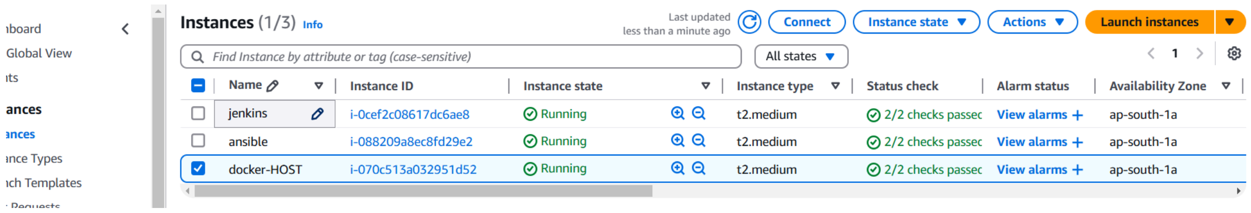
Jenkins Project :

Automated CI/CD pipeline for web application using Aws, git, GitHub, Jenkins, docker, Docker hub - A real time CI/CD pipeline on Jenkins with GitHub integration. -Creating webhook in GitHub that will trigger builds in Jenkins. - Create Docker file and build image and launch container using docker.

****

Step1 : Infrastructure Setup

Create 3 servers:
- -- Jenkins-Server
- -- Ansible-Server+dockerhub
- -- DockerHost



Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone
jenkins	i-0cef2c08617dc6ae8	Running	t2.medium	2/2 checks passec	View alarms +	ap-south-1a
ansible	i-088209a8ec8fd29e2	Running	t2.medium	2/2 checks passec	View alarms +	ap-south-1a
✓ docker-HOST	i-070c513a032951d52	Running	t2.medium	2/2 checks passec	View alarms +	ap-south-1a

****

Step2: Configure Jenkins-Server

--Install Jenkins with dependencies

```
-- yum update -y
-- yum install wget vim -y
-- yum install fontconfig java-17-openjdk -y
-- java -version
-- wget -O /etc/yum.repos.d/jenkins.repo \  https://pkg.jenkins.io/redhat-stable/jenkins.repo

-- rpm --import https://pkg.jenkins.io/redhat-stable/jenkins.io-2023.key
-- yum upgrade
-- yum install jenkins -y
-- systemctl daemon-reload
-- systemctl enable --now jenkins

-- systemctl status jenkins
```

--Port allow
-- 8080 in security group

-- Access Jenkins through web browser
-- <jenkins-ip:8080>

-- Unlock Jenkins
-- /var/lib/jenkins/secrets/initialAdminPassword >> get password >> paste password

****

Configure Ansible-Server

-- configure epel

```
-- yum update -y
-- subscription-manager repos --enable codeready-builder-for-rhel-9-$(arch)-rpms
-- dnf install https://dl.fedoraproject.org/pub/epel/epel-release-latest-9.noarch.rpm -y
```

-- Install ansible

```
-- yum install ansible* -y
-- ansible --version
```

-- setup ansible directory

```
-- mkdir ansible
-- cd ansible
-- touch ansible.cfg inventory cloud.pem
```

-- Configure ansible.cfg file

-- vim ansible.cfg

```
[defaults]
inventory=/root/ansible/inventory
remote_user=ec2-user
ask_pass=false
private_key_file=/root/ansible/cloud.pem
host_key_checking=false
```

```
[privilege_escalation]
become=true
become_method=sudo
become_user=root
become_ask_pass=false
```

