**Code**

**https://github.com/ManoharShetty507/CI-CD-Deployment-Jenkins-Ansible-Kubernetes.git**

**Create a Server in AWS**

Ec2 – t2-Micro Server

**Install Docker**

sudo apt update

sudo apt install apt-transport-https ca-certificates curl software-properties-common

curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add –

sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu focal stable"

apt-cache policy docker-ce

sudo apt install docker-ce

sudo systemctl status docker

**Create Tomcat Container**

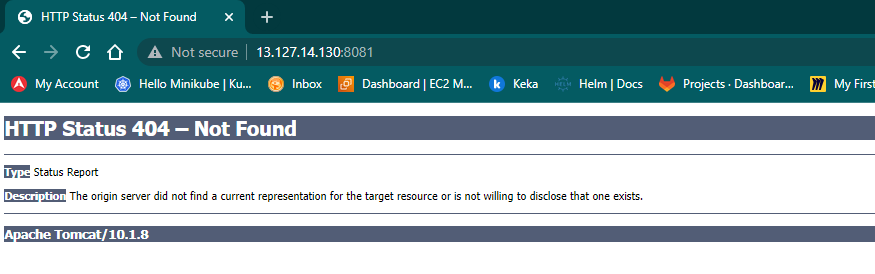
vim /etc/hostname

dockerhost

**Pull the image and Run the Container**

docker pull tomcat

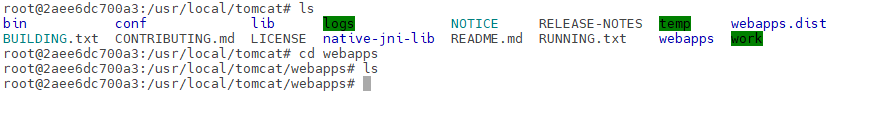
docker run -d --name tomcat-container -p 8081:8080 tomcat



