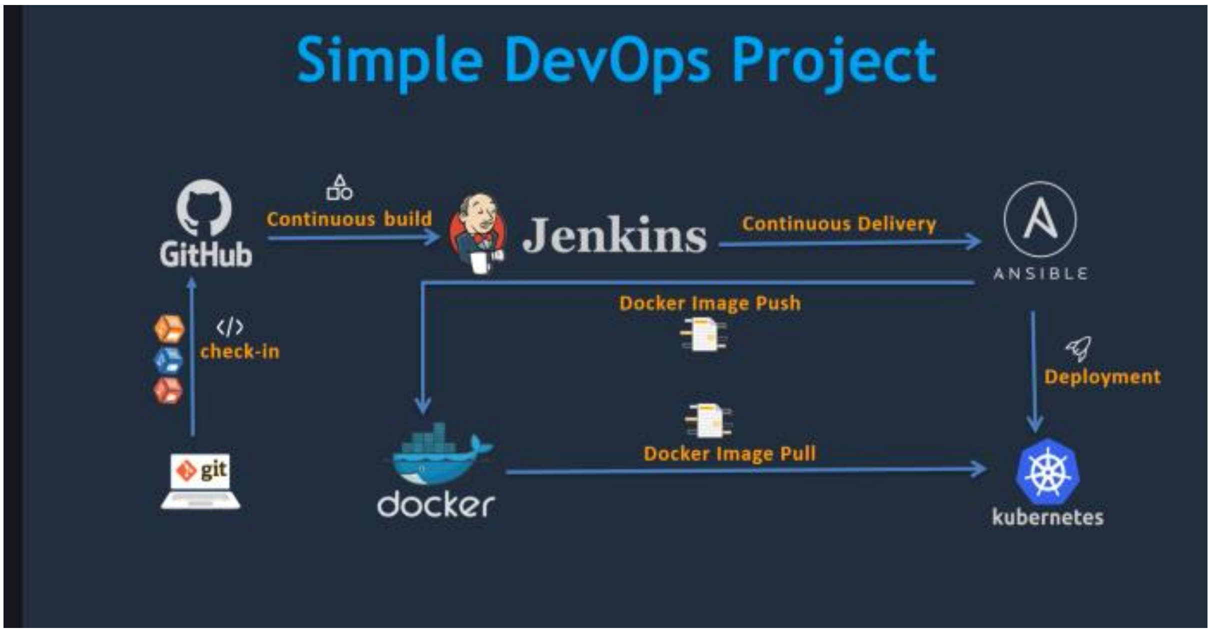
NAME: GAURAV PRAKASH SINGH

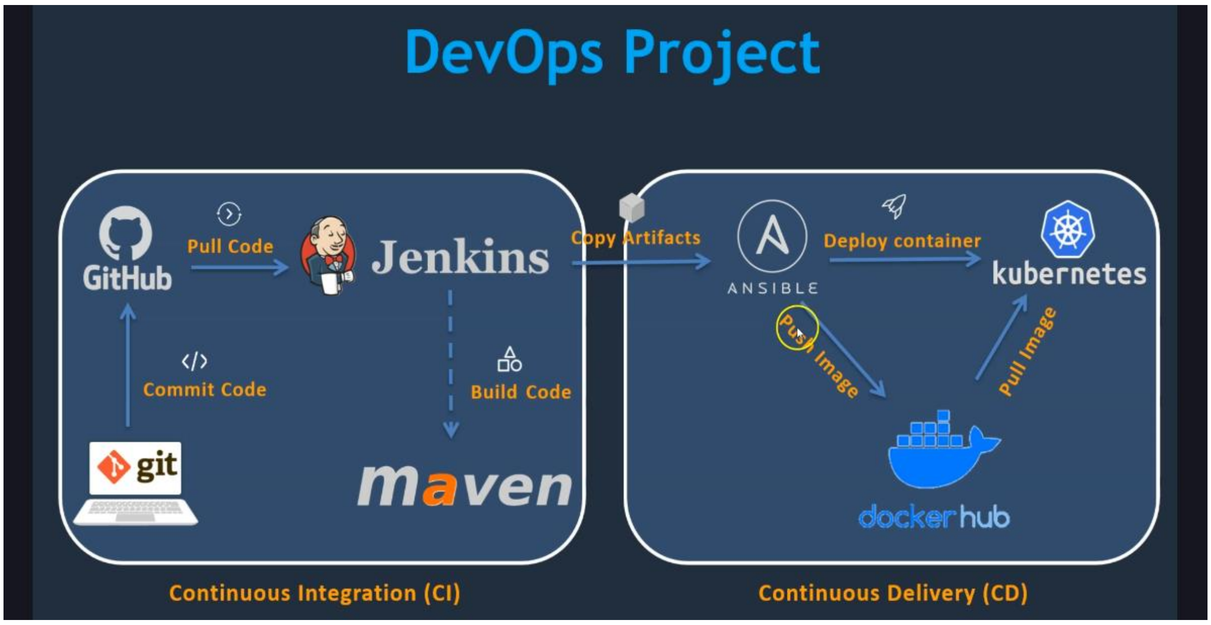
TOPIC:

SIMPLE DEVOPS project with:

- ➔ JENKINS
- ➔ DOCKER
- ➔ KUBERNETES
- ➔ ANSIBEL

General Design





Technologies Used

Programming & Build Tools

- **Java**: Core application development (likely a Maven-based project).
- **Maven**: Project building and dependency management.

Version Control & Collaboration

- **Git & GitHub**: Source code management and version control.
- **Webhooks**: Trigger Jenkins jobs automatically on GitHub commits.

CI/CD & Automation

- **Jenkins**: Automates the CI/CD pipeline (build, test, deploy).
- **Ansible**: Automates server configurations and deployments.

Containerization & Orchestration

- **Docker**: Containerizes the application for portability and ease of deployment.
- **Kubernetes**: Manages and orchestrates Docker containers.
- **kubectl**: CLI tool for interacting with the Kubernetes cluster.

Cloud Infrastructure

- **AWS (Amazon Web Services)**: Deployment platform.
    - **EC2 Instances**: Hosts for Jenkins, Ansible, and Kubernetes nodes.

Configuration Management

- **YAML**: Used for Ansible playbooks and Kubernetes manifests.

INSTALLING JENKINS AND SETTING UP

Steps:

Become root use r: sudo su –

Install java and Jenkins

sudo yum update -y

sudo yum install java-17-amazon-corretto -y

sudo amazon-linux-extras install epel

java -version

sudo yum install -y wget
sudo wget -O /etc/yum.repos.d/jenkins.repo https://pkg.jenkins.io/redhat-stable/jenkins.repo
sudo rpm --import https://pkg.jenkins.io/redhat-stable/jenkins.io.key
sudo rpm --import https://pkg.jenkins.io/redhat-stable/jenkins.io-2023.key
sudo yum install -y jenkins

sudo systemctl enable jenkins
sudo systemctl start jenkins

Sudo amazon-linux-extras install epel

Integrate Git with Jenkins

Pulling from Github in Jenkins:

Jenkins does action under workspace

We can go there and check

Integrate Maven With Jenknis

Download Maven

Download Maven on Linux under Direcort /opt , and using link

wget https://dlcdn.apache.org/maven/maven-3/3.9.10/binaries/apache-maven-3.9.10-bin.tar.gz

and make a maven dir move it in and extract t using

- tar -xvzf apache-maven-3.9.10-bin.tar

Now set it up in the Env Variables

Go to :

Cd ~

Ll -a

In .bash_profile edit it, add Java path and M2_Home

/opt/maven/apache-maven-3.9.10

Find path using : find / -name java-17-amazon-corretto.x86_64

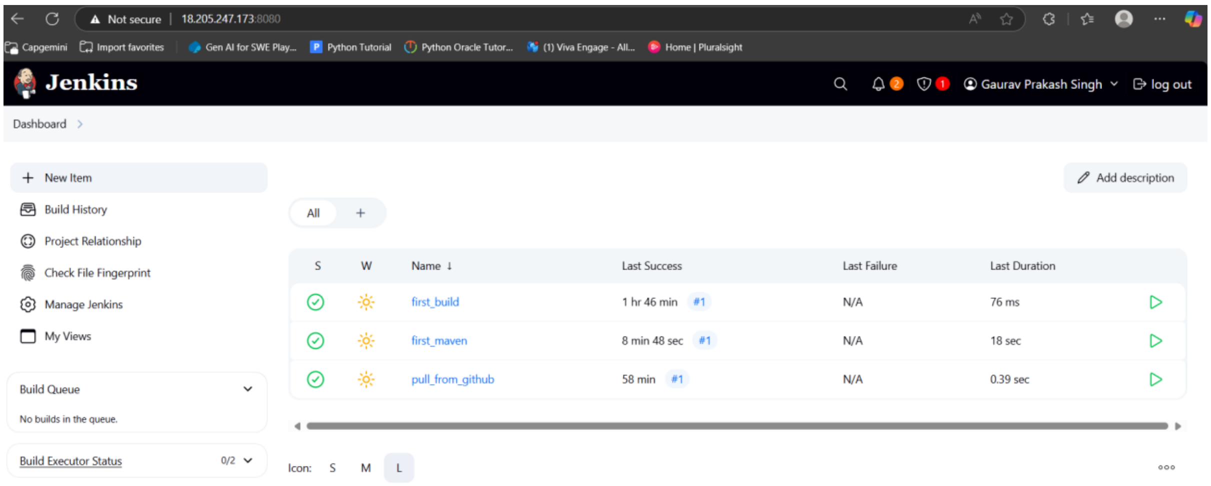
/usr/lib/jvm/java-17-amazon-corretto.x86_64

Set M2 _Home , M2 and JAVA_HOME in vi .bash_profile

M2_HOME: /opt/maven/apache-maven-3.9.10 :

M2: /opt/maven/apache-maven-3.9.10/bin

Now configure Jenkins , after installing plugin , set the global tools for JDK and Maven


Jenkins

Dashboard

Gaurav Prakash Singh | log out

- New Item
- Build History
- Project Relationship
- Check File Fingerprint
- Manage Jenkins
- My Views

Build Queue

No builds in the queue.

Build Executor Status 0/2

Add description

All +

S	W	Name ↓	Last Success	Last Failure	Last Duration
✓	☀	first_build	1 hr 46 min #1	N/A	76 ms
✓	☀	first_maven	8 min 48 sec #1	N/A	18 sec
✓	☀	pull_from_github	58 min #1	N/A	0.39 sec

Icon: S M L

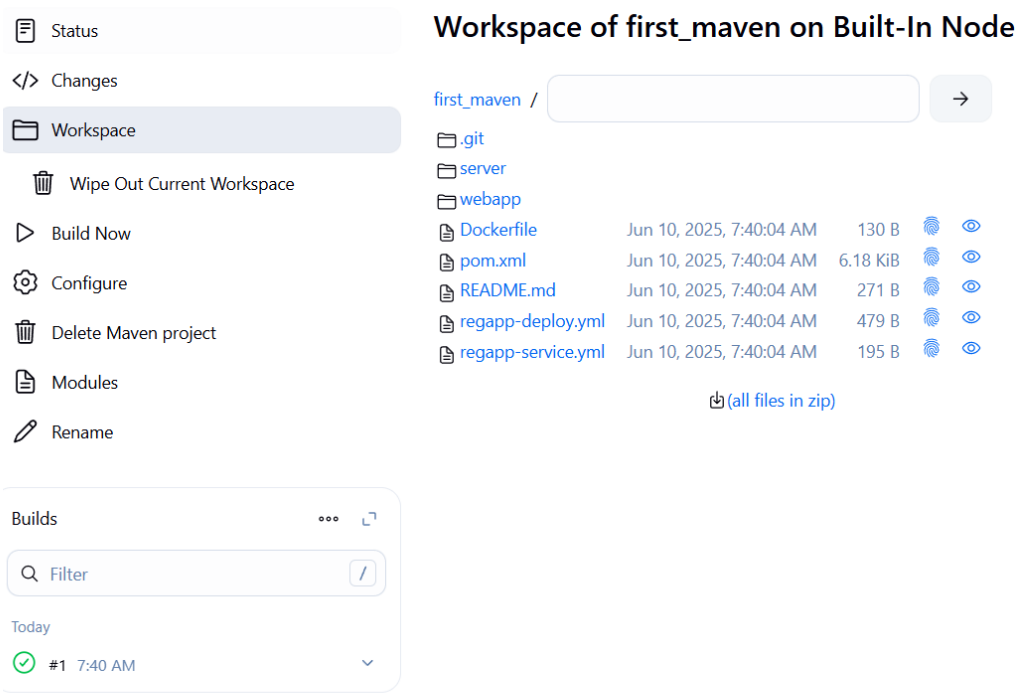
Now Build Job

By the end of this we have made a Maven Build in Jenkisn the flow is

Our artifact will be inside webapp/target

GIT-> GITHUB->JENKINS

MAVEN


- Status
- Changes
- Workspace
  - Wipe Out Current Workspace
- Build Now
- Configure
- Delete Maven project
- Modules
- Rename

Builds

Filter

Today

✓ #1 7:40 AM

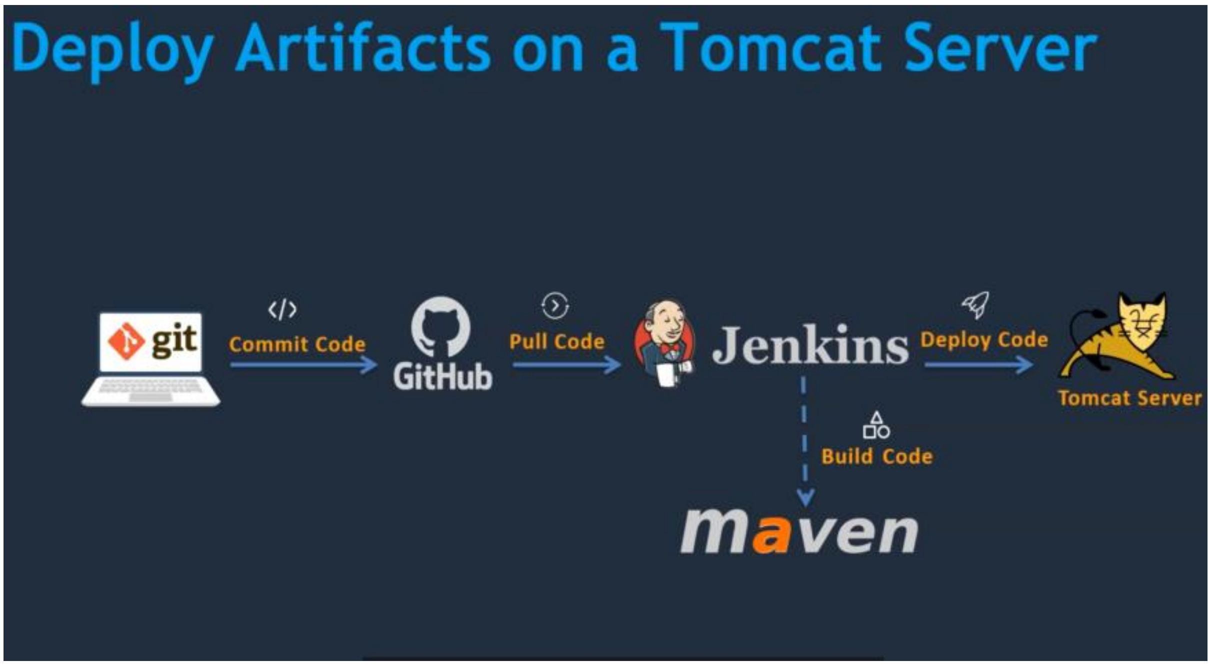
Workspace of first_maven on Built-In Node

first_maven /

.git		
server		
webapp		
Dockerfile	Jun 10, 2025, 7:40:04 AM	130 B
pom.xml	Jun 10, 2025, 7:40:04 AM	6.18 KiB
README.md	Jun 10, 2025, 7:40:04 AM	271 B
regapp-deploy.yml	Jun 10, 2025, 7:40:04 AM	479 B
regapp-service.yml	Jun 10, 2025, 7:40:04 AM	195 B

(all files in zip)

SECTION 2: Deploy Artifact on TomCAt Server



Now Setup a Linux EC2 instace for Tomcat Server , install java

Install java In new EC2: sudo yum install -y java-17-amazon-corretto

Switch to /opt and configure tomcat same way as Maven

Download the using wget and link , extratct using tar -xvzf

After that go in bin dir and do ./startup.sh this will start the tomcat server

Access using ec2 : ip:8080

Now To access the manager , to need to access from outside we need to update the context.xml file.

So go back in the console

Find the context.xml file. In the tomcat dir using find like , find /-name context.xml f

We need to update :

1. /opt/tomcat/apache-tomcat-10.1.42/webapps/host-manager/META-INF/context.xml

In this file comment out the below one:

<!-- <Valve className="org.apache.catalina.valves.RemoteAddrValve"

allow="127\.\d+\.\d+\.\d+|::1|0:0:0:0:0:0:0:1" /> -->

2. /opt/tomcat/apache-tomcat-10.1.42/webapps/manager/META-INF/context.xml

Same here too

These 2 files

Now restart the tomcat server by going in bin

Now for credential go to users.xml and update it , this is inside conf dir

Add users in it

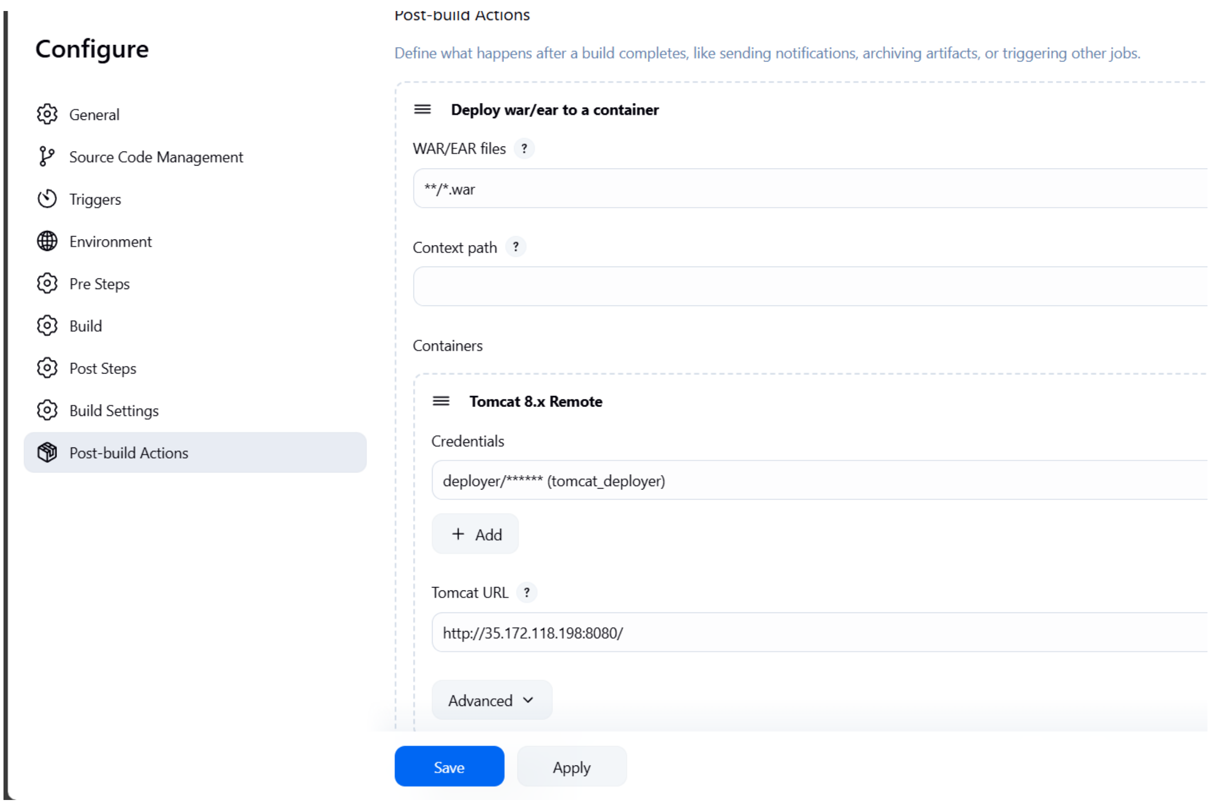
```
<role rolename="manager-gui"/>
 <role rolename="manager-script"/>
 <role rolename="manager-jmx"/>
 <role rolename="manager-status"/>
 <user username="admin" password="admin" roles="manager-gui, manager-script, manager-jmx,
manager-status"/>
 <user username="deployer" password="deployer" roles="manager-script"/>
 <user username="tomcat" password="s3cret" roles="manager-gui"/>
```

Sigin using credentials

INTEGRATE TOMCAT with Jenkins:

To do this we need a plug in called deploy to container

Configure the Job :



Post-build Actions

Define what happens after a build completes, like sending notifications, archiving artifacts, or triggering other jobs.