-- add dockerhost ip in inventory file

-- vim inventory

```
[docker]
3.110.196.25
```

-- Add key in file

-- vim cloud.pem

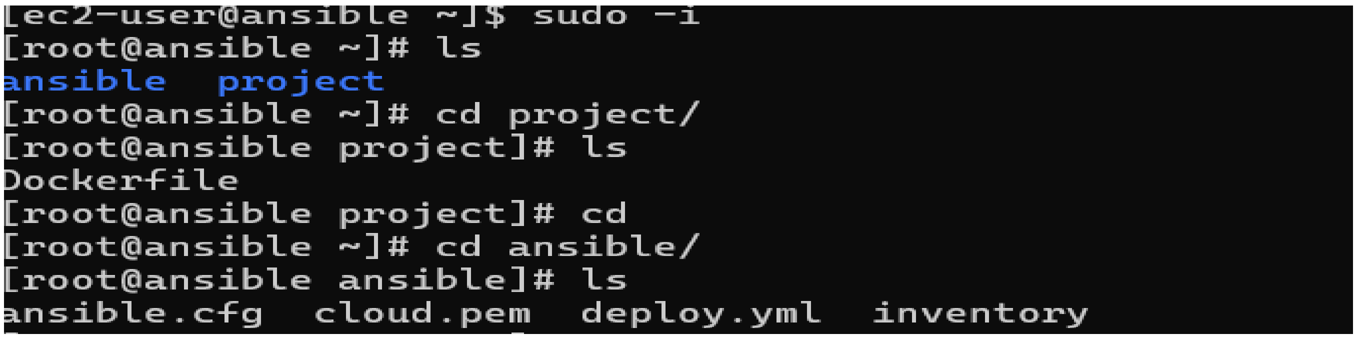
-- Change permission of key file

-- chmod 400 cloud.pem

-- Test connectivity with docker host

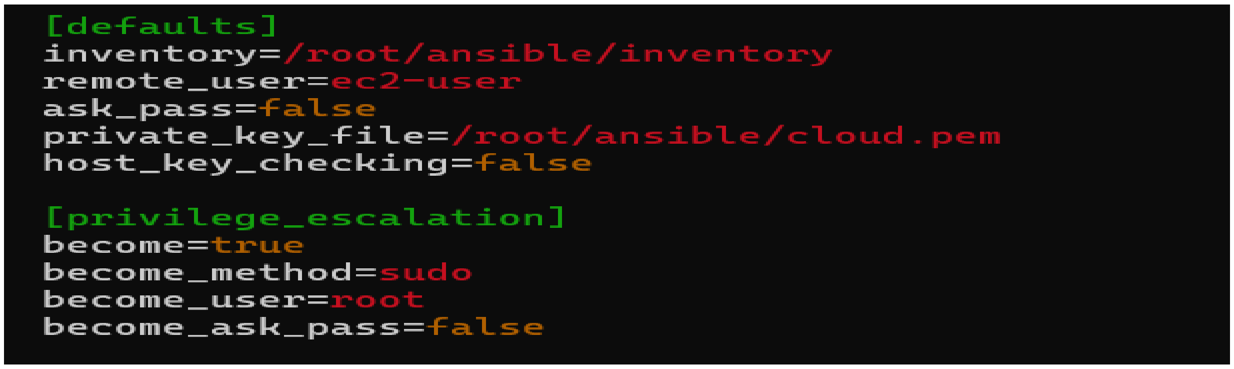
-- ansible all -m ping

****



```
[ec2-user@ansible ~]$ sudo -i
[root@ansible ~]# ls
ansible  project
[root@ansible ~]# cd project/
[root@ansible project]# ls
Dockerfile
[root@ansible project]# cd
[root@ansible ~]# cd ansible/
[root@ansible ansible]# ls
ansible.cfg  cloud.pem  deploy.yml  inventory
```

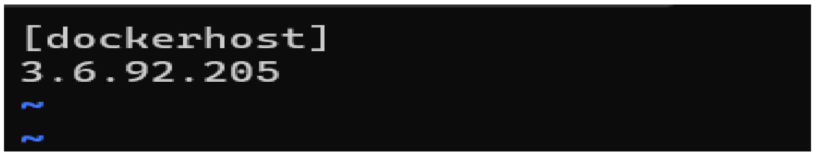
#ansible.cfg files entry :



```
[defaults]
inventory=/root/ansible/inventory
remote_user=ec2-user
ask_pass=false
private_key_file=/root/ansible/cloud.pem
host_key_checking=false

[privilege_escalation]
become=true
become_method=sudo
become_user=root
become_ask_pass=false
```