**It is not able to get the page we are requesting**

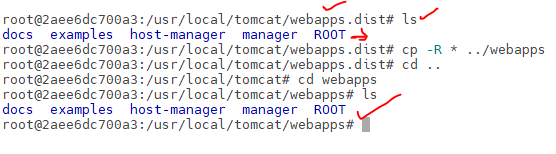
Login to Tomcat Container

docker exec -it tomcat-container /bin/bash

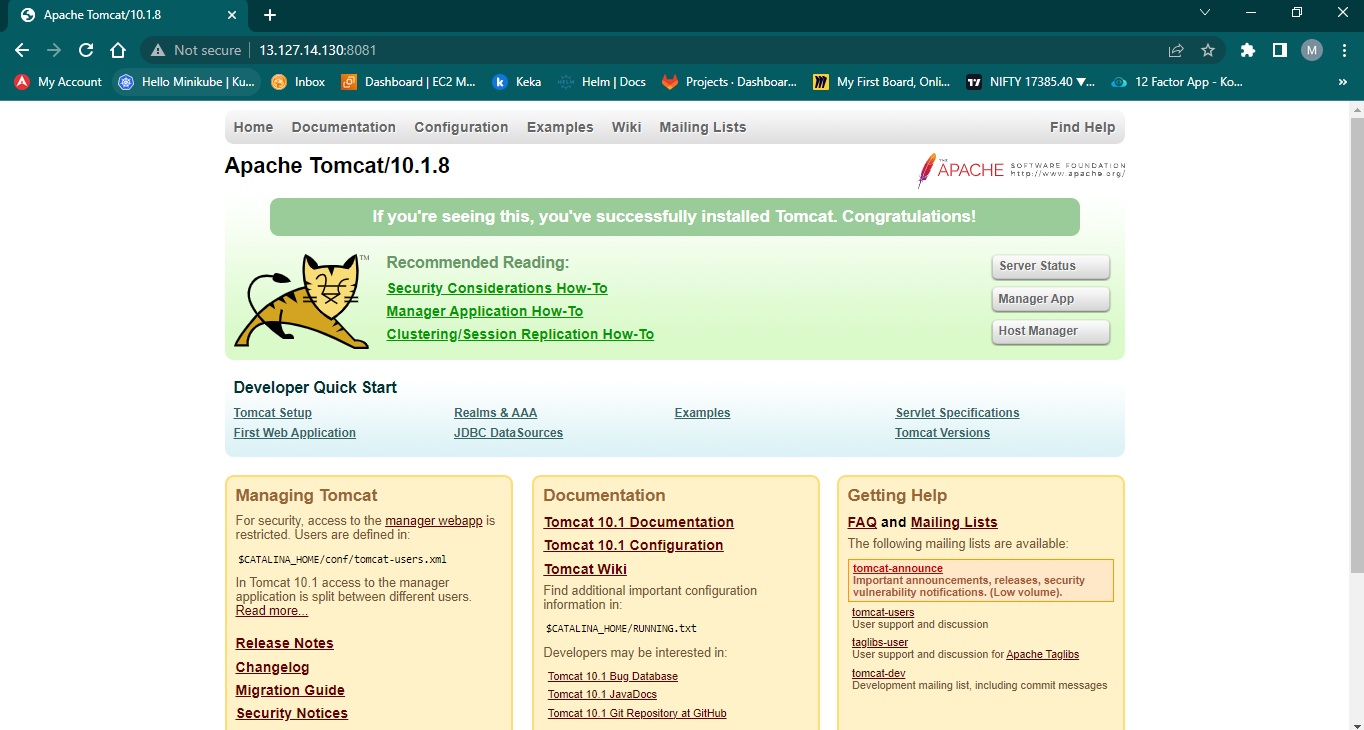


Everything is there in webapps.dist

We need to copy everything from webapps.dist to webapps



**Able to Access in the Browser**



As the Container is Temporary – We can create a Dockerfile and deploy it

FROM ubuntu:latest

RUN apt-get update && \

    apt-get install -y openjdk-8-jdk

RUN mkdir /opt/tomcat

WORKDIR /opt/tomcat

ADD https://dlcdn.apache.org/tomcat/tomcat-9/v9.0.74/bin/apache-tomcat-9.0.74.tar.gz .

RUN tar -xvxf apache-tomcat-9.0.74.tar.gz

RUN mv apache-tomcat-9.0.74/\* /opt/tomcat

EXPOSE 8080

CMD ["/opt/tomcat/bin/catalina.sh", "run"]

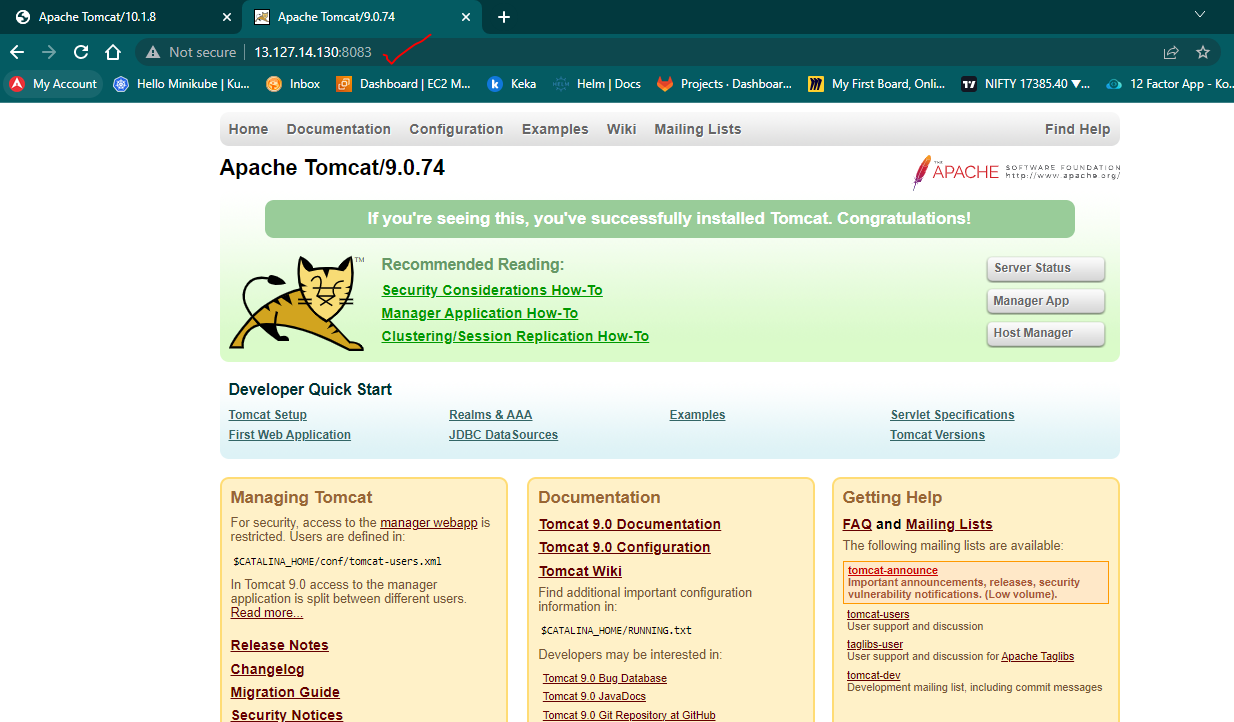
docker build -t mytomcat .