Deploy war/ear to a container

WAR/EAR files: **/*.war

Context path:

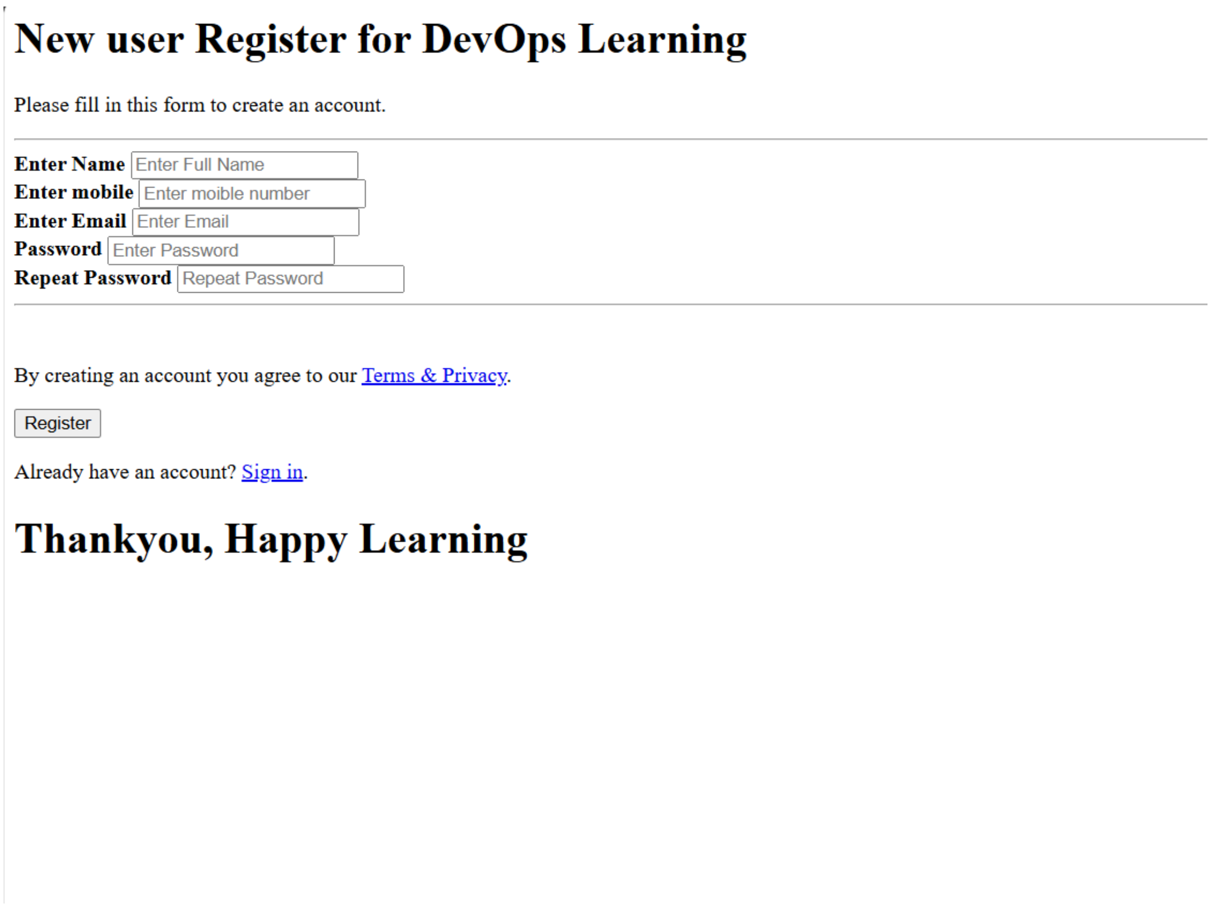
Containers

Tomcat 8.x Remote

Credentials: deployer/****** (tomcat_deployer)

Tomcat URL: http://35.172.118.198:8080/

We can see in the Tomcat manager , a new /webapp is created click on it and we will see,.



New user Register for DevOps Learning

Please fill in this form to create an account.

Enter Name	Enter Full Name
Enter mobile	Enter moible number
Enter Email	Enter Email
Password	Enter Password
Repeat Password	Repeat Password

By creating an account you agree to our Terms & Privacy.

Already have an account? Sign in.

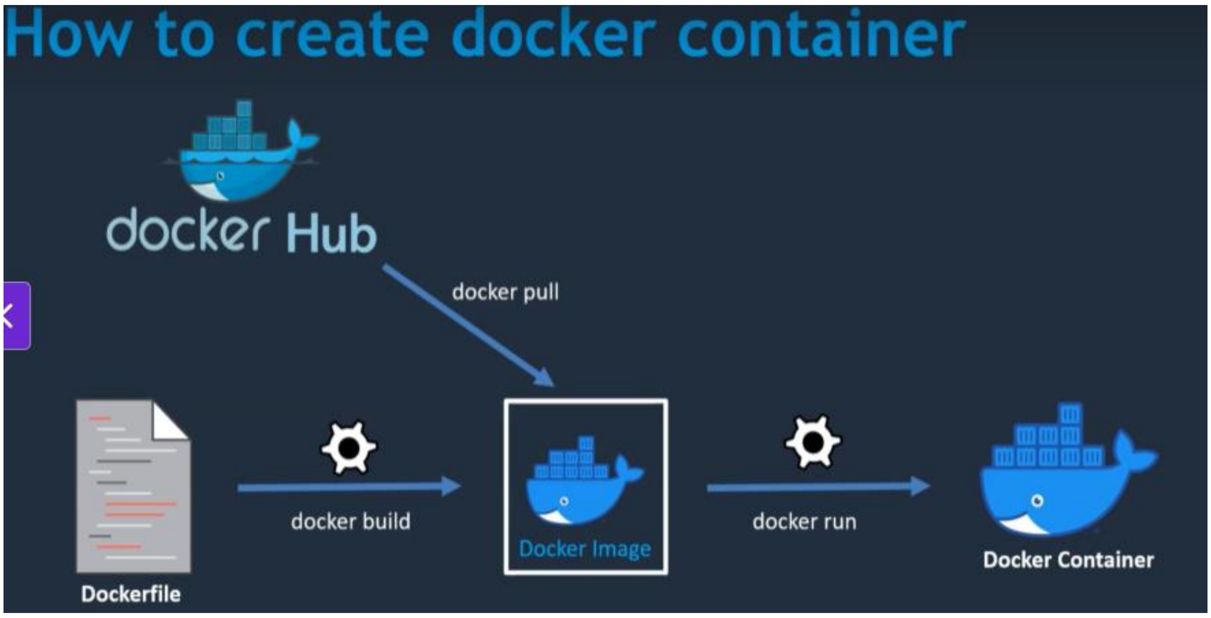
Thankyou, Happy Learning

Apply Build schedules and Build Triggers as required

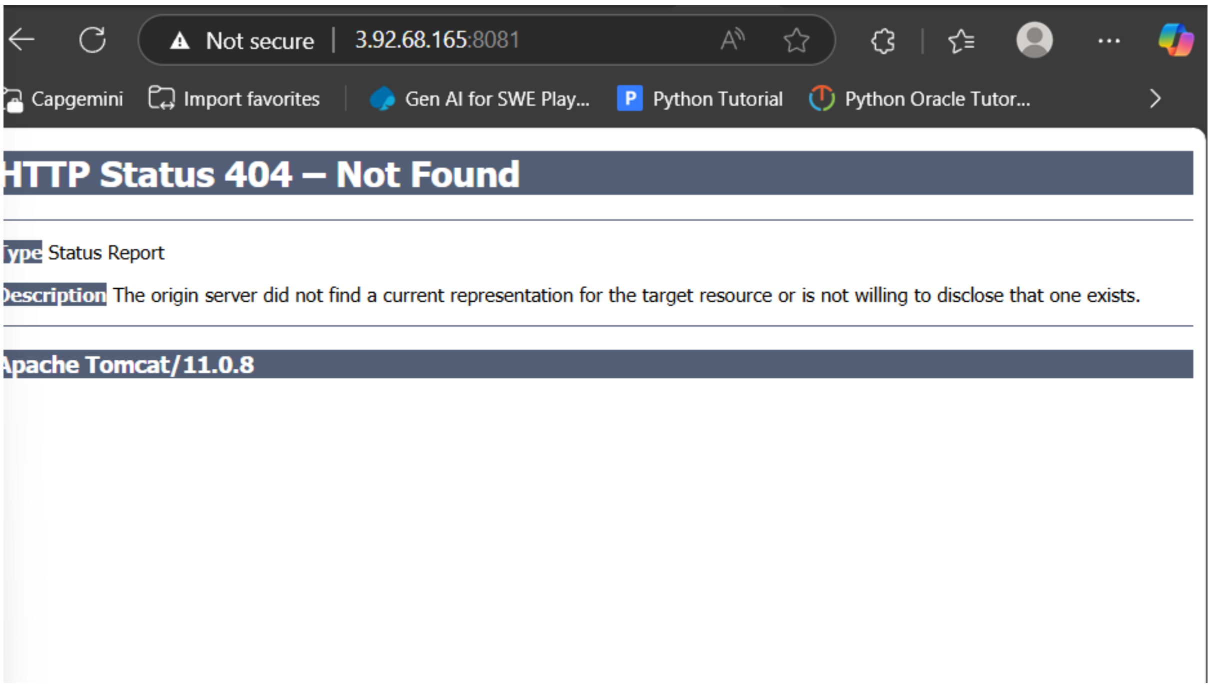
Deploy on the Docker Cntainer

Set up Docker Host on new EC2

::Now create a Tomcat container



Create Container of Tomcat using image from dockerhub



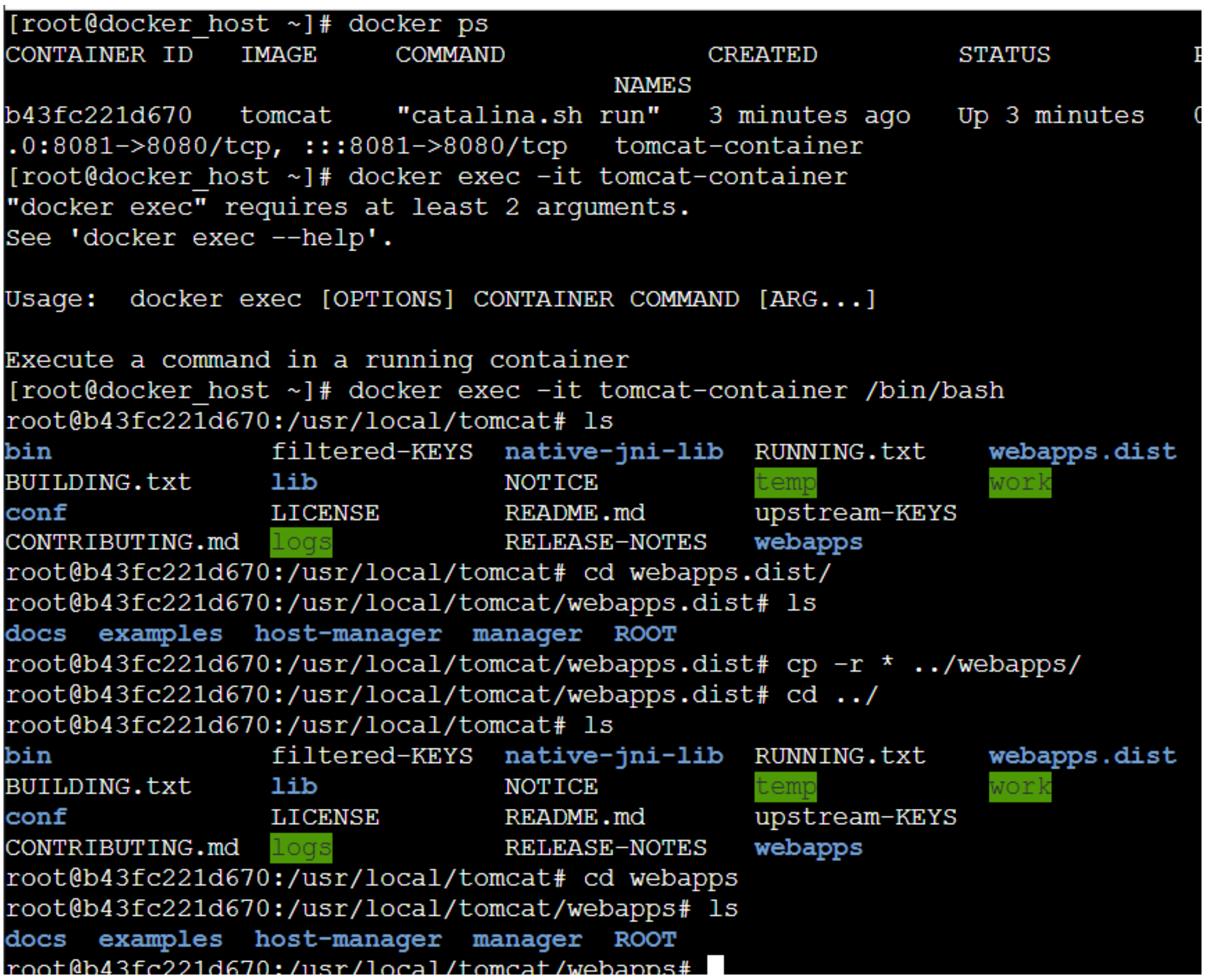
HTTP Status 404 – Not Found

Type Status Report

Description The origin server did not find a current representation for the target resource or is not willing to disclose that one exists.

Apache Tomcat/11.0.8

To fix this issue go inside docker container : docker exec -it tomcat-container /bin/bash

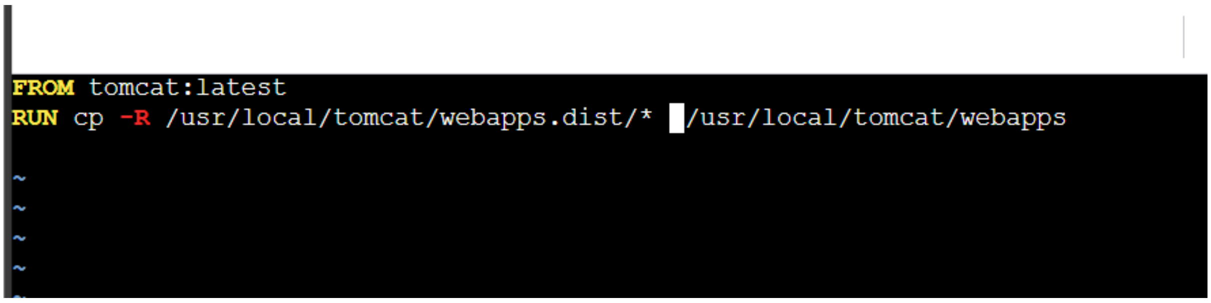


```
[root@docker_host ~]# docker ps
CONTAINER ID   IMAGE     COMMAND             CREATED         STATUS          P
                                       NAMES
b43fc221d670   tomcat    "catalina.sh run"   3 minutes ago   Up 3 minutes    0
.0:8081->8080/tcp, :::8081->8080/tcp   tomcat-container
[root@docker_host ~]# docker exec -it tomcat-container
"docker exec" requires at least 2 arguments.
See 'docker exec --help'.

Usage:  docker exec [OPTIONS] CONTAINER COMMAND [ARG...]

Execute a command in a running container
[root@docker_host ~]# docker exec -it tomcat-container /bin/bash
root@b43fc221d670:/usr/local/tomcat# ls
bin              filtered-KEYS  native-jni-lib  RUNNING.txt    webapps.dist
BUILDING.txt     lib            NOTICE          temp           work
conf             LICENSE        README.md       upstream-KEYS
CONTRIBUTING.md  logs           RELEASE-NOTES   webapps
root@b43fc221d670:/usr/local/tomcat# cd webapps.dist/
root@b43fc221d670:/usr/local/tomcat/webapps.dist# ls
docs  examples  host-manager  manager  ROOT
root@b43fc221d670:/usr/local/tomcat/webapps.dist# cp -r * ../webapps/
root@b43fc221d670:/usr/local/tomcat/webapps.dist# cd ../
root@b43fc221d670:/usr/local/tomcat# ls
bin              filtered-KEYS  native-jni-lib  RUNNING.txt    webapps.dist
BUILDING.txt     lib            NOTICE          temp           work
conf             LICENSE        README.md       upstream-KEYS
CONTRIBUTING.md  logs           RELEASE-NOTES   webapps
root@b43fc221d670:/usr/local/tomcat# cd webapps
root@b43fc221d670:/usr/local/tomcat/webapps# ls
docs  examples  host-manager  manager  ROOT
root@b43fc221d670:/usr/local/tomcat/webapps#
```

After this it will work

NOTE: Now everytime we make a new container , the previous issue of 404 will still be there as the webapps in the new container do not have that files , to solev this issue we can use docker file , In which we can specify the customization.

