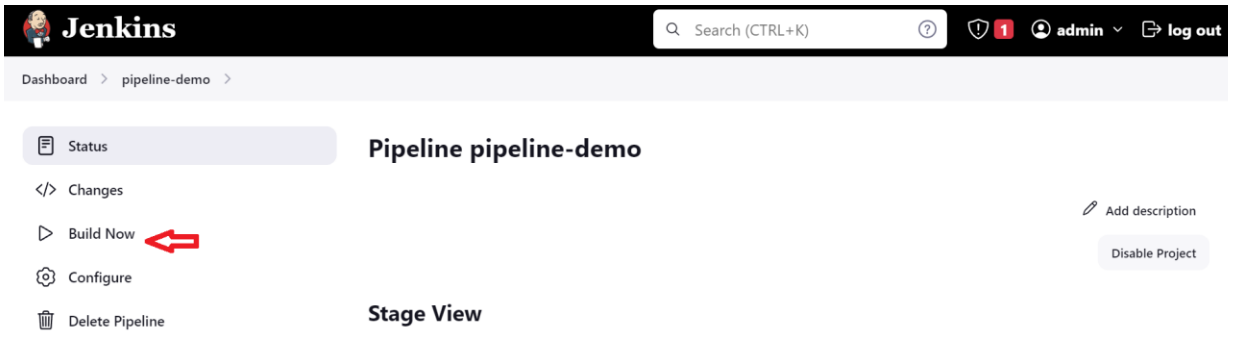
Navigate to pipeline configuration select pipeline script and copy the pipeline groovy script then click on apply and save.



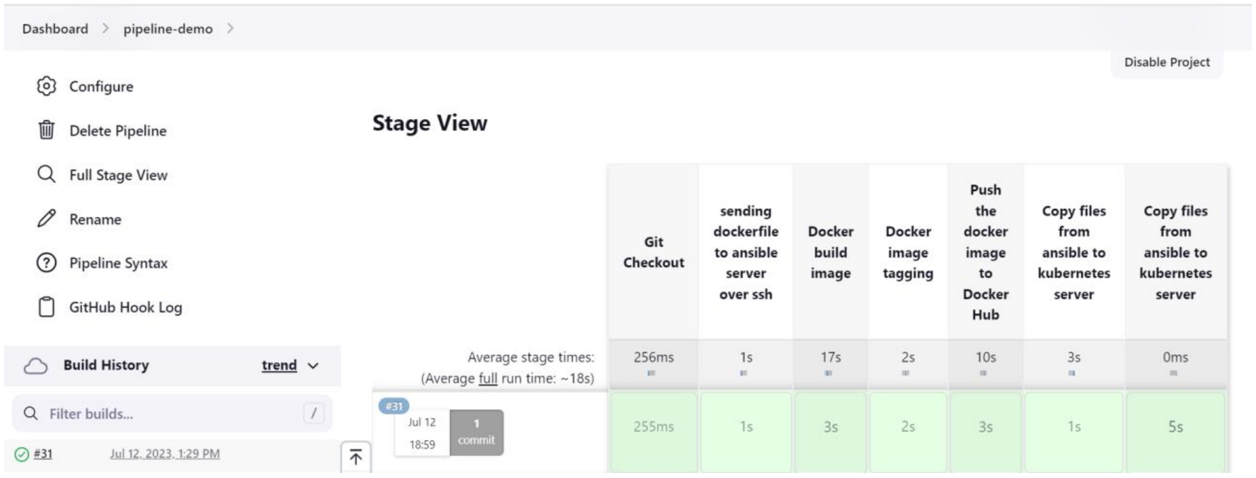
1. **Replace your ansible server username and IP address**(ubuntu@172.31.18.35)
2. **Replace your kubernetes server username and IP address** (ubuntu@172.31.81.94).

```
node {
  stage('Git Checkout'){
    git branch: 'main', url: 'https://github.com/kloudinsight/Kubernetes_Project.git'
  }
  stage('sending dockerfile to ansible server over ssh'){
    sshagent(['ansibledemo']) {
     sh 'ssh -o StrictHostKeyChecking=no ubuntu@172.31.18.35'
     sh 'scp /var/lib/jenkins/workspace/pipeline-demo/* ubuntu@172.31.18.35:/home/ubuntu/'
    }
  }
  stage('Docker build image'){
    sshagent(['ansibledemo']){
      sh 'ssh -o StrictHostKeyChecking=no ubuntu@172.31.18.35 cd /home/ubuntu/'
      sh 'ssh -o StrictHostKeyChecking=no ubuntu@172.31.18.35 docker image build -t $JOB_NAME:v1.$BUILD_ID .'
    }
  }
  stage('Docker image tagging'){
    sshagent(['ansibledemo']){
          sh 'ssh -o StrictHostKeyChecking=no ubuntu@172.31.18.35 cd /home/ubuntu/'

          sh 'ssh -o StrictHostKeyChecking=no ubuntu@172.31.18.35 docker image tag $JOB_NAME:v1.$BUILD_ID
kloudinsight/$JOB_NAME:v1.$BUILD_ID'

          sh 'ssh -o StrictHostKeyChecking=no ubuntu@172.31.18.35 docker image tag $JOB_NAME:v1.$BUILD_ID
kloudinsight/$JOB_NAME:latest'


        }

    }

    stage ('Push the docker image to Docker Hub'){

        sshagent(['ansibledemo']){

            withCredentials([string(credentialsId: 'dockehubpasswd', variable: 'dockehubpasswd')]) {

                sh "ssh -o StrictHostKeyChecking=no ubuntu@172.31.18.35 docker login -u kloudinsight -p ${dockehubpasswd}"

                sh 'ssh -o StrictHostKeyChecking=no ubuntu@172.31.18.35 docker image push kloudinsight/$JOB_NAME:v1.$BUILD_ID'

                sh 'ssh -o StrictHostKeyChecking=no ubuntu@172.31.18.35 docker image push kloudinsight/$JOB_NAME:latest'

            }

        }

    }

     stage ('Copy files from ansible to kubernetes server'){

         sshagent(['ansibledemo']){

             sh 'ssh -o StrictHostKeyChecking=no ubuntu@172.31.81.94 cd /home/ubuntu/'

             sh 'scp /var/lib/jenkins/workspace/pipeline-demo/* ubuntu@172.31.81.94:/home/ubuntu/'


         }

     }

     stage ('Copy files from ansible to kubernetes server'){

         sshagent(['ansibledemo']){

             sh 'ssh -o StrictHostKeyChecking=no ubuntu@172.31.18.35 cd /home/ubuntu/'

             sh 'ssh -o StrictHostKeyChecking=no ubuntu@172.31.18.35 ansible-playbook ansible.yml'

         }

     }

}
```