docker images

docker ps

docker run -d --name mytomcat-server -p 8083:8080 mytomcat

docker ps



Rather than logging into the container – rebuid the docker image using the default image – tweek certain things in the new image

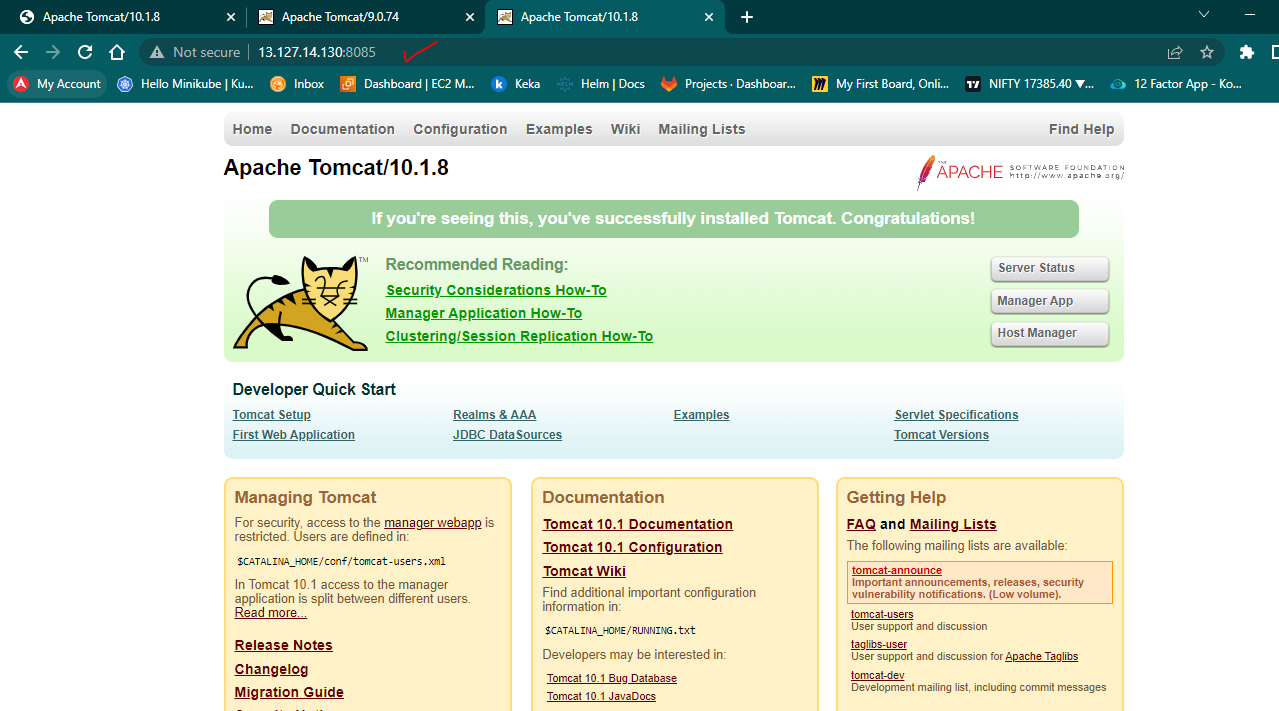
Customized Tomcat File

FROM tomcat:latest

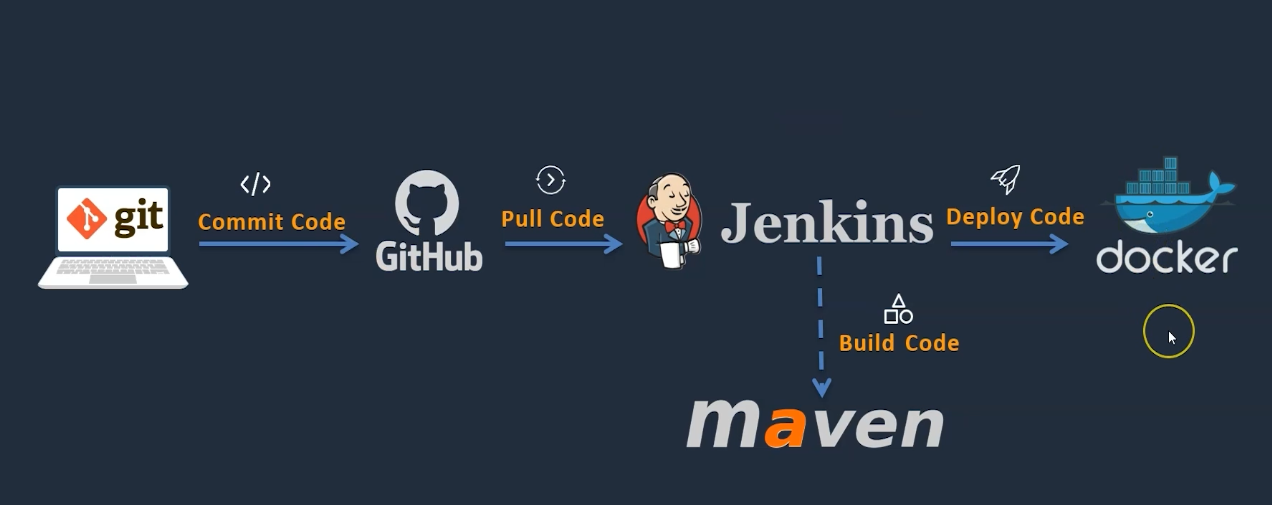
RUN cp -R /usr/local/tomcat/webapps.dist/\* /usr/local/tomcat/webapps

docker build -t demotomcat .

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