MAKE A CUTOMIZE docker image using dockerfile



```
FROM tomcat:latest
RUN cp -R /usr/local/tomcat/webapps.dist/* /usr/local/tomcat/webapps

~
~
~
~
```

Build image

docker build -t demotomcat

- ➔ Run the image

docker run -d --name demotomcat-container -p 8085:8080 demotomcat

- ➔ Now that eror will not occur

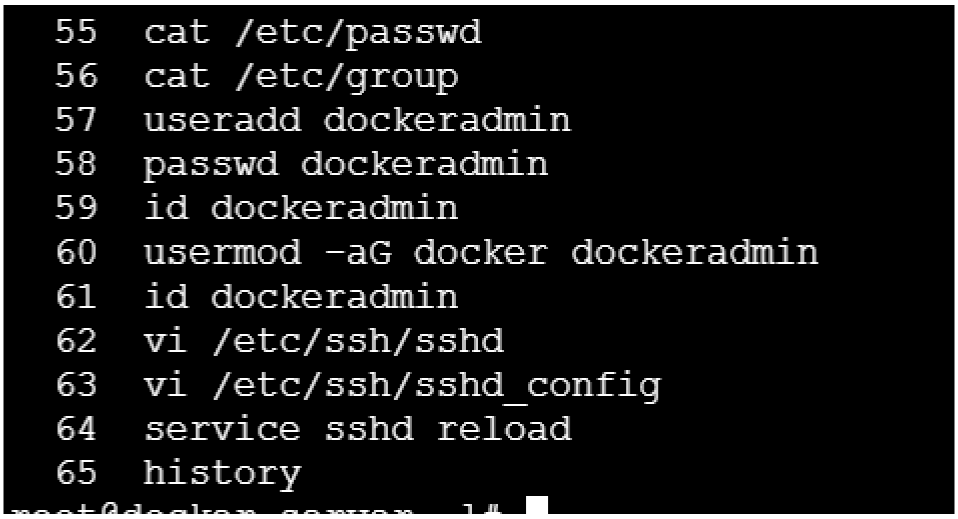
INTEGRATE DOCKER with JENKINS

- ➔ Create dockeradmin user
- ➔ Install Publish OVER SSH plugin
- ➔ Add Dockerhost to Jenkins configure system

To get user list :   cat /etc/passwd

Make user :

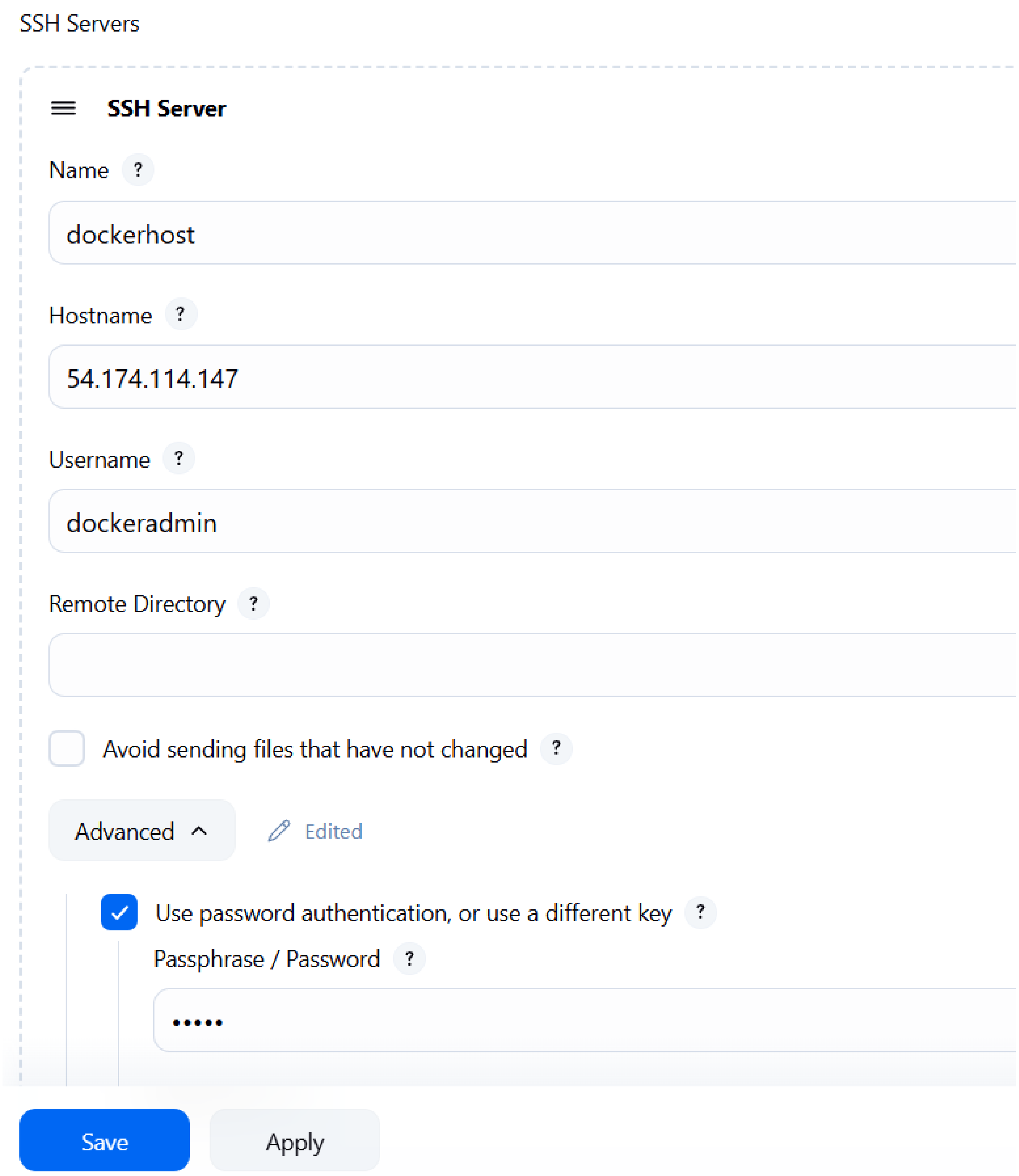
Useradd dockeradmin, add user to the docker group and edit sshd_config so password based authentication is possible



```
  55  cat /etc/passwd
  56  cat /etc/group
  57  useradd dockeradmin
  58  passwd dockeradmin
  59  id dockeradmin
  60  usermod -aG docker dockeradmin
  61  id dockeradmin
  62  vi /etc/ssh/sshd
  63  vi /etc/ssh/sshd_config
  64  service sshd reload
  65  history
```

Now set up PUBLISH over ssh in Jenkins plugin

Configure it in Jenkins



SSH Servers

SSH Server

Name: dockerhost

Hostname: 54.174.114.147

Username: dockeradmin

Remote Directory:

☐ Avoid sending files that have not changed

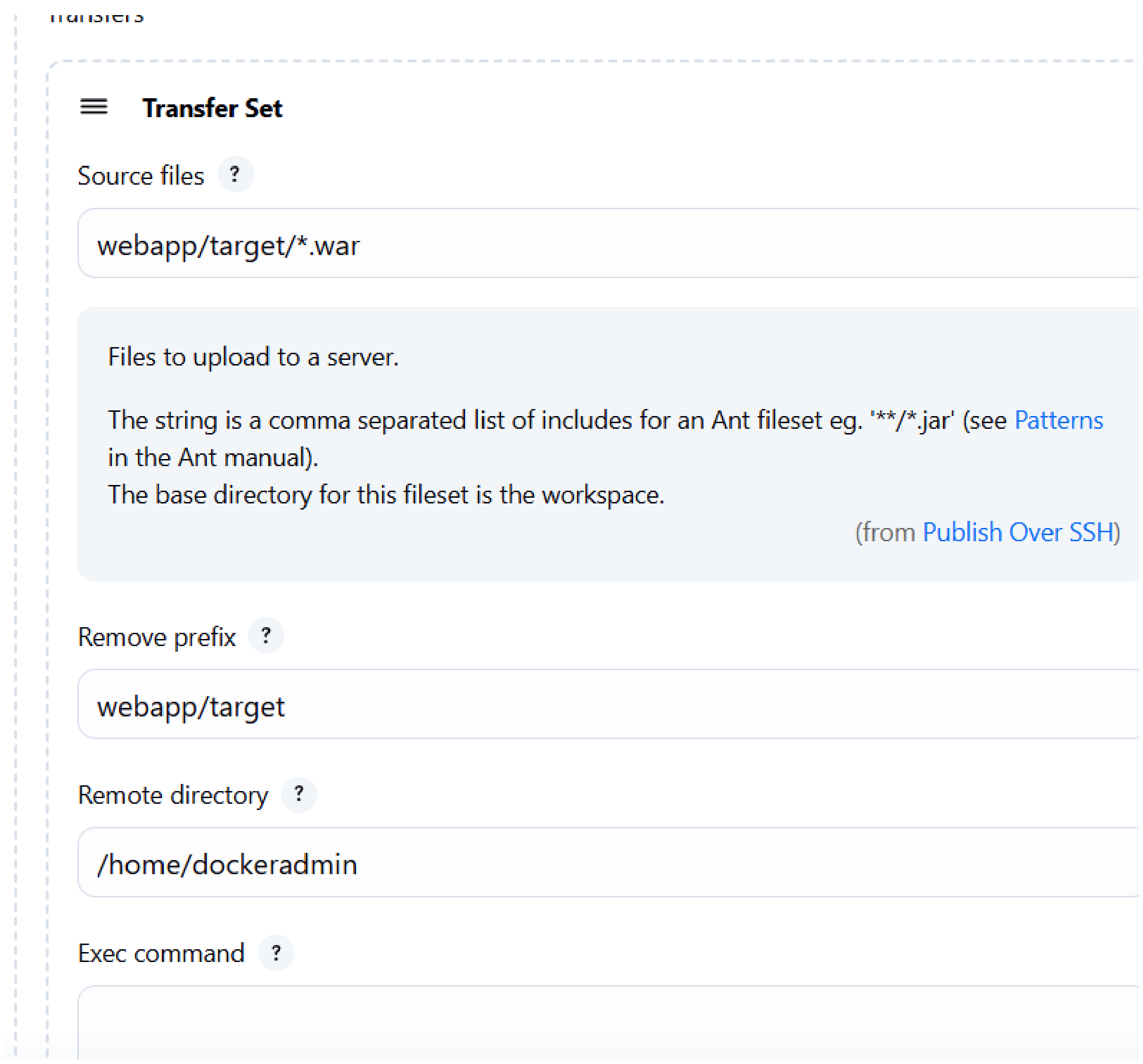
Advanced — Edited

☑ Use password authentication, or use a different key

Passphrase / Password: •••••

NOW WE CAN BUILD AND DEPLOY THE ARTIFACTS ON DOCKER CONTAINER

Copy the build config ,and in post build action select send build artifacts over SSH


Transfers

Transfer Set

Source files ?

webapp/target/*.war

Files to upload to a server.

The string is a comma separated list of includes for an Ant fileset eg. '**/*.jar' (see Patterns in the Ant manual).
The base directory for this fileset is the workspace.

(from Publish Over SSH)

Remove prefix ?

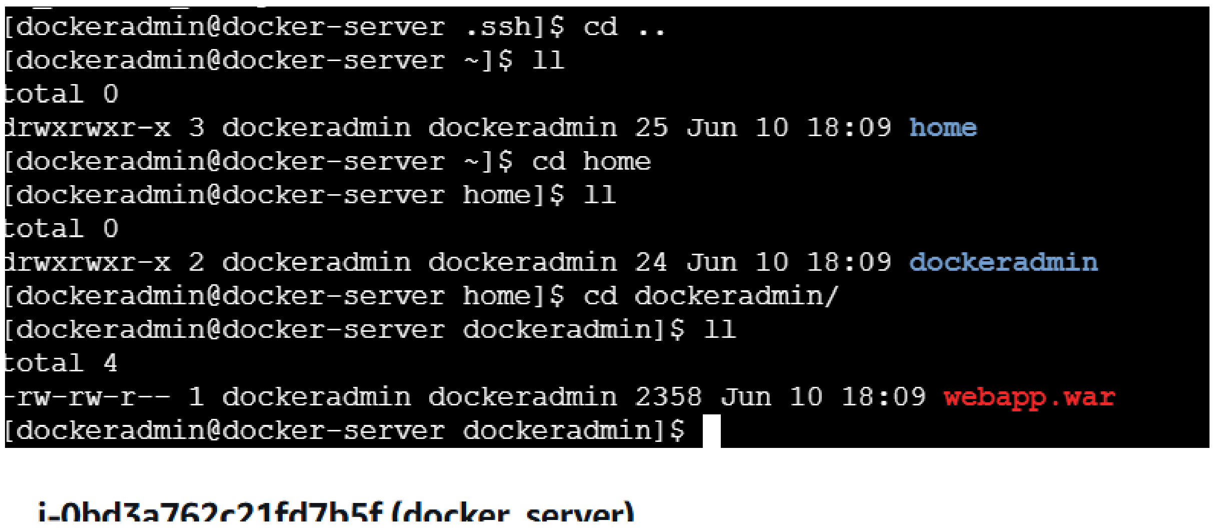
webapp/target

Remote directory ?

/home/dockeradmin

Exec command ?

NOW BUuild it , we can see the articat in webapp.war

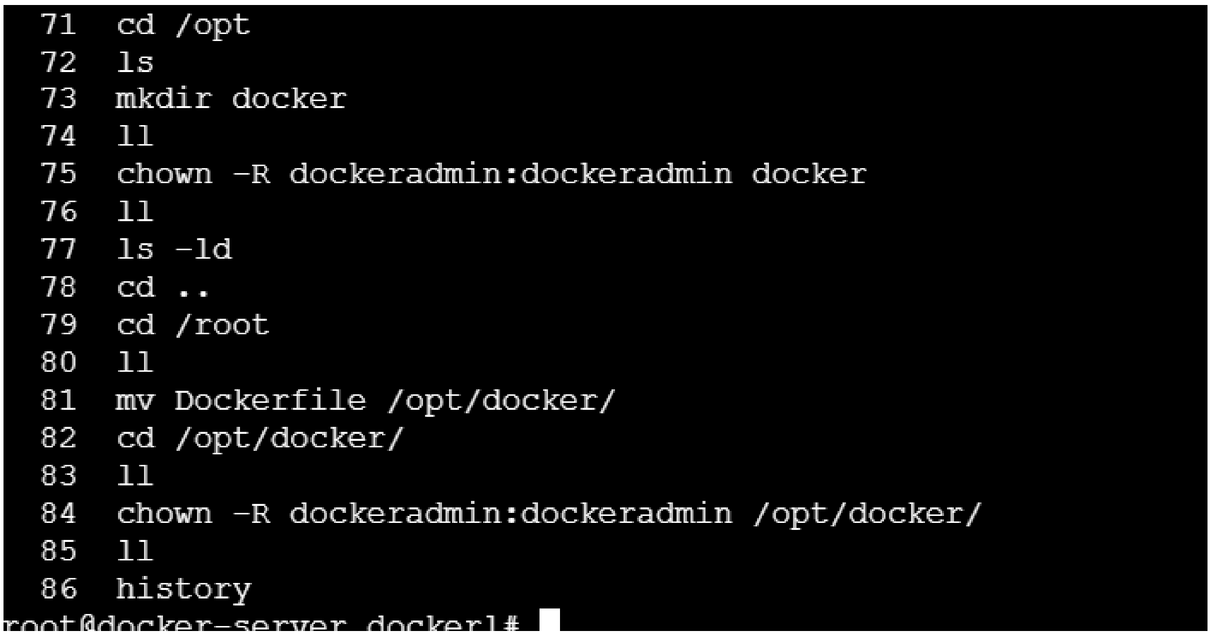

```
[dockeradmin@docker-server .ssh]$ cd ..
[dockeradmin@docker-server ~]$ ll
total 0
drwxrwxr-x 3 dockeradmin dockeradmin 25 Jun 10 18:09 home
[dockeradmin@docker-server ~]$ cd home
[dockeradmin@docker-server home]$ ll
total 0
drwxrwxr-x 2 dockeradmin dockeradmin 24 Jun 10 18:09 dockeradmin
[dockeradmin@docker-server home]$ cd dockeradmin/
[dockeradmin@docker-server dockeradmin]$ ll
total 4
-rw-rw-r-- 1 dockeradmin dockeradmin 2358 Jun 10 18:09 webapp.war
[dockeradmin@docker-server dockeradmin]$
```

i-0bd3a762c21fd7b5f (docker_server)

➔ Now we want to copy the webapp.war in the docker container

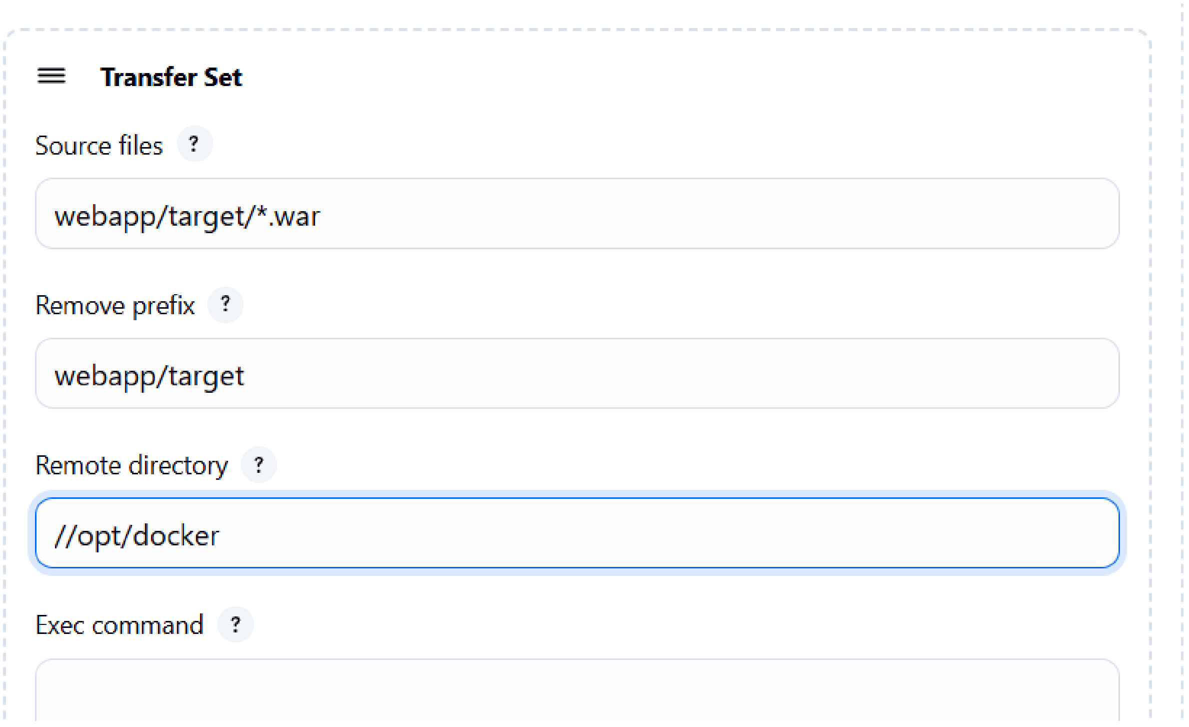
We can do one thing make a separate directory "docker" and in that we can have out artifact i.e, "webapps.war" and our Dockerfile from which we can make image , also be careful to add to give the permission on dir and Dockerfile to the dockeradmin as currently it is owned by the root

Do following:



```
   71  cd /opt
   72  ls
   73  mkdir docker
   74  ll
   75  chown -R dockeradmin:dockeradmin docker
   76  ll
   77  ls -ld
   78  cd ..
   79  cd /root
   80  ll
   81  mv Dockerfile /opt/docker/
   82  cd /opt/docker/
   83  ll
   84  chown -R dockeradmin:dockeradmin /opt/docker/
   85  ll
   86  history
root@docker-server docker]#
```