Login into Jenkins server navigate to pipeline click on build now.

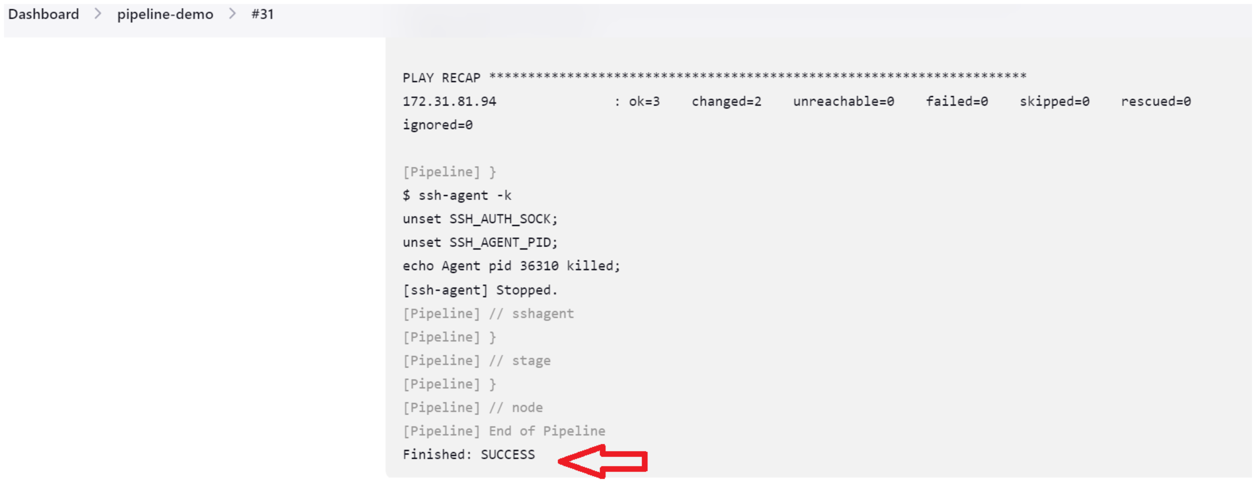


Pipeline Demo project build success.