# inventory file : (paste public ip of dockerhost machine from aws)



```
[dockerhost]
3.6.92.205
~
~
```

****

Step3 : **Configure docker on Ansible-Server**

-- Docker install

-- dnf -y install dnf-plugins-core
-- dnf config-manager --add-repo https://download.docker.com/linux/rhel/docker-ce.repo
-- dnf install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin -y
-- systemctl enable --now docker
-- systemctl status docker

-- Login Docker

-- docker login -u vikash1269
password: 9079387608

****

Step4 : Configure dockerhost Server

-- Docker install

```
    -- dnf -y install dnf-plugins-core
    -- dnf config-manager --add-repo https://download.docker.com/linux/rhel/docker-ce.repo
    -- dnf install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-
plugin -y
    -- systemctl enable --now docker
    -- systemctl status docker
```

****

Step5 : Doing passwordless authentication b/w Jenkins & Ansible-server

**** switch to Jenkins Server**

-- enable root user on Jenkins server

```
  -- vim /etc/ssh/sshd_config
     PermitRootLogin yes
     PasswordAuthentication yes

  -- vim /etc/ssh/sshd_config.d/50-cloud-init.conf
     PasswordAuthentication yes
```

-- set root password

```
  passwd root
```

-- service restart

```
  systemctl restart sshd
```

****

-- switch to Ansible Server

-- enable root user on Ansible server

```
  -- vim /etc/ssh/sshd_config
     PermitRootLogin yes
     PasswordAuthentication yes

  -- vim /etc/ssh/sshd_config.d/50-cloud-init.conf
     PasswordAuthentication yes
```

-- set root password
  passwd root

-- service restart
  systemctl restart sshd

****

Step6 : Setup Passwordless authentication

-- key generate on Jenkins server

-- ssh-keygen

-- key copy from Jenkins server to ansible server:

-- ssh-copy-id root@ansibleserver-ip

-- check passwordless connectivity:

-- ssh root@ansibleserver-ip

****

Step7: Install publish over ssh plugin (On Jenkins Page:)

-- jenkins dashboard>manage>plugin>availableplugin>search>publish over ssh>install
  restart Jenkins
  again login Jenkins with user and password

SSH server add for Jenkins

-- Dashboard>ManageJenkins>System>ssh server>add
  name-->jenkins-root
  hostname--> Jenkins ip (public ip)
  username--> root
  advanced>Use password authentication>enter password of root user

-- Test configuration
  ensure output show **success**

SSH server add for Ansible

-- Dashboard>ManageJenkins>System>ssh server>add
  name-->ansible-root
  hostname--> ansible ip
  username--> root
  advanced>Use password authentication>enter password of root user