Now in Jenkins configure the remote dir where the artifice will be copied



Transfer Set

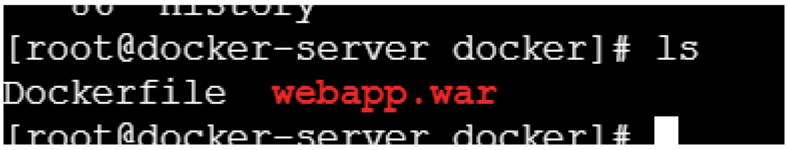
Source files
webapp/target/*.war

Remove prefix
webapp/target

Remote directory
//opt/docker

Exec command

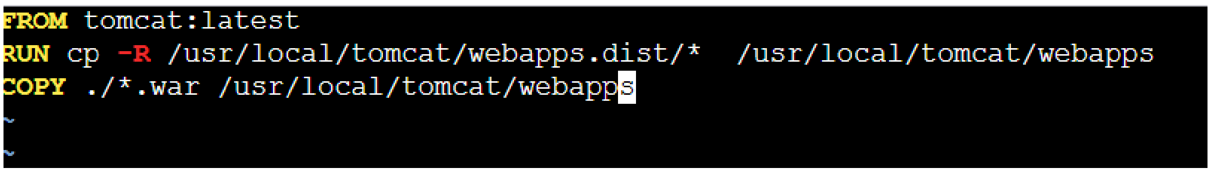
And build it



```
[root@docker-server docker]# ls
Dockerfile  webapp.war
[root@docker-server docker]#
```

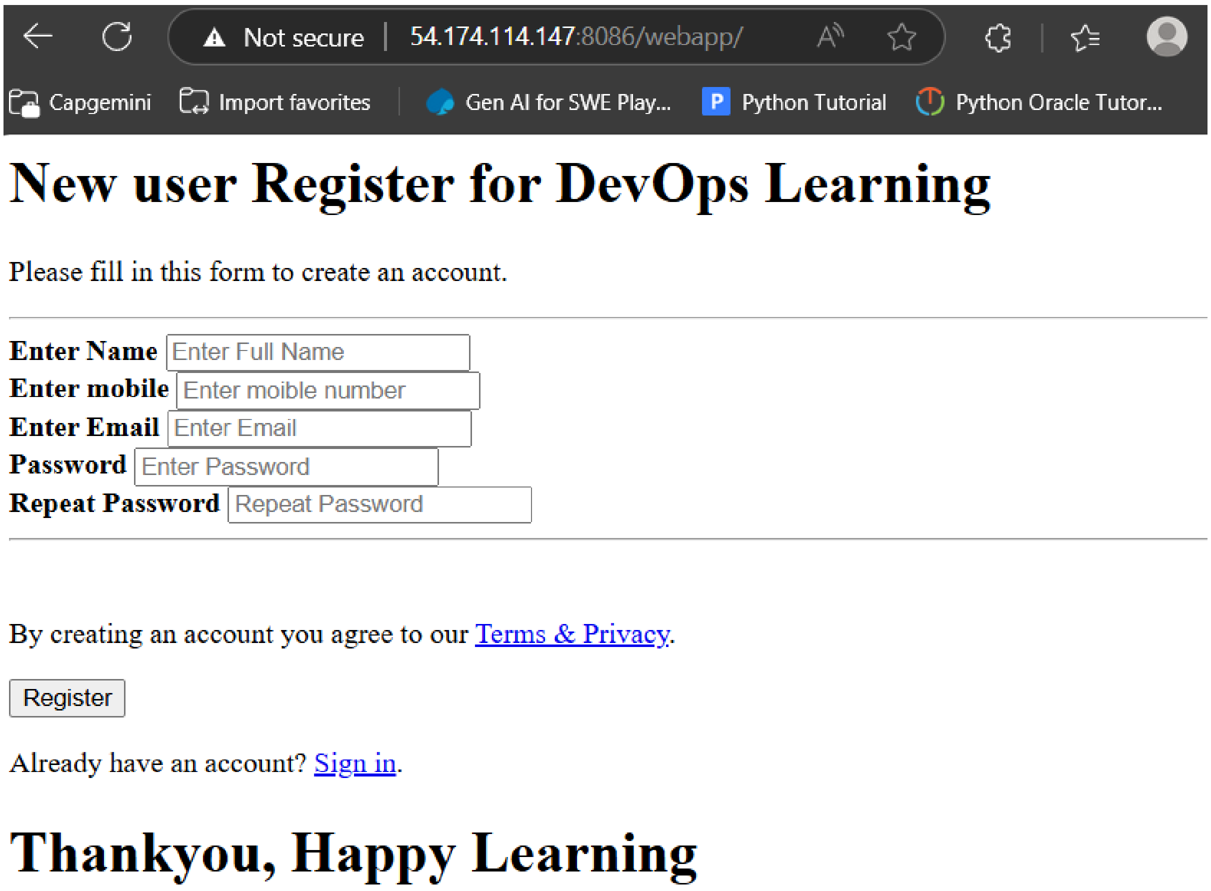
We can see that at one place our artifact and Dockerfile is available

Now we can make a image and container make some changes in docker file as



```
FROM tomcat:latest
RUN cp -R /usr/local/tomcat/webapps.dist/*  /usr/local/tomcat/webapps
COPY ./*.war /usr/local/tomcat/webapps
~
~
```

➔ After this make image and container , and we can deploy the atrifact using docker container



New user Register for DevOps Learning

Please fill in this form to create an account.

Enter Name	Enter Full Name
Enter mobile	Enter moible number
Enter Email	Enter Email
Password	Enter Password
Repeat Password	Repeat Password

By creating an account you agree to our Terms & Privacy.

Register

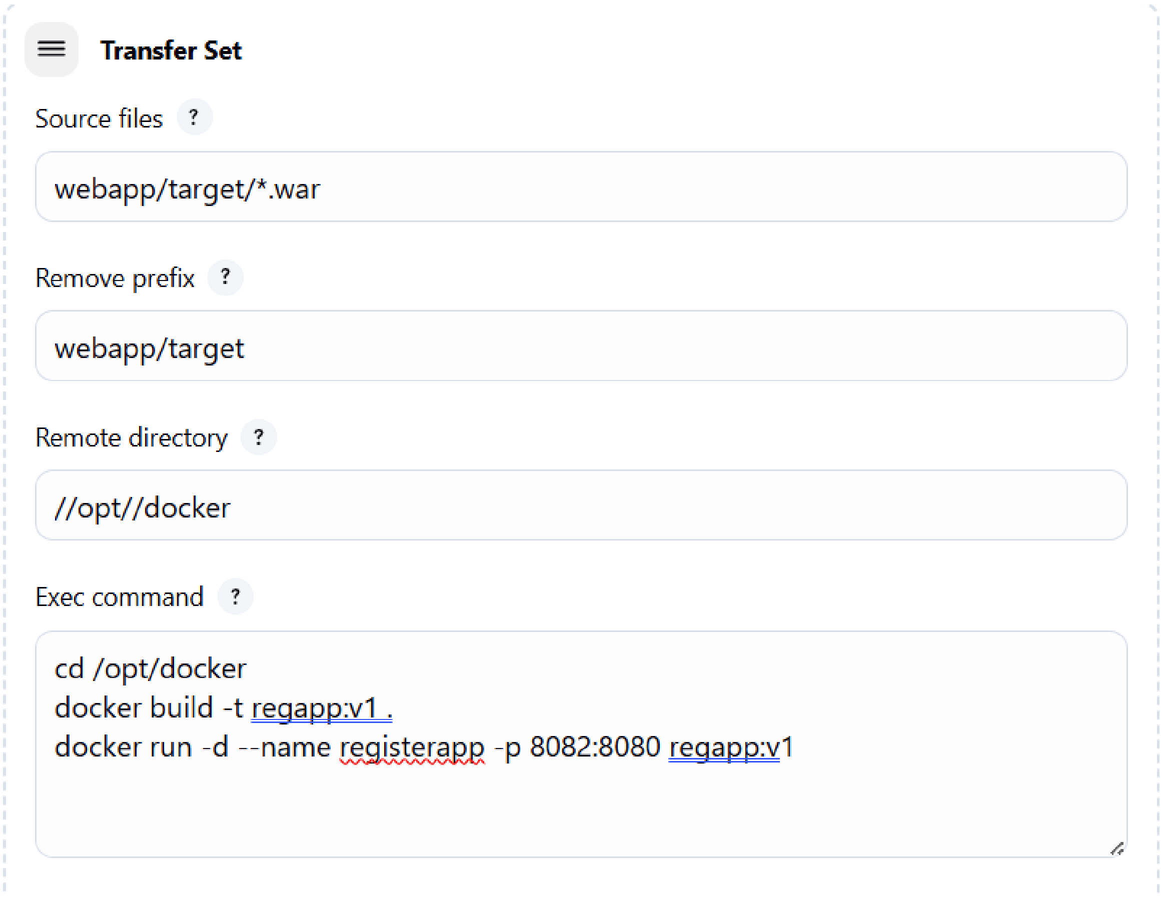
Already have an account? Sign in.

Thankyou, Happy Learning

➔ Currently we are doing the making of image and container manually, we need to specify to the Jenkins to make it fully automated

AUTOMATING BUILD AND DEPLOY ON DOCKER CONTANER

➔ Update the build config of the deploy in the Transfer set add exec command.



Transfer Set

Source files ?

```
webapp/target/*.war
```

Remove prefix ?

```
webapp/target
```

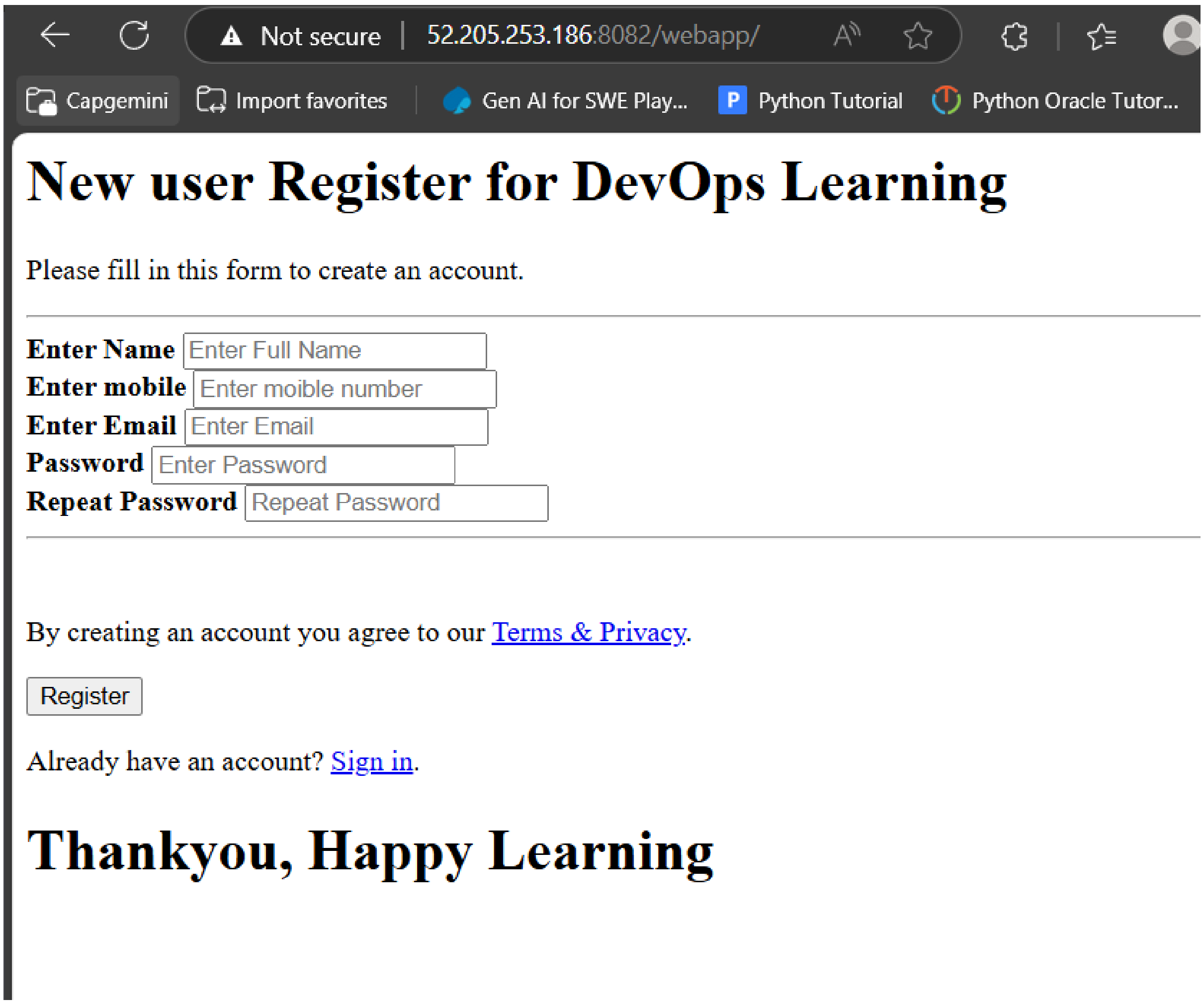
Remote directory ?

```
//opt//docker
```

Exec command ?

```
cd /opt/docker
docker build -t regapp:v1 .
docker run -d --name registerapp -p 8082:8080 regapp:v1
```

For clarity remove all the images and containers
Now build and deploy ,


Not secure | 52.205.253.186:8082/webapp/

Capgemini | Import favorites | Gen AI for SWE Play... | Python Tutorial | Python Oracle Tutor...

New user Register for DevOps Learning

Please fill in this form to create an account.

Enter Name	Enter Full Name
Enter mobile	Enter moible number
Enter Email	Enter Email
Password	Enter Password
Repeat Password	Repeat Password

By creating an account you agree to our Terms & Privacy.

Register

Already have an account? Sign in.

Thankyou, Happy Learning

NOTE: WHEN new changes are made in repo and the Jenkins pulls it from github , the build will be success but the docker build and docker run will be failed , as the container will be created with the same name as previous one , and this will cause erroer as 2 containers cannot have same name.

SOLUTION:

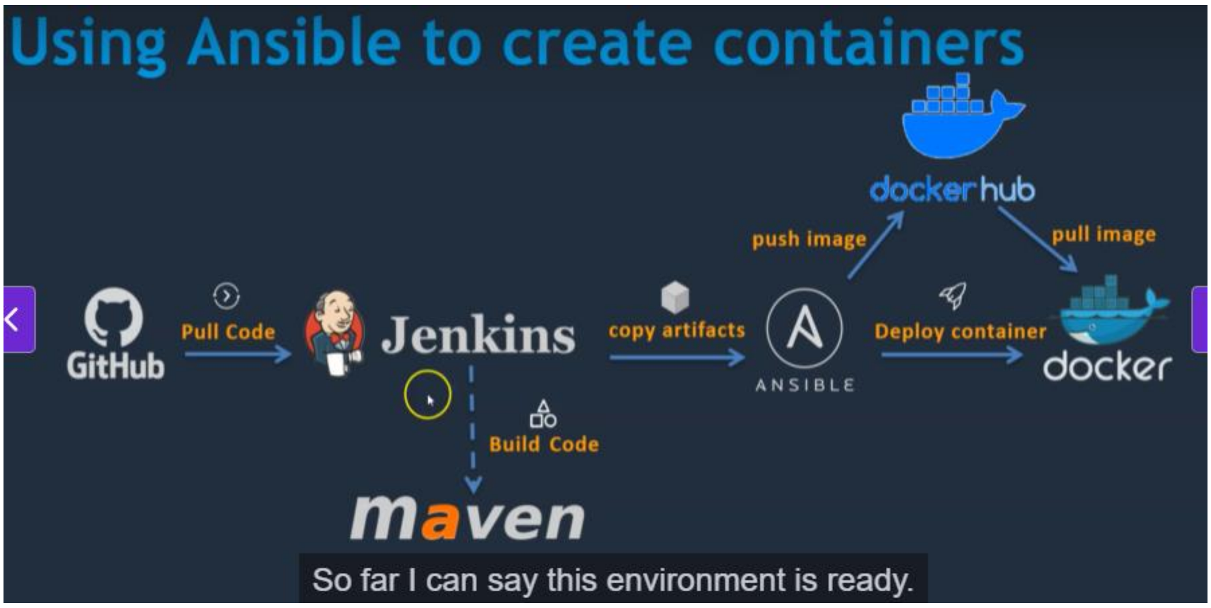
➔ Edit the job config , the basic idea is that before creating a container just delete the container if that container exixst like this :

```
Cd /opt/docker

Docker build -t regapp:v1 .

docker stop registerapp

docker rm registerapp

docker run -d –name registerapp -p 8082:8080 regapp:v1
```

NOTE: This is not the write way , that's why we need build tools like Ansible.