	Git Checkout	sending dockerfile to ansible server over ssh	Docker build image	Docker image tagging	Push the docker image to Docker Hub	Copy files from ansible to kubernetes server	Copy files from ansible to kubernetes server
Average stage times: (Average full run time: ~18s)	256ms	1s	17s	2s	10s	3s	0ms
#31 Jul 12 18:59 1 commit	255ms	1s	3s	2s	3s	1s	5s

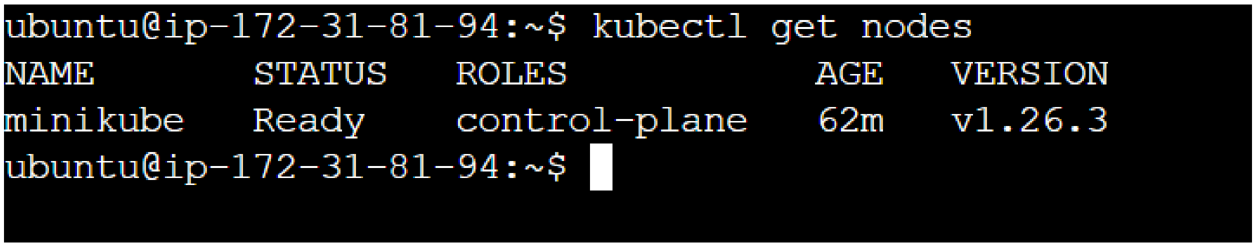
Check the pipeline demo console output / logs.



```
PLAY RECAP *********************************************************************
172.31.81.94               : ok=3    changed=2    unreachable=0    failed=0    skipped=0    rescued=0
ignored=0

[Pipeline] }
$ ssh-agent -k
unset SSH_AUTH_SOCK;
unset SSH_AGENT_PID;
echo Agent pid 36310 killed;
[ssh-agent] Stopped.
[Pipeline] // sshagent
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```

Connect to kubernetes cluster machine and check the minikube cluster is ready or not execute the below command

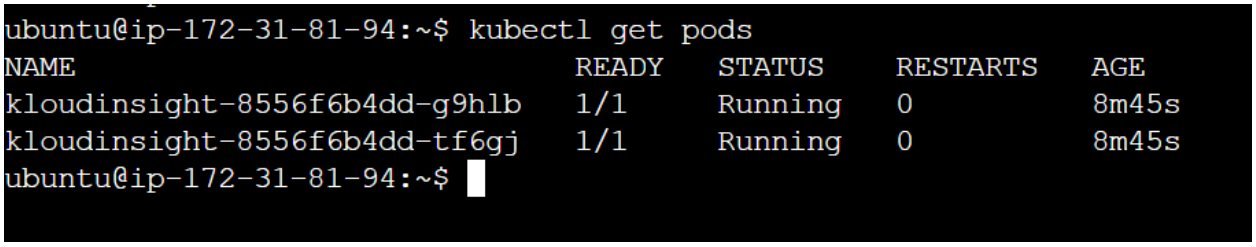
# kubectl get nodes



```
ubuntu@ip-172-31-81-94:~$ kubectl get nodes
NAME       STATUS   ROLES           AGE   VERSION
minikube   Ready    control-plane   62m   v1.26.3
ubuntu@ip-172-31-81-94:~$
```

Verify the pods running or not execute the below command.

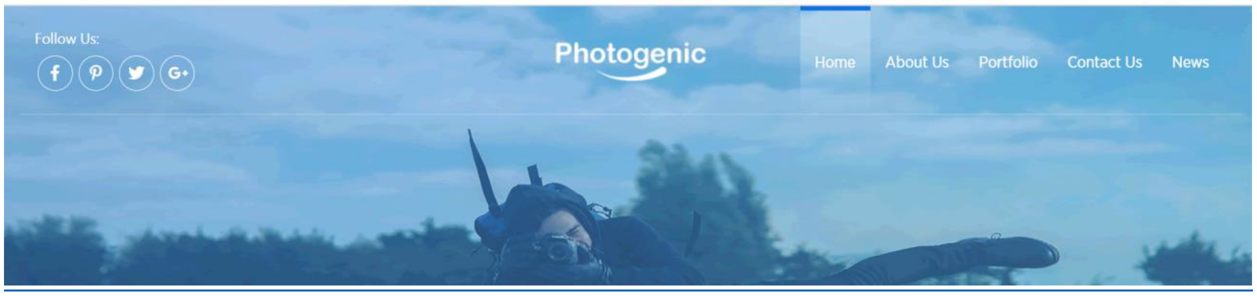
# kubectl get pods



```
ubuntu@ip-172-31-81-94:~$ kubectl get pods
NAME                             READY   STATUS    RESTARTS   AGE
kloudinsight-8556f6b4dd-g9hlb    1/1     Running   0          8m45s
kloudinsight-8556f6b4dd-tf6gj    1/1     Running   0          8m45s
ubuntu@ip-172-31-81-94:~$
```

Verify the application home page.

http://54.144.213.63:31200



---END---

Thank you for reading 😊

Venkat Kumar