-- Test configuration
  ensure output show **success**

-- apply and save

****

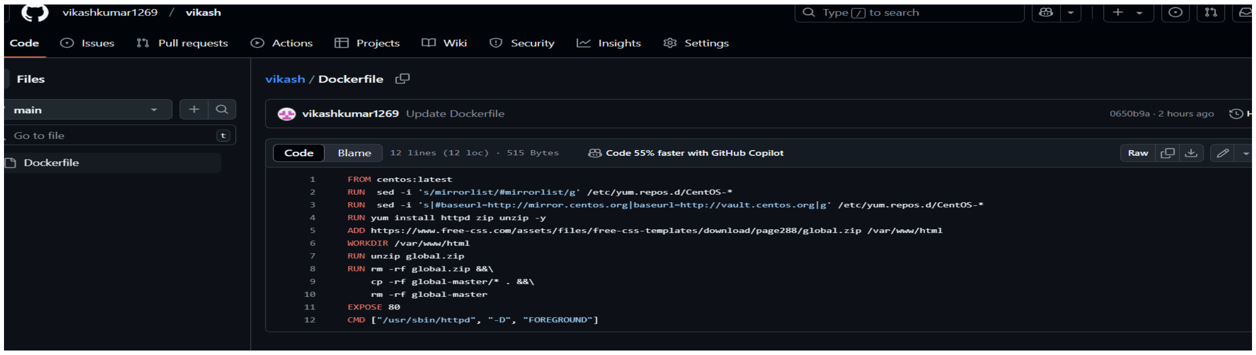
Step8: Configure GitHub

-- create repository
-- add docker file

Docker File

```
        FROM centos:latest
        RUN sed -i 's/mirrorlist/#mirrorlist/g' /etc/yum.repos.d/CentOS-*
        RUN sed -i 's|#baseurl=http://mirror.centos.org|baseurl=http://vault.centos.org|g'
/etc/yum.repos.d/CentOS-*
        RUN yum install httpd zip wget unzip -y
        RUN wget -O /var/www/html/applight.zip https://www.free-css.com/assets/files/free-css-
templates/download/page295/applight.zip
        WORKDIR /var/www/html
        RUN unzip applight.zip
        RUN cp -rf Applight/* . && \
        rm -rf Applight
        EXPOSE 80
        CMD ["/usr/sbin/httpd", "-D", "FOREGROUND"]
```

****



vikashkumar1269 Update Dockerfile 0650b9a · 2 hours ago

12 lines (12 loc) · 515 Bytes

```
FROM centos:latest
RUN  sed -i 's/mirrorlist/#mirrorlist/g' /etc/yum.repos.d/CentOS-*
RUN  sed -i 's|#baseurl=http://mirror.centos.org|baseurl=http://vault.centos.org|g' /etc/yum.repos.d/CentOS-*
RUN yum install httpd zip unzip -y
ADD https://www.free-css.com/assets/files/free-css-templates/download/page288/global.zip /var/www/html
WORKDIR /var/www/html
RUN unzip global.zip
RUN rm -rf global.zip &&\
    cp -rf global-master/* . &&\
    rm -rf global-master
EXPOSE 80
CMD ["/usr/sbin/httpd", "-D", "FOREGROUND"]
```

****

Step9 : Job(project) create on Jenkins page

--new item>enter name of job(project.. Eg :- Docker-project)>freestyle>ok

Configure job on Jenkins page :

--Source Code Management>git>Repository URL>select branch according to GitHub(*/main)
Build Triggers>GitHub hook trigger for GITScm polling>apply & save

Install git on Jenkins server

-- yum install git -y

****

Step10: Configure webhook on github :

go to GitHub>settings>webhook>add webhook>payload url(http://13.233.253.0:8080/github-webhook/)>add webhhok

**(http://13.233.253.0:8080→ jenkins url)

****

Step11 : Go On Jenkins Page :

Select : Send files or execute commands over SSH

SSH Server

Select : jenkins server

Exec command :

```
 rsync -avh /var/lib/jenkins/workspace/project-1/Dockerfile root@ansible-ip:/root/project
```

SSH Server

Select : ansible server

Exec command :

```
cd /root/project
docker build -t $JOB_NAME:v$BUILD_ID .
docker tag $JOB_NAME:v$BUILD_ID vikash1269/$JOB_NAME:latest
docker tag $JOB_NAME:v$BUILD_ID vikash1269/$JOB_NAME:v$BUILD_ID
docker push vikash1269/$JOB_NAME:latest
docker push vikash1269/$JOB_NAME:v$BUILD_ID
docker rmi -f $(docker images -q)


cd /root/ansible
ansible-playbook deploy.yml
```

****
Step12 : Go on Ansible server :

*Create a playbook under ansible directory : (eg : deploy.yml)

```
# vim deploy.yml :
---
 - name: Launch Docker Container
   hosts: all
   vars:
     docker_image: "vikash1269/docker-project"
     docker_tag: "latest"
     container_name: "vkcontainer"
     host_port: "80"
     container_port: "80"
   tasks:
     - name: Check if the container is already running
       docker_container:
         name: "{{ container_name }}"
```

```
        state: absent
      register: container_stats
      ignore_errors: yes

    - name: Remove old container if it exists
      docker_container:
        name: "{{ container_name }}"
        state: absent

    - name: Remove all Docker images
      shell: docker rmi -f $(docker images -q)

    - name: Pull the Docker image
      docker_image:
        name: "{{ docker_image }}"
        tag: "{{ docker_tag }}"
        source: pull

    - name: Run the new Docker Container
      docker_container:
        name: "{{ container_name }}"
        image: "{{ docker_image }}:{{ docker_tag }}"
        state: started
        published_ports: "{{ host_port }}:{{ container_port }}"
```