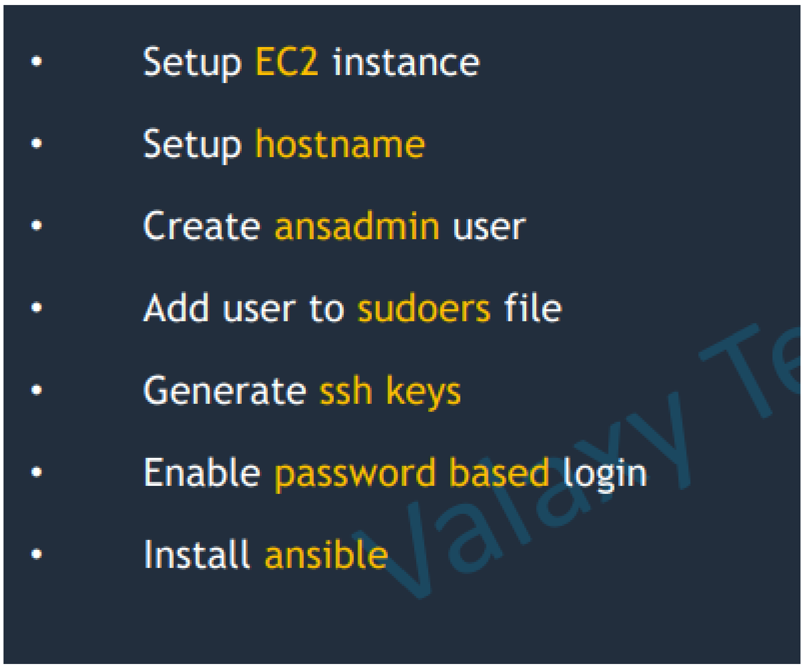
ASIBLE

- ➔ *IT is an Deployment tool*
- ➔ *Ansible task will be to build images from artifacts*



Integrate Ansible with Jenkins

: Prepare Ansible Server:



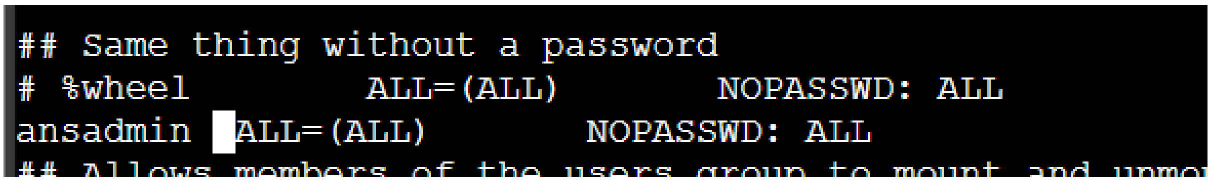
- Setup EC2 instance
- Setup hostname
- Create ansadmin user
- Add user to sudoers file
- Generate ssh keys
- Enable password based login
- Install ansible

Do:

useradd ansiadmin

passwd ansiadmin

visudo : edit the file and add the user



```
## Same thing without a password
# %wheel        ALL=(ALL)       NOPASSWD: ALL
ansadmin  ALL=(ALL)       NOPASSWD: ALL
## Allows members of the users group to mount and unmo
```

edit the /etc/ssh/ssh_config file for password based auth

next generate ssh key in the ansadmin user for our admin user

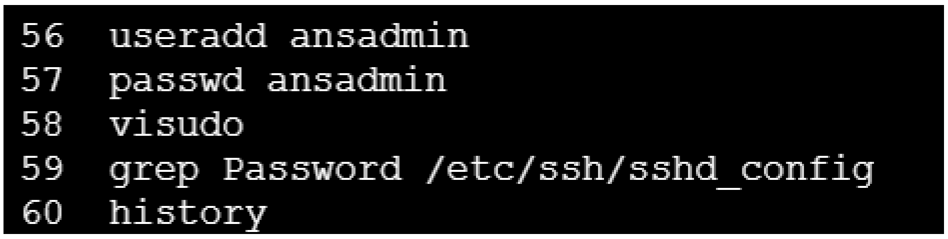
- **After generating switch back to root and install ansible**
- **amazon-linux-extras install ansible2**

INTEGRATING ASIBLE WITH DOCKER

PREPARE ASIBLE SYSTEM TO CREATE DOCKER IMAGES

ON Docker host:

- ***Create ansadmin***
- ***Add ansdmin to sudoers files : visudo***
- ***Enable password based login : edit /etc/ssh/sshd_config file***



```
56  useradd ansadmin
57  passwd ansadmin
58  visudo
59  grep Password /etc/ssh/sshd_config
60  history
```

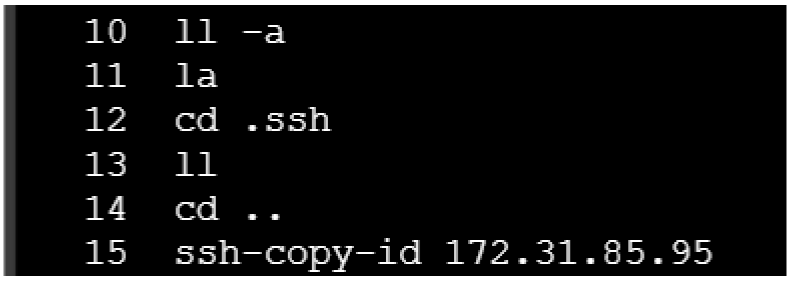
On Ansible Node: here we need to add dockerhost as a manged node in Ansible and we can do that in default file at /etc/ansible/hosts, delete everything there and add docker host ip address (private)

Now copy ansadminuser keys to target

- ***Add to hosts file***

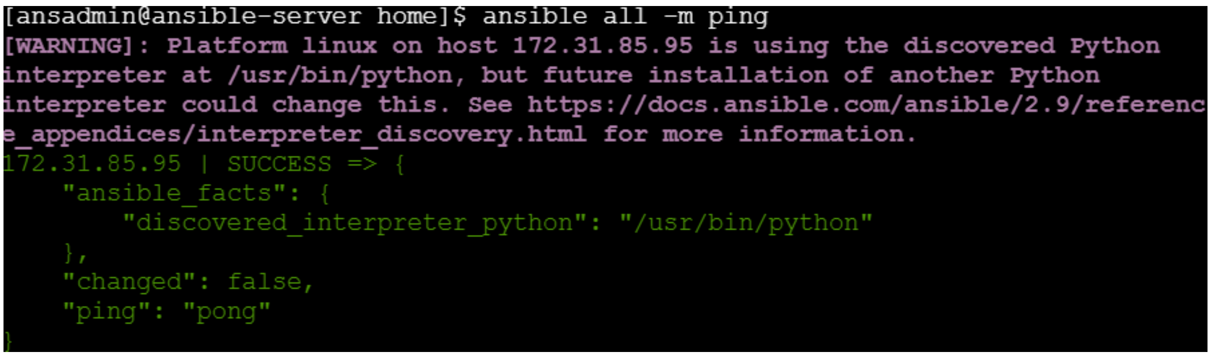
  vi /etc/ansible/hosts

- ***Copy ssh keys***



```
10  ll -a
11  la
12  cd .ssh
13  ll
14  cd ..
15  ssh-copy-id 172.31.85.95
```

➔ ***Test the connections***



```
[ansadmin@ansible-server home]$ ansible all -m ping
[WARNING]: Platform linux on host 172.31.85.95 is using the discovered Python
interpreter at /usr/bin/python, but future installation of another Python
interpreter could change this. See https://docs.ansible.com/ansible/2.9/referenc
e_appendices/interpreter_discovery.html for more information.
172.31.85.95 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python"
    },
    "changed": false,
    "ping": "pong"
}
```

NOTE: sudo systemctl restart sshd

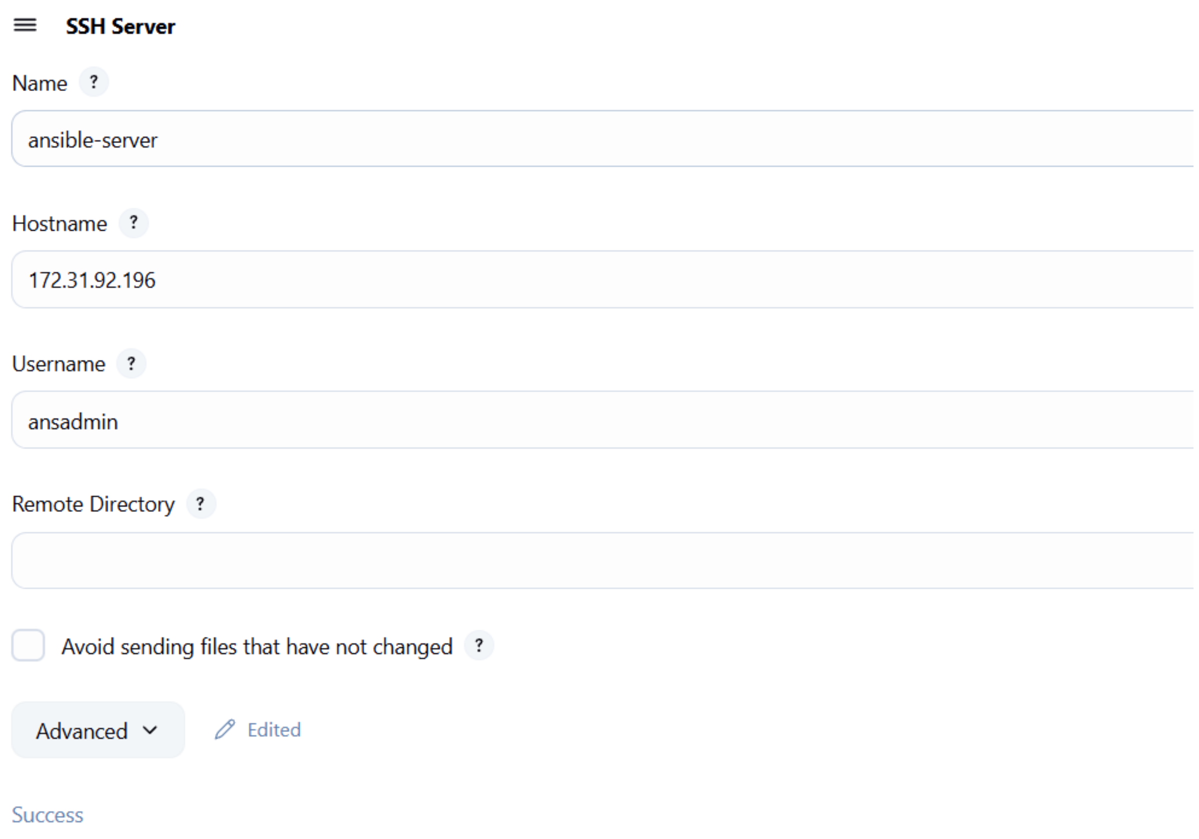
INTEGRATING ANSIBLE WITH JENKINS

Jenkins can able to copy artifacts onto Ansible systems, Ansible can able to create image or it can deploy the containers on Docker host.

➔ ***Jenkins will handle the build task***

➔ ***Ansible can take care of deployment tasks***

IN Jenkins system configure , add ansible



SSH Server

Name: ansible-server

Hostname: 172.31.92.196

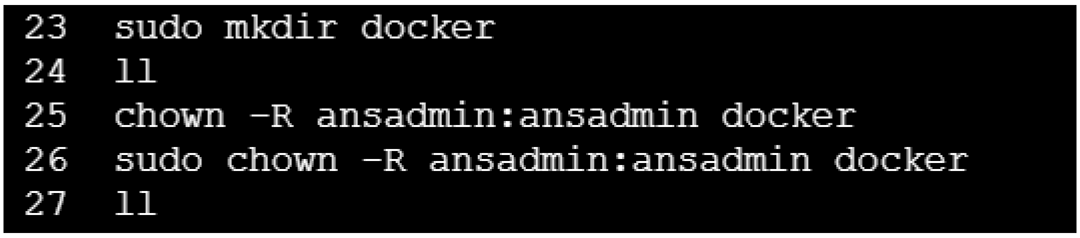
Username: ansadmin

Remote Directory:

☐ Avoid sending files that have not changed

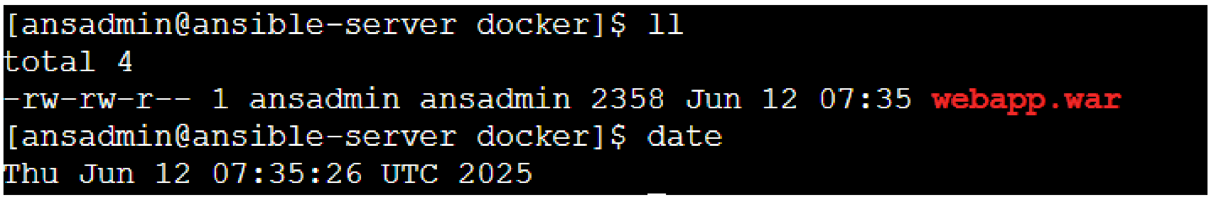
- ***Make a new Build by copying the old build , also chose Asinble server this time in send over build artifacts over ssh***

Note: make the dir in ansible server in /opt , wehee the artifacts will be copied



```
 23  sudo mkdir docker
 24  ll
 25  chown -R ansadmin:ansadmin docker
 26  sudo chown -R ansadmin:ansadmin docker
 27  ll
```

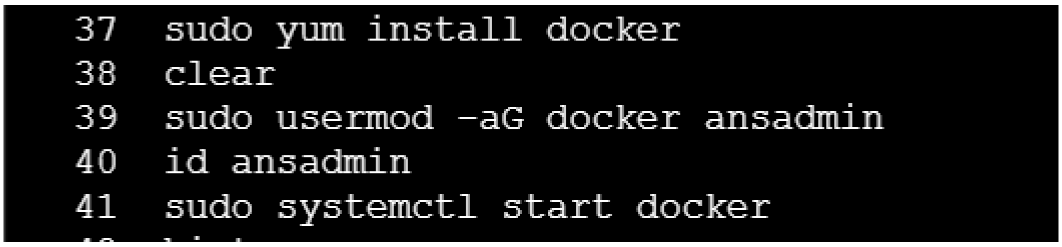
We can check thar artifacts are build in ansible serer



```
[ansadmin@ansible-server docker]$ ll
total 4
-rw-rw-r-- 1 ansadmin ansadmin 2358 Jun 12 07:35 webapp.war
[ansadmin@ansible-server docker]$ date
Thu Jun 12 07:35:26 UTC 2025
```

BUILDING IMAGE and Container on Ansible server

Do this first ,

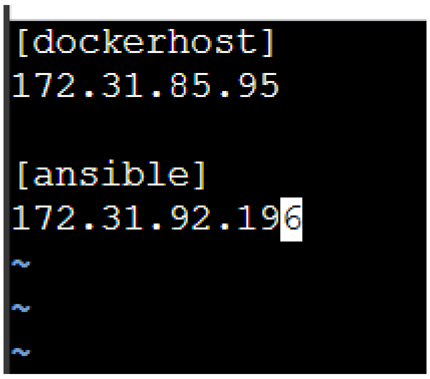


```
   37  sudo yum install docker
   38  clear
   39  sudo usermod -aG docker ansadmin
   40  id ansadmin
   41  sudo systemctl start docker
```

Make dockerfile and image and container and check

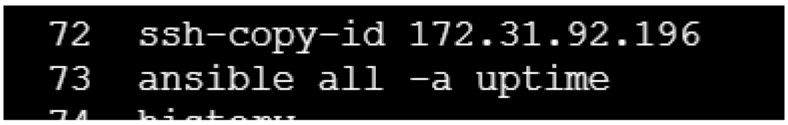
Creating Ansible Playbook which can do all this

Jus for convienec add , group In file of /etc/ansible/host



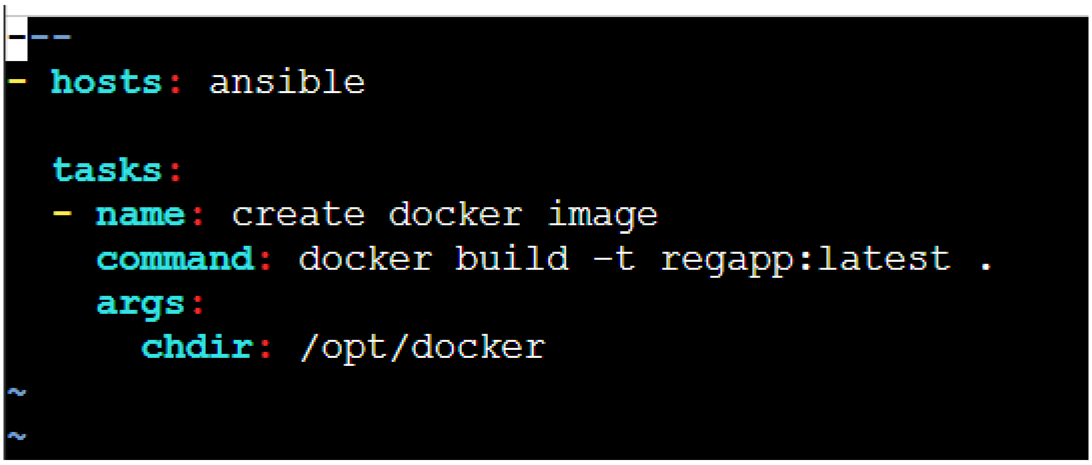
```
[dockerhost]
172.31.85.95

[ansible]
172.31.92.196
~
~
~
```

Copy ssh key onto our own system



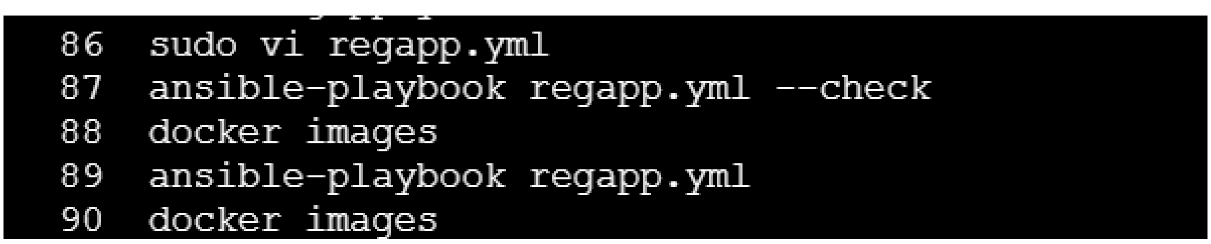
```
 72  ssh-copy-id 172.31.92.196
 73  ansible all -a uptime
 74  history
```

Do this after making playbook



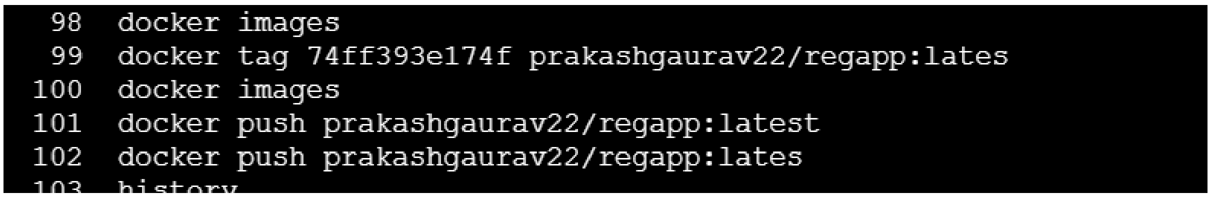
```
---
- hosts: ansible

  tasks:
  - name: create docker image
    command: docker build -t regapp:latest .
    args:
      chdir: /opt/docker
~
~
```

COPY IMAGE to DOckerhub



```
  86  sudo vi regapp.yml
  87  ansible-playbook regapp.yml --check
  88  docker images
  89  ansible-playbook regapp.yml
  90  docker images
```

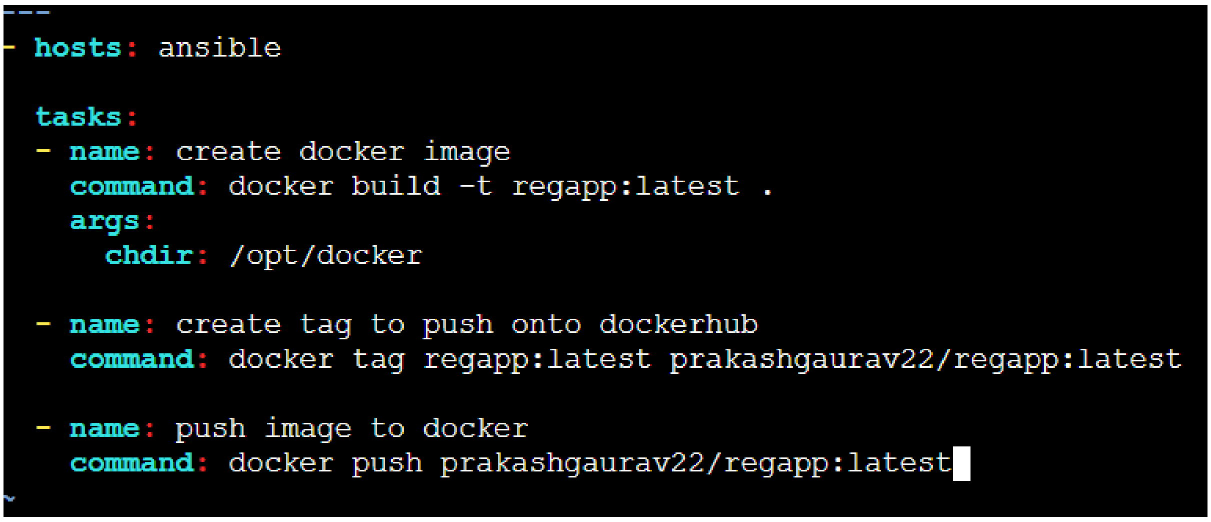
Push In Dockerhub after login



```
  98  docker images
  99  docker tag 74ff393e174f prakashgaurav22/regapp:lates
 100  docker images
 101  docker push prakashgaurav22/regapp:latest
 102  docker push prakashgaurav22/regapp:lates
 103  history
```

JENKINS JOB TO BUILD IMAGE ONTO ANSIBLE

➔ *Automate the docker push onto dockerhub by editing the playbook file*

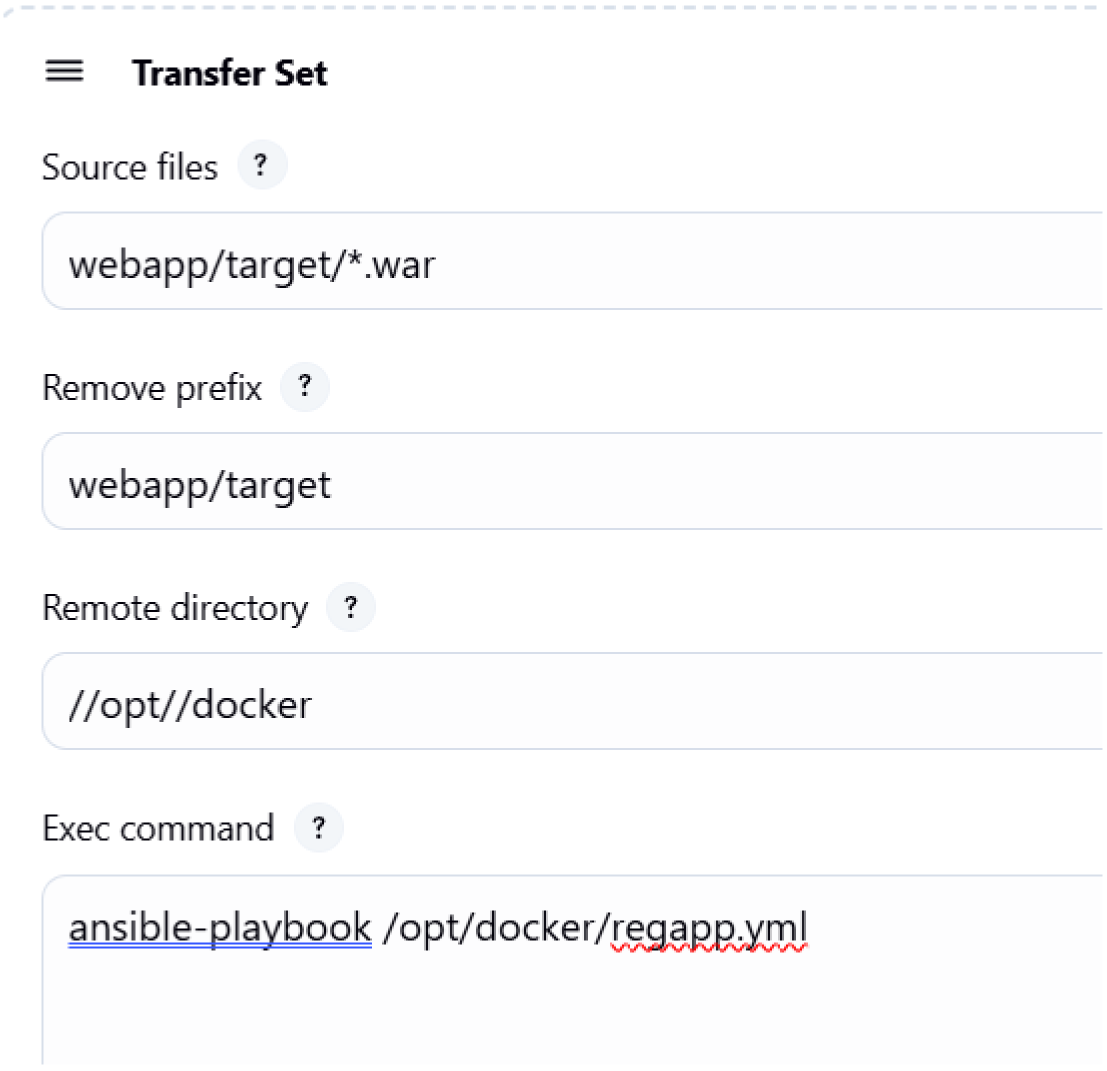


```
---
- hosts: ansible

  tasks:
  - name: create docker image
    command: docker build -t regapp:latest .
    args:
      chdir: /opt/docker

  - name: create tag to push onto dockerhub
    command: docker tag regapp:latest prakashgaurav22/regapp:latest

  - name: push image to docker
    command: docker push prakashgaurav22/regapp:latest
~
```

We can execute and check it as:
ansible-playbook regapp.yml –check
ansible-playbook regapp.yml   : it will execute it on all
ansible-playbook regapp.yml –limit 172.31.92.196

Now we are doing manually here , image cration and pushing , we need to add this playbook to Jenkins so it can be automated'

JUST ADD BELOW in THE CONFIG:


Transfer Set

Source files

webapp/target/*.war

Remove prefix

webapp/target

Remote directory

//opt//docker

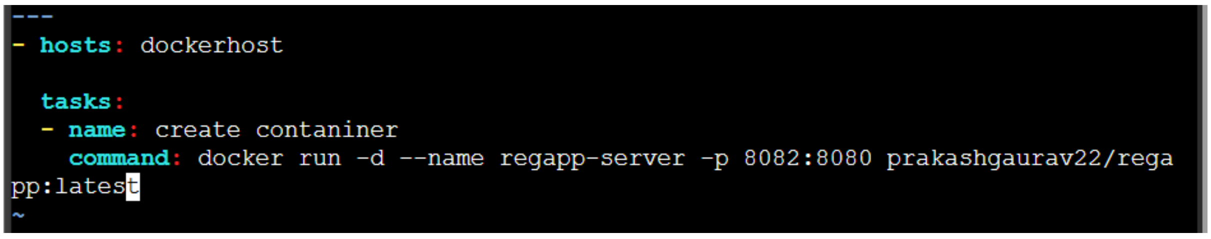
Exec command

```
ansible-playbook /opt/docker/regapp.yml
```

CRETING CONTANINER OUT of THAT IMAGE

: write a playbook o Ansible that tells to crate a new container on docker host after pulling image from dockerhub

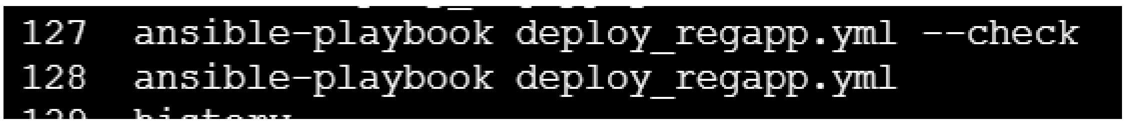
Create a new playbook as deploy_regapp.yml , this will have the command to tell the docker host to make a container



```
---
- hosts: dockerhost

  tasks:
  - name: create contaniner
    command: docker run -d --name regapp-server -p 8082:8080 prakashgaurav22/rega
pp:latest
~
```

NOTE: Give permission for the /var/run/docker.sock: as
chmod 777 /var/run/docker.sock

After that do this

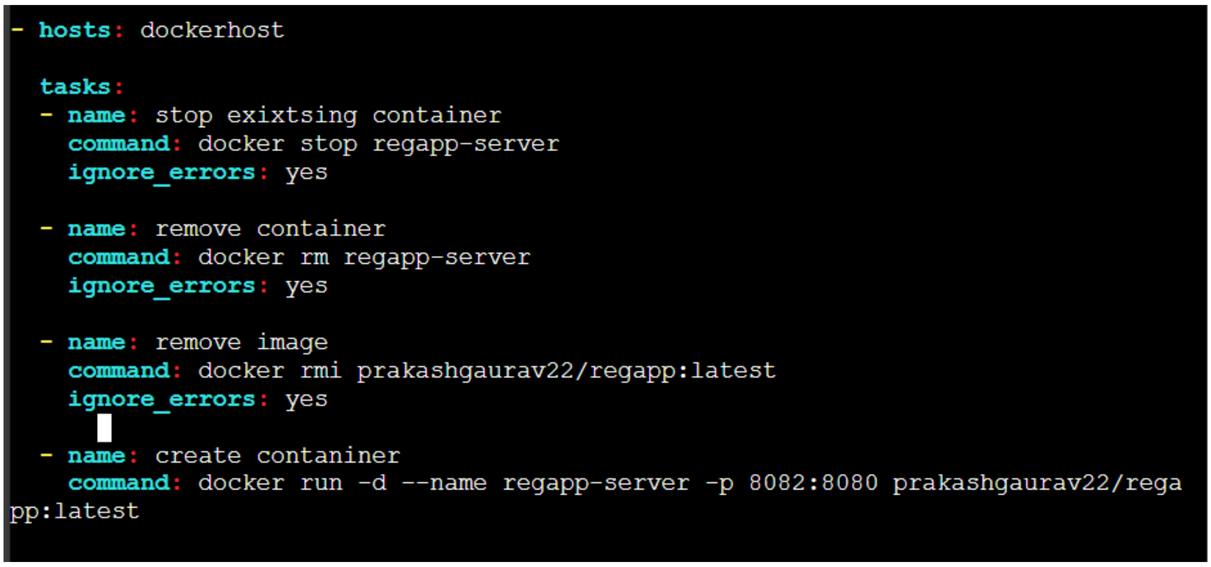


```
127  ansible-playbook deploy_regapp.yml --check
128  ansible-playbook deploy_regapp.yml
```

Also we might need to give permission the docker host for some error .

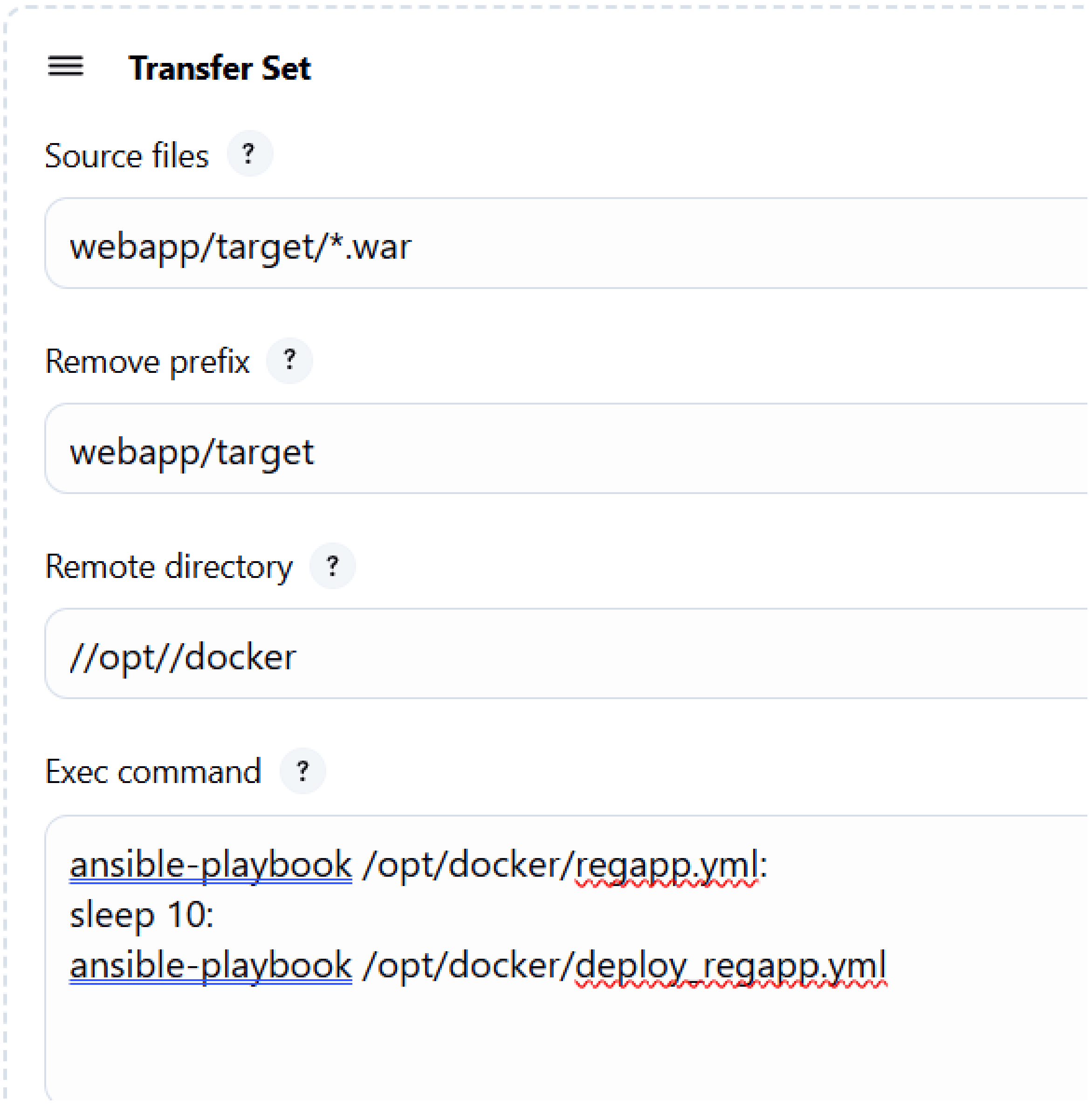
NOTE: if we try to run again the playbook , the previous issue of same sontainer name will arise and give error

TO SOLVE Edit playbook



```
- hosts: dockerhost

  tasks:
  - name: stop exixtsing container
    command: docker stop regapp-server
    ignore_errors: yes

  - name: remove container
    command: docker rm regapp-server
    ignore_errors: yes

  - name: remove image
    command: docker rmi prakashgaurav22/regapp:latest
    ignore_errors: yes

  - name: create contaniner
    command: docker run -d --name regapp-server -p 8082:8080 prakashgaurav22/rega
pp:latest
```

ADD the playbopok in Jenkins so all things becomes automated


Transfer Set

Source files

```
webapp/target/*.war
```

Remove prefix

```
webapp/target
```

Remote directory

```
//opt//docker
```

Exec command

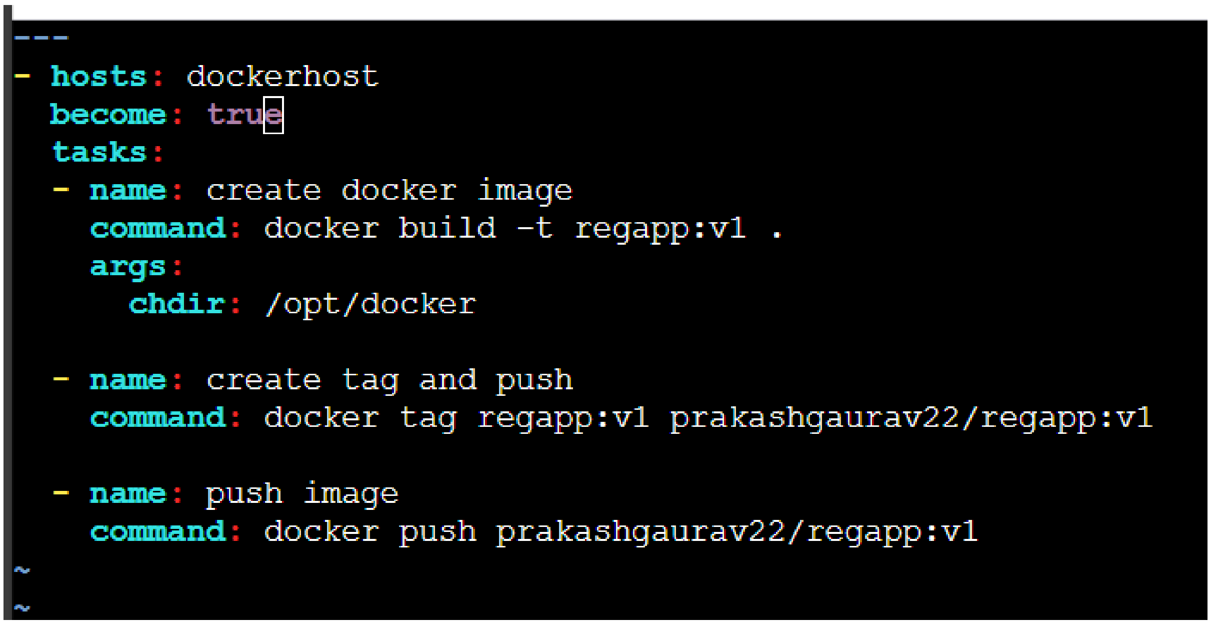
```
ansible-playbook /opt/docker/regapp.yml;
sleep 10;
ansible-playbook /opt/docker/deploy_regapp.yml
```

Section Conclusion:
All right, we have configured our Jenkins job in such a way that if somebody modify code it should automatically build the code, create a image, create a container and we could able to access those changes from the browser.But if you see the problem whenever there are some changes we are terminating the existing container and creating new container, during this time, end user cannot able to access the application. Another thing is if our container is terminated how we can come to know that it is not working? Or how we can create new container automatically? We don't have such kind of mechanisms over here,
that is where container management system comes into the picture.