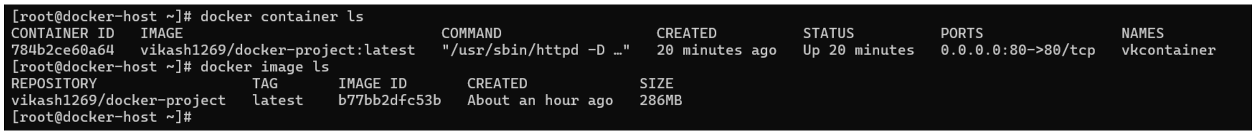
****

Now Go on Jenkins Page and click on **build now**

****

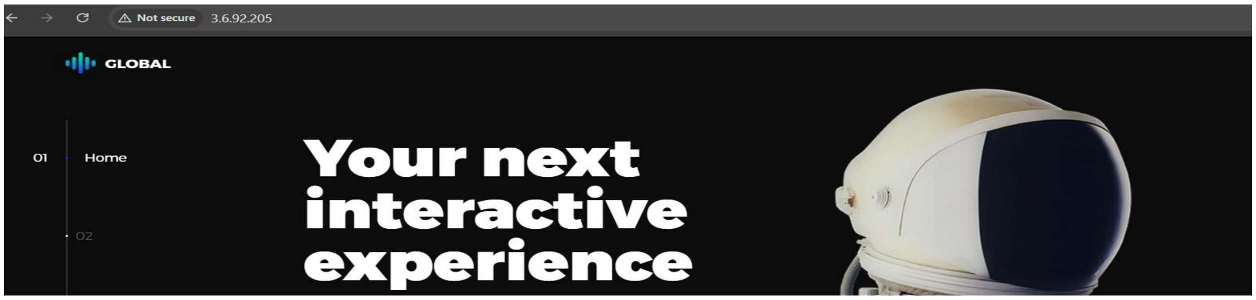
Step13 : Go in Docker-HOST server and check the Container and Image are present or not and webpage is opening by public ip or not :

docker container ls
docker image ls



```
[root@docker-host ~]# docker container ls
CONTAINER ID   IMAGE                               COMMAND                  CREATED          STATUS          PORTS                NAMES
784b2ce60a64   vikash1269/docker-project:latest   "/usr/sbin/httpd -D …"   20 minutes ago   Up 20 minutes   0.0.0.0:80->80/tcp   vkcontainer
[root@docker-host ~]# docker image ls
REPOSITORY                  TAG       IMAGE ID       CREATED             SIZE
vikash1269/docker-project   latest    b77bb2dfc53b   About an hour ago   286MB
[root@docker-host ~]#
```

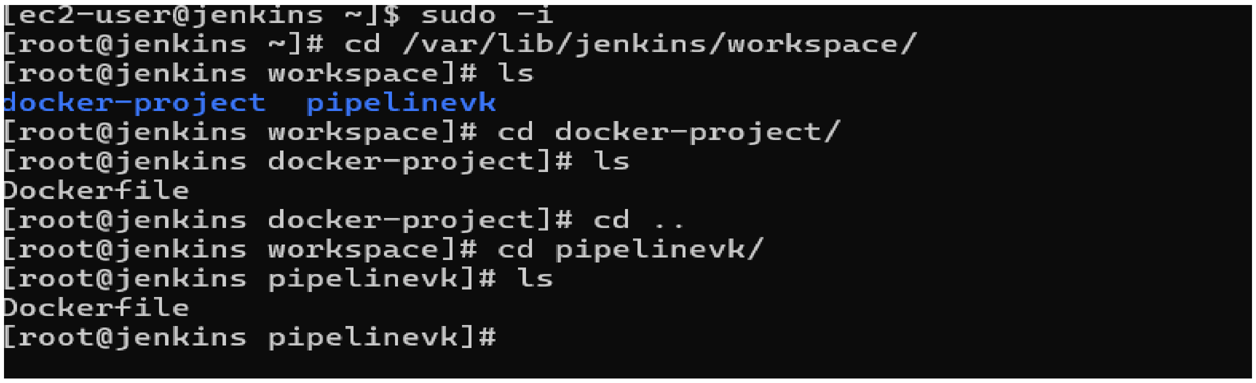
Webpage is opening by public of Docker-HOST server :