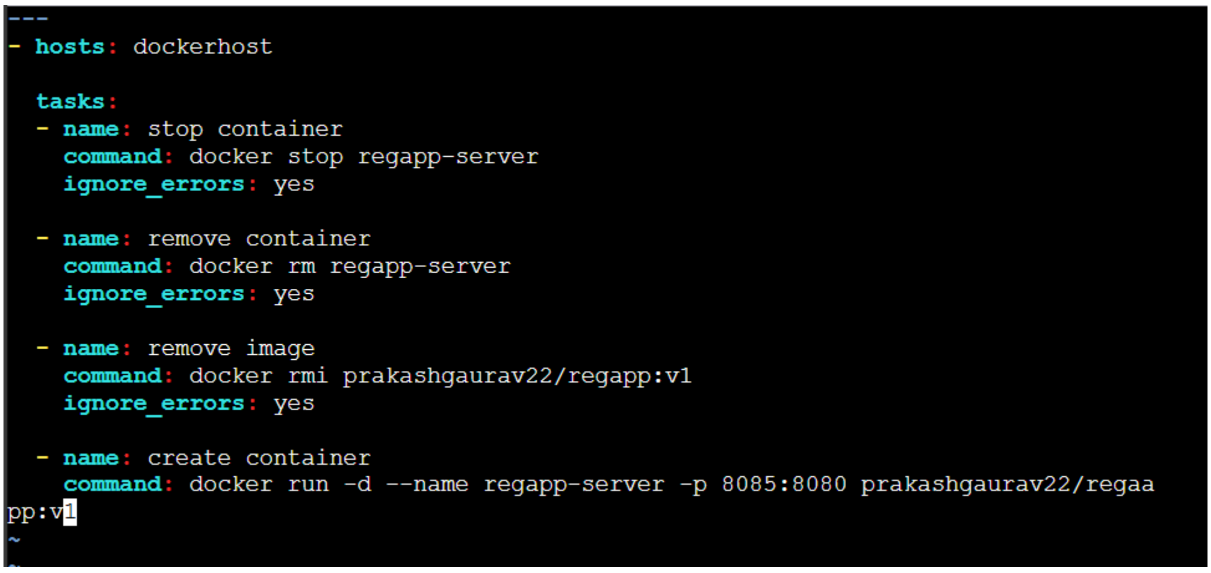
NOTE: A DIFFERENT APPROCH

➔ *I have tried using ansible as an orchestration only i.e. docker is not installed on this server and the image building , pushing and container creation all takes place on the docker-server*

➔ *Some things need to be kept in mind:*
   - *Add ansadmin to the docker group : usermod -aG docker ansadmn*
   - *Give chomd 777 /var/run/docker.sock*
   -



```
---
- hosts: dockerhost
  become: true
  tasks:
  - name: create docker image
    command: docker build -t regapp:v1 .
    args:
      chdir: /opt/docker

  - name: create tag and push
    command: docker tag regapp:v1 prakashgaurav22/regapp:v1

  - name: push image
    command: docker push prakashgaurav22/regapp:v1
~
~
```



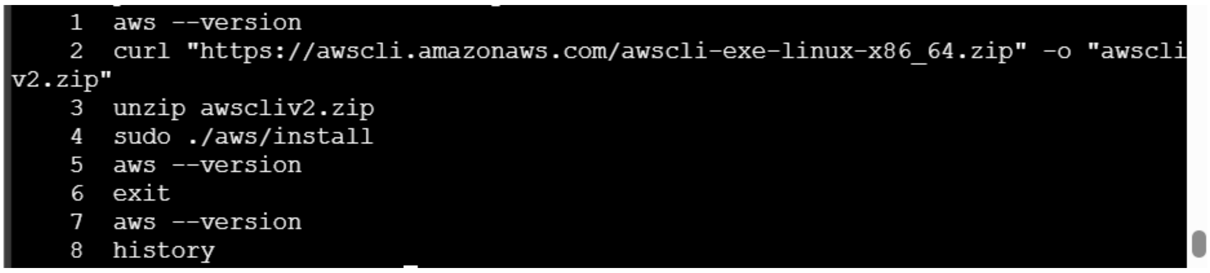
```
---
- hosts: dockerhost

  tasks:
  - name: stop container
    command: docker stop regapp-server
    ignore_errors: yes

  - name: remove container
    command: docker rm regapp-server
    ignore_errors: yes

  - name: remove image
    command: docker rmi prakashgaurav22/regapp:v1
    ignore_errors: yes

  - name: create container
    command: docker run -d --name regapp-server -p 8085:8080 prakashgaurav22/regaa
pp:v1
~
```

DEPLOYMENT on a as A POD

Kubernetes Installation:

➔ *Setting up on AWS EKS:*
We will be using EKSctl

➔ *Setting up Bootstarp server for eksctl*



```
    1  aws --version
    2  curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscli
v2.zip"
    3  unzip awscliv2.zip
    4  sudo ./aws/install
    5  aws --version
    6  exit
    7  aws --version
    8  history
```

➔

➔ *Install abd update aws version , above 2.2*

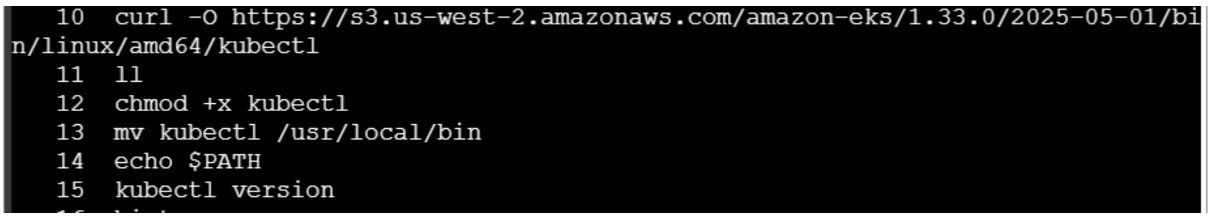
➔ *Link:*

curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"

unzip awscliv2.zip

sudo ./aws/install

➔ *Install kubectl*



```
   10  curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.33.0/2025-05-01/bi
n/linux/amd64/kubectl
   11  ll
   12  chmod +x kubectl
   13  mv kubectl /usr/local/bin
   14  echo $PATH
   15  kubectl version
```

➔ *Install eksctl from docs*

➔ *Set up IAM role in AWS:*
- *Ec2fullaccess*
- *Cloudformationfullaccess*
- *Iam*

➔ *After this give the role to the bootstrap server*

CREATING CLUSTER

*eksctl create cluster --name gaurav *

*--region us-east-1 *

*--node-type t2.small *

This will start creating a cluster , we can see in the cloudformation tab of aws, it wil take 20-25 mis.

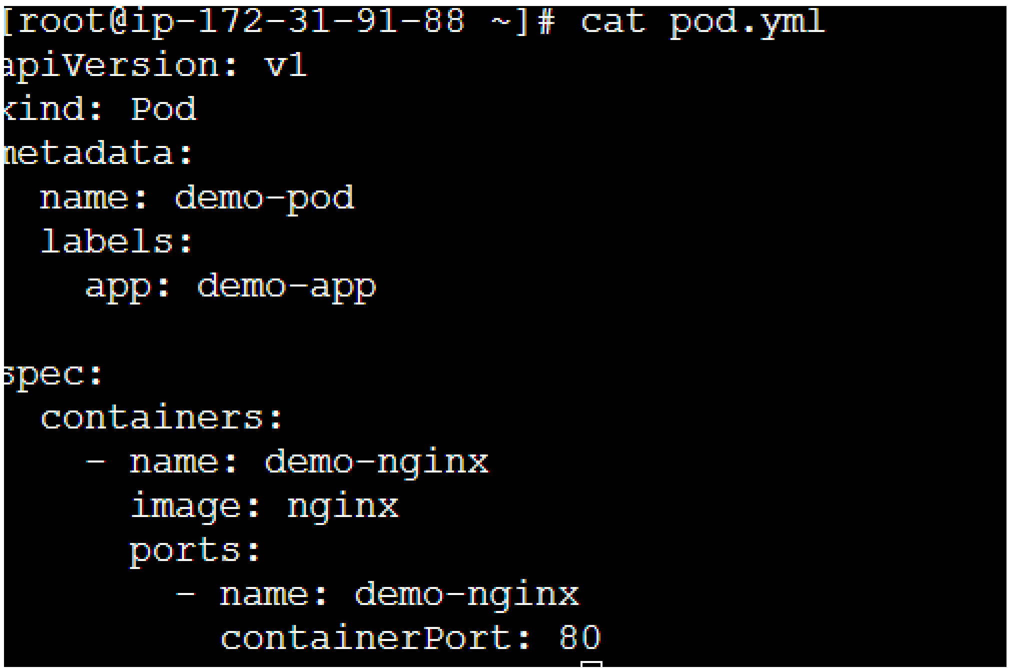
- *➔ /root/.kube/config is important file to access this cluster , anyone with this file can access and do the activities*
- *➔ Kubectl is used to communicate with our cluster*
- *➔ kubectl get nodes*
- *➔ kubectl delete cluster valaxy --region us-east-1*
- *➔ creating pods :*
    - *kubectl run weball --image=httpd*

DEPLOYING THROUGH COMMAND LINE

- *➔ deploying an Nginx container in a pod*
- *➔ creating pods with command:*
    - *kubectl create deployment demo-nginx –image=nginx –port=80 –replicas=2*
- *➔ kubectl get deplyments*
- *➔ kubectl get relpicaset*
- *➔ kubectl get pod*
- *➔ kubectl get all*
- *➔ Exposing to the external network:*
    - *Kubectl expose deployment demo-nginx –port=80 –type=LoadBalancer*
    - *As we have used a loadbalaner it will create a loadbalancer in aws and see in loadbalancer tab*
- *➔ Access it using the given external ip*

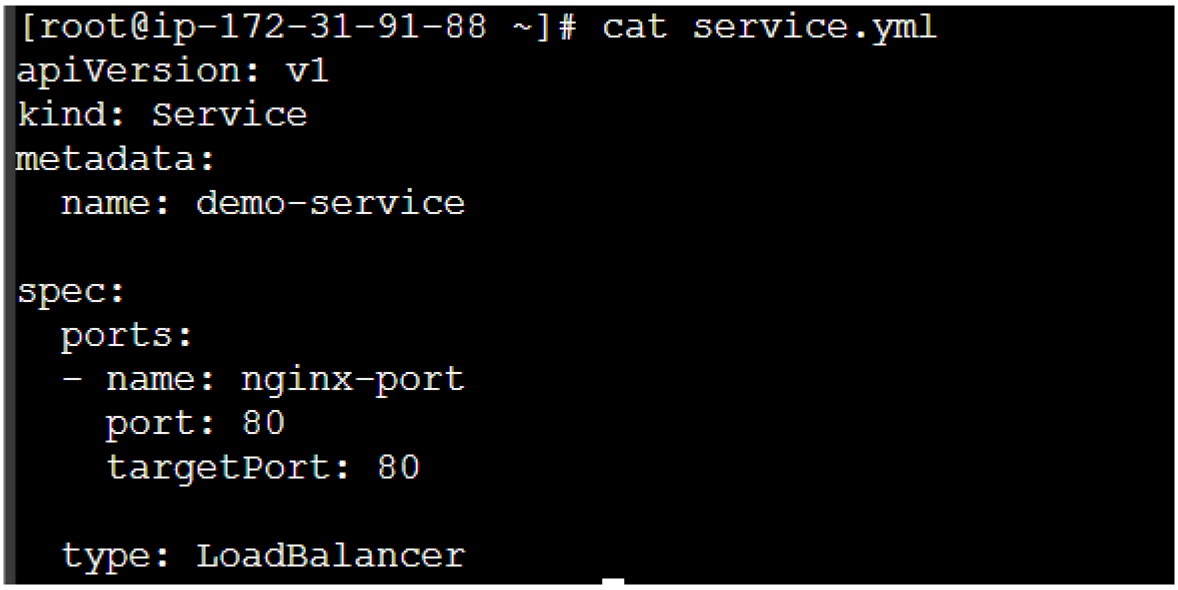
CRETING MANFIEST FILES TO DEPLOY

- ➔ *Delete previous deploymenst:*
    - *Kubectl delete deployment demo-nginx*
    - *kubectl delete service/demo-nginx*
- ➔ *Creating using .yml file*



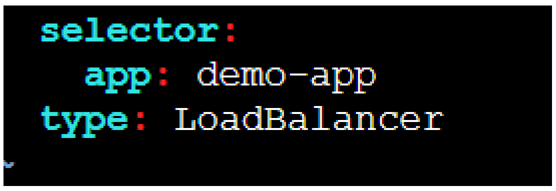
```
[root@ip-172-31-91-88 ~]# cat pod.yml
apiVersion: v1
kind: Pod
metadata:
  name: demo-pod
  labels:
    app: demo-app

spec:
  containers:
    - name: demo-nginx
      image: nginx
      ports:
        - name: demo-nginx
          containerPort: 80
```

- ➔ *Creating service*

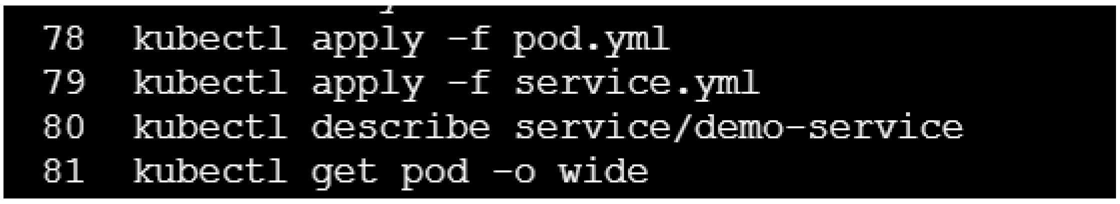


```
[root@ip-172-31-91-88 ~]# cat service.yml
apiVersion: v1
kind: Service
metadata:
  name: demo-service

spec:
  ports:
  - name: nginx-port
    port: 80
    targetPort: 80

  type: LoadBalancer
```

NOTE: The loadbalacer will show instances are outofservice as , we have not added selectors in Service.yml and labels in pod.yml , this is necessary as only this tells , service should transfer to which pod add this ,:

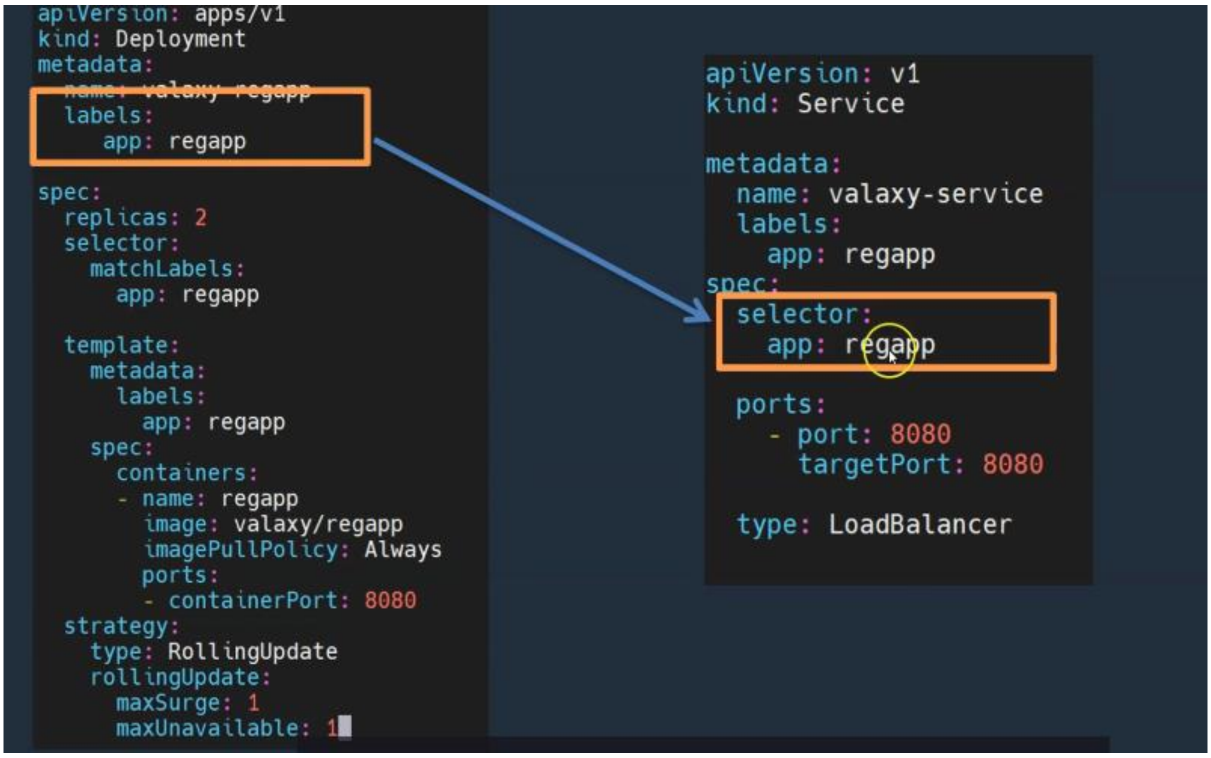


```
 selector:
   app: demo-app
 type: LoadBalancer
```



```
 78  kubectl apply -f pod.yml
 79  kubectl apply -f service.yml
 80  kubectl describe service/demo-service
 81  kubectl get pod -o wide
```

WRITING A DEPLOYMENT :



```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: valaxy-regapp
  labels:
     app: regapp

spec:
  replicas: 2
  selector:
    matchLabels:
      app: regapp

  template:
    metadata:
      labels:
        app: regapp
    spec:
      containers:
      - name: regapp
        image: valaxy/regapp
        imagePullPolicy: Always
        ports:
        - containerPort: 8080
  strategy:
    type: RollingUpdate
    rollingUpdate:
      maxSurge: 1
      maxUnavailable: 1
```

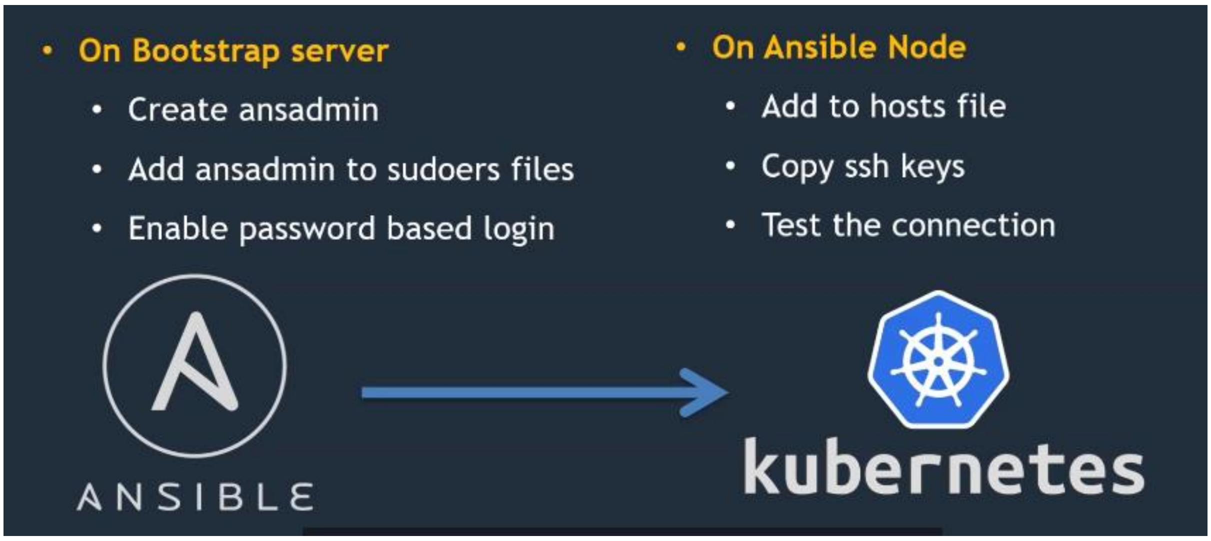
```yaml
apiVersion: v1
kind: Service

metadata:
  name: valaxy-service
  labels:
    app: regapp
spec:
  selector:
    app: regapp

  ports:
    - port: 8080
      targetPort: 8080

  type: LoadBalancer
```

- ➔ *It has replica sets and rolling updates*
- ➔ *Apply both deployment and service*

kubectl describe/service/valaxy-service

- ➔ *GO to AWS loadbalancer and access through DNS , also end ith :8080*
- ➔ *Demonstrate pod creation by deleting it*

INEGRATING KUBERNETES CLUSTER WITH ANSIBLE

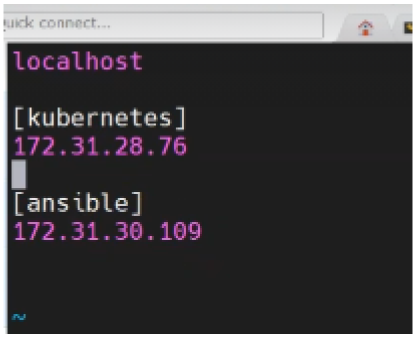


- On Bootstrap server
  - Create ansadmin
  - Add ansadmin to sudoers files
  - Enable password based login
- On Ansible Node
  - Add to hosts file
  - Copy ssh keys
  - Test the connection

ANSIBLE
kubernetes

Now In our Ansible server , inside our previous Docker directory, the previous playbook of creating the image and pushing it to the dockerhub will not change , but for the k8s deployment we need new playbook .

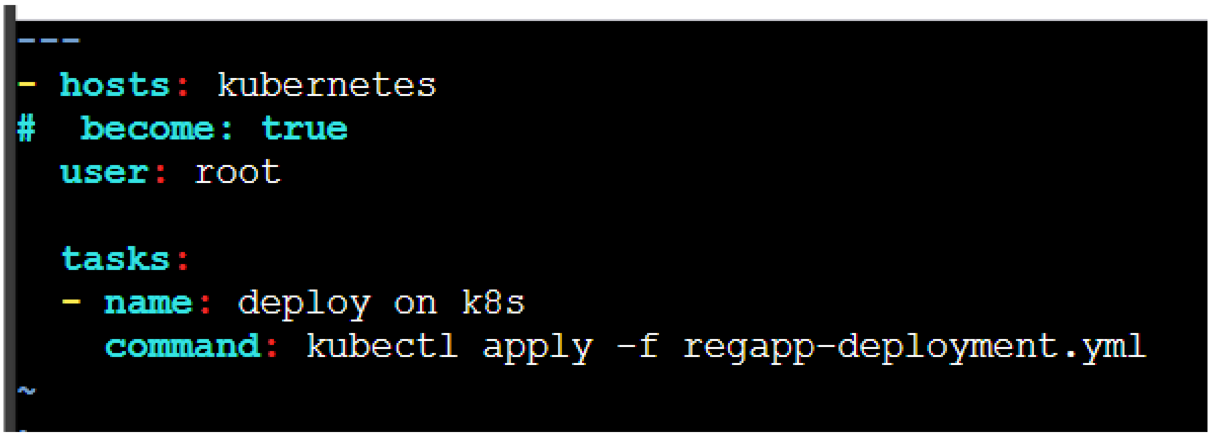
- *Alos we make a new inventory file here : as vi hosts*



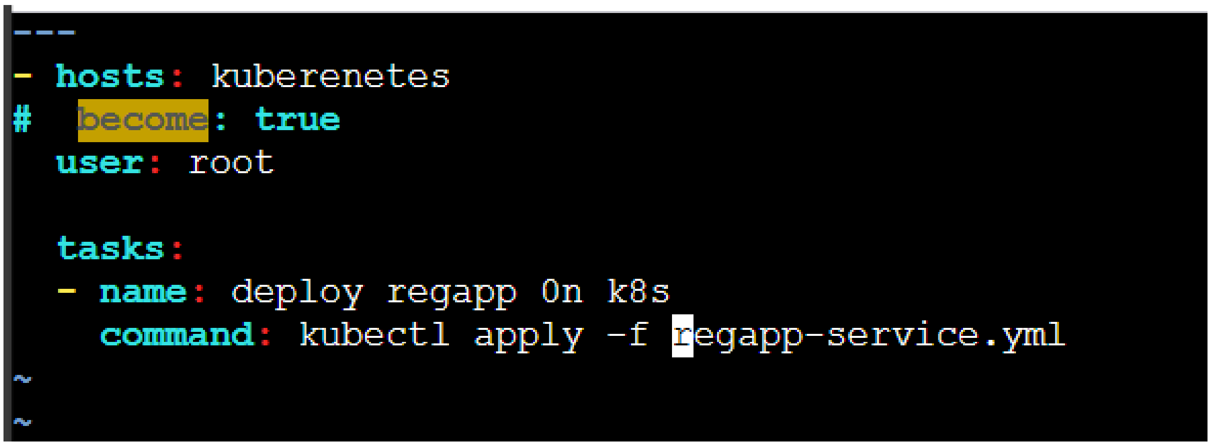
```
localhost

[kubernetes]
172.31.28.76

[ansible]
172.31.30.109
```

- ➔ *After thic copy the key to the bootstrap server:*
  - *Ssh-copy-id ip*
- ➔ *Now to use the current location hosts file we use :*
  - *ansible -i hosts all -a uptime*

Make a kube_deployment.yml and service file over here in /opt/docker in ansible



```
---
- hosts: kubernetes
#   become: true
  user: root

  tasks:
  - name: deploy on k8s
    command: kubectl apply -f regapp-deployment.yml
~
```



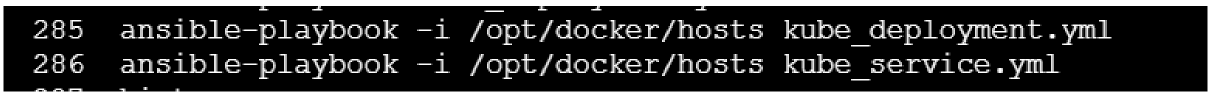
```
---
- hosts: kuberenetes
#   become: true
  user: root

  tasks:
  - name: deploy regapp 0n k8s
    command: kubectl apply -f regapp-service.yml
~
~
```

- ➔ *Before applying the ansible playbook , delete the deployment and service file previously , made in bootstrap server*
  - *kubectl delete -f regapp-deployment.yml*
  - *kubectl delete -f regapp-service.yml*

we need to run it in root user , otherwise it will give error , ALSO WE NEED TO DO:

- *ssh-copy-id ip_of__root: copy ansadmin key to the root user*
  - *ssh-copy-id root@...*
- *We need to set the password of root user of bootstrap before;*
  - *Passwd root*

➔ *After this execute the playbook*

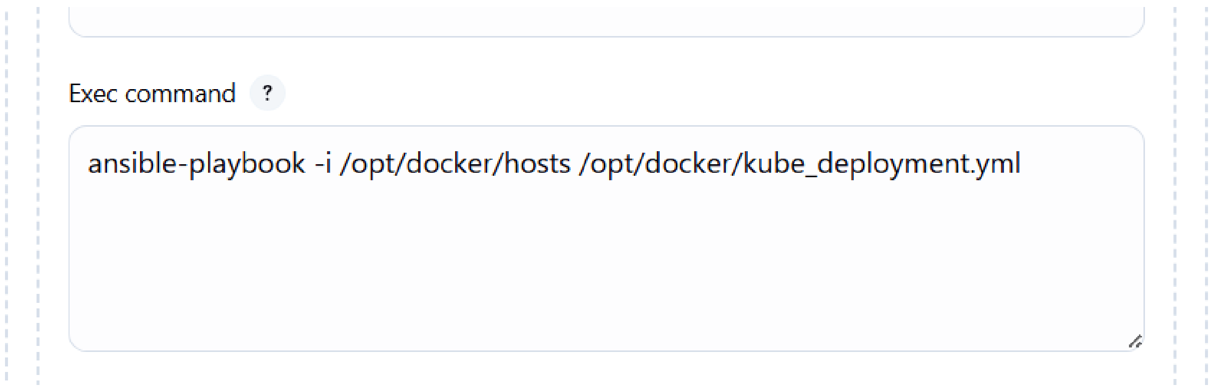


```
 285  ansible-playbook -i /opt/docker/hosts kube_deployment.yml
 286  ansible-playbook -i /opt/docker/hosts kube_service.yml
```

# USING JENKINS TO EXECUTE thise jobs

- ➔ *Setting up Jenkins*
- ➔ *Make a new freestyle project with Post buold action of send build artifacts over ssh*
- ➔ *Choose ansible server*
- ➔ *In the ecec shell ecxecute these commands*

:Merge both the deployment and service playbook into one and make changes as neede

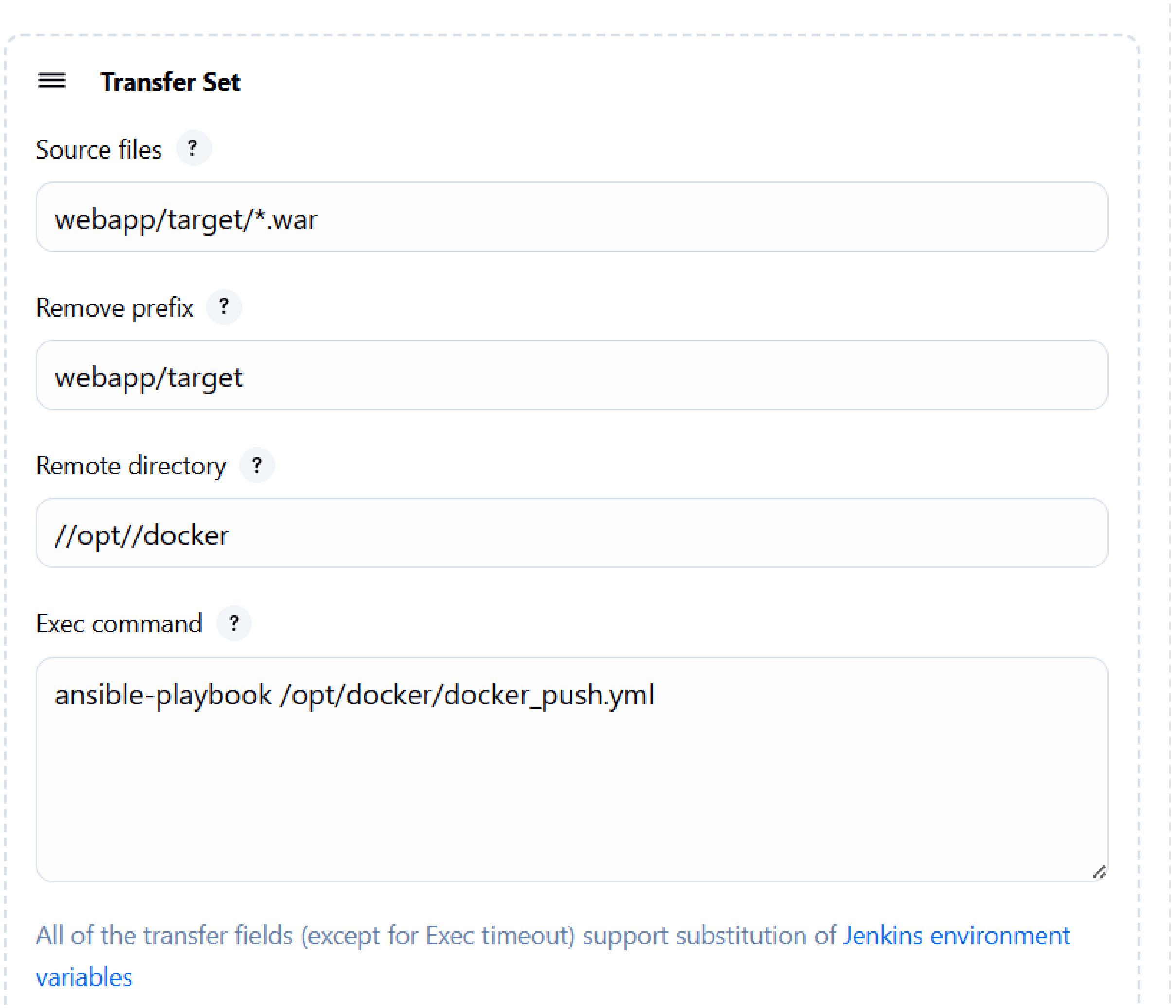


```
ansible-playbook -i /opt/docker/hosts /opt/docker/kube_deployment.yml
```

- ➔ *This wil make deployment and service in our bootstrap server*

:: CI JOB to create image for K8s

- ➔ *Make a new job , Regapp_CI_Job , copy it from artifacts one*
- ➔ *Or make a new job as , Build code with help of maven and create an image on ansible and push it onto dockerhib*



Transfer Set

Source files ?

webapp/target/*.war

Remove prefix ?

webapp/target

Remote directory ?

//opt//docker

Exec command ?

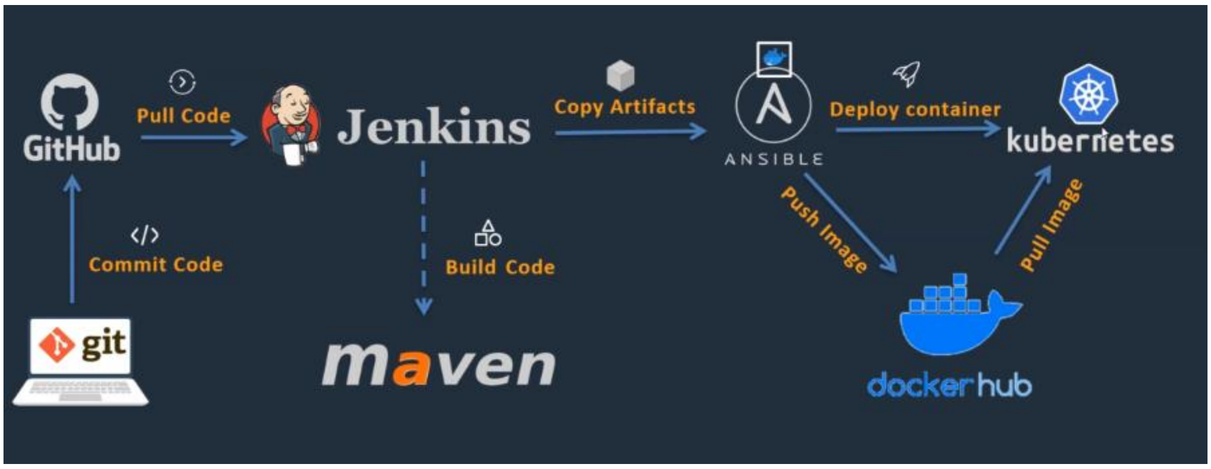
```
ansible-playbook /opt/docker/docker_push.yml
```

All of the transfer fields (except for Exec timeout) support substitution of Jenkins environment variables

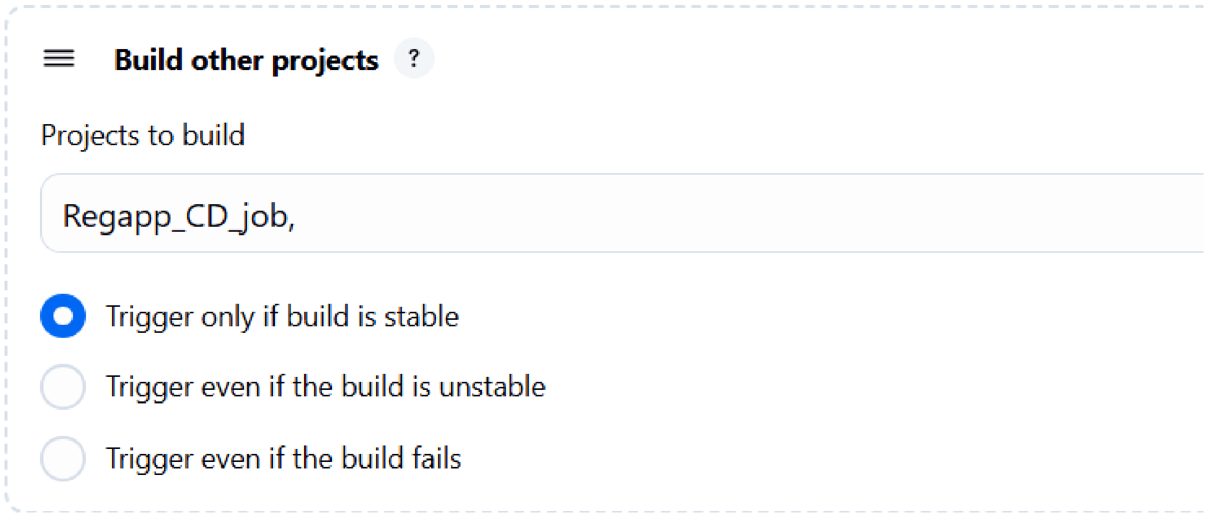
- ➔ *This job will only do the part of making image from artifacta and imge pushing to docker husb*
- ➔ *Of CD we have different job regapp_CD_Job whhihc we made earlier*

Now we just need to integrate our CD and CI jobs

- ➔ *Our CI/Cd so far..*



➔ *To integrate both jobs , in Ci job go to post build action and select , BUILD OTHER PROJECTS*



Build other projects ?

Projects to build

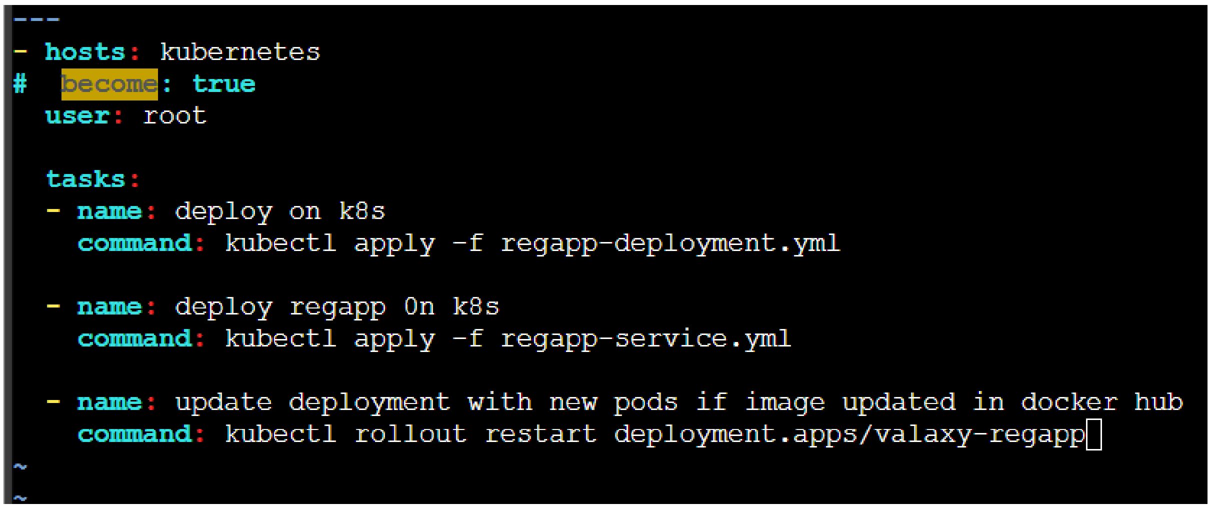
Regapp_CD_job,

[✓ Trigger only if build is stable]

Trigger even if the build is unstable

Trigger even if the build fails

Note: If we build now , or any update is made in the project , the new pods will not be reflected as we have not added the rolling updates command.
Add it in the playbook kube_deploy.yml
- name: update deployment with new pods if image updated in docker hub
command: kubectl rollout restart deployment.v1.apps/valaxy-deployment



```
---
- hosts: kubernetes
#  become: true
  user: root

  tasks:
  - name: deploy on k8s
    command: kubectl apply -f regapp-deployment.yml

  - name: deploy regapp On k8s
    command: kubectl apply -f regapp-service.yml

  - name: update deployment with new pods if image updated in docker hub
    command: kubectl rollout restart deployment.apps/valaxy-regapp
~
~
```