****

Step14 : Check Jenkins server , files are present or not :

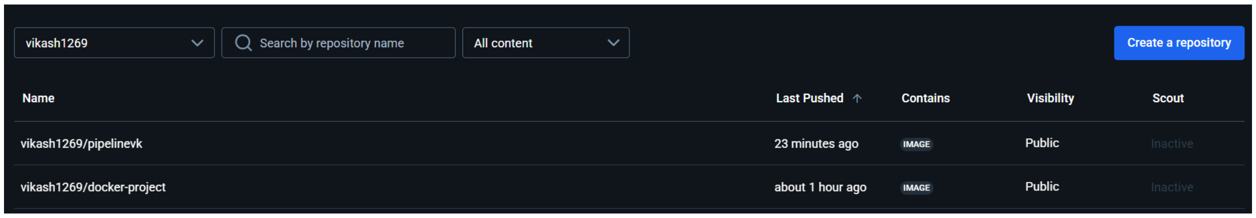
# cd /var/lib/jenkins/workspace/project-name/
(Dockerfile should be present)



```
[ec2-user@jenkins ~]$ sudo -i
[root@jenkins ~]# cd /var/lib/jenkins/workspace/
[root@jenkins workspace]# ls
docker-project  pipelinevk
[root@jenkins workspace]# cd docker-project/
[root@jenkins docker-project]# ls
Dockerfile
[root@jenkins docker-project]# cd ..
[root@jenkins workspace]# cd pipelinevk/
[root@jenkins pipelinevk]# ls
Dockerfile
[root@jenkins pipelinevk]#
```

PROCESS : NOW if we do some changes in Dockerfile in github , a pipeline line is automatically triggered , which we can see on jenkins page and changes will be reflected on the Docker-HOST webpage and the new version of image will automatically pushed on DockerHUB .
****

Step15 : Check Images are transferred to DockerHUB or not :


vikash1269 | Search by repository name | All content | Create a repository

Name	Last Pushed	Contains	Visibility	Scout
vikash1269/pipelinevk	23 minutes ago	IMAGE	Public	Inactive
vikash1269/docker-project	about 1 hour ago	IMAGE	Public	Inactive

****

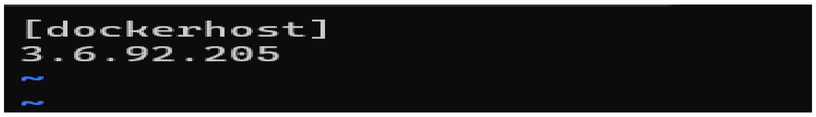
PROJECT COMPLETE

****

TO AVOID ERRORS :

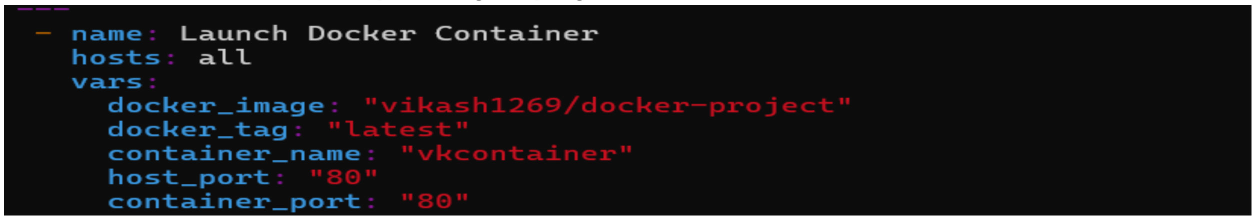
1. **Use public ip**

2. **Do not use '-' in inventory file while assigning group [docker-host]**



```
[dockerhost]
3.6.92.205
~
~
```

3. **Use name of Dockerhub repository in playbook :**



```
---
 - name: Launch Docker Container
   hosts: all
   vars:
     docker_image: "vikash1269/docker-project"
     docker_tag: "latest"
     container_name: "vkcontainer"
     host_port: "80"
     container_port: "80"
```

4. **Use sudo -i , not sudo su .**

****

****

****