**Integrate Dockerhost with the Jenkins to Deploy the Application**



**Install Jenkins**

**Install Java**

sudo apt update

sudo apt install openjdk-11-jre

java -version

**Install Jenkins**

curl -fsSL https://pkg.jenkins.io/debian/jenkins.io-2023.key | sudo tee \

/usr/share/keyrings/jenkins-keyring.asc **>** /dev/null

echo deb **[**signed-by**=**/usr/share/keyrings/jenkins-keyring.asc] \

https://pkg.jenkins.io/debian binary/ | sudo tee \

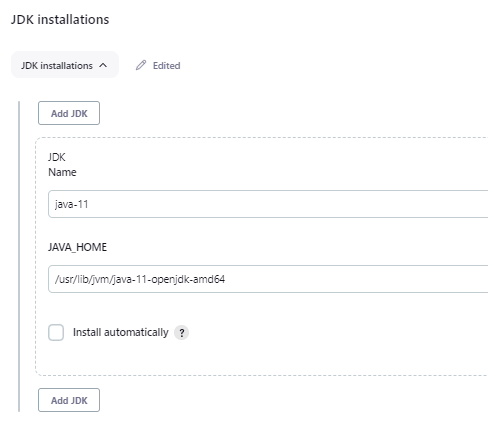
/etc/apt/sources.list.d/jenkins.list **>** /dev/null

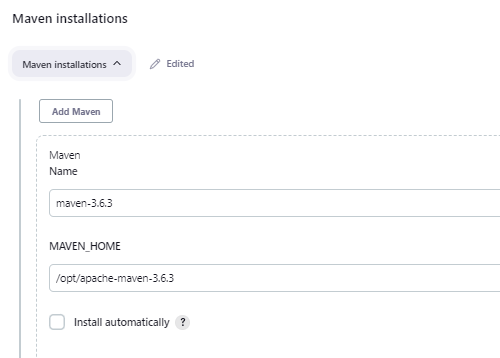
sudo apt-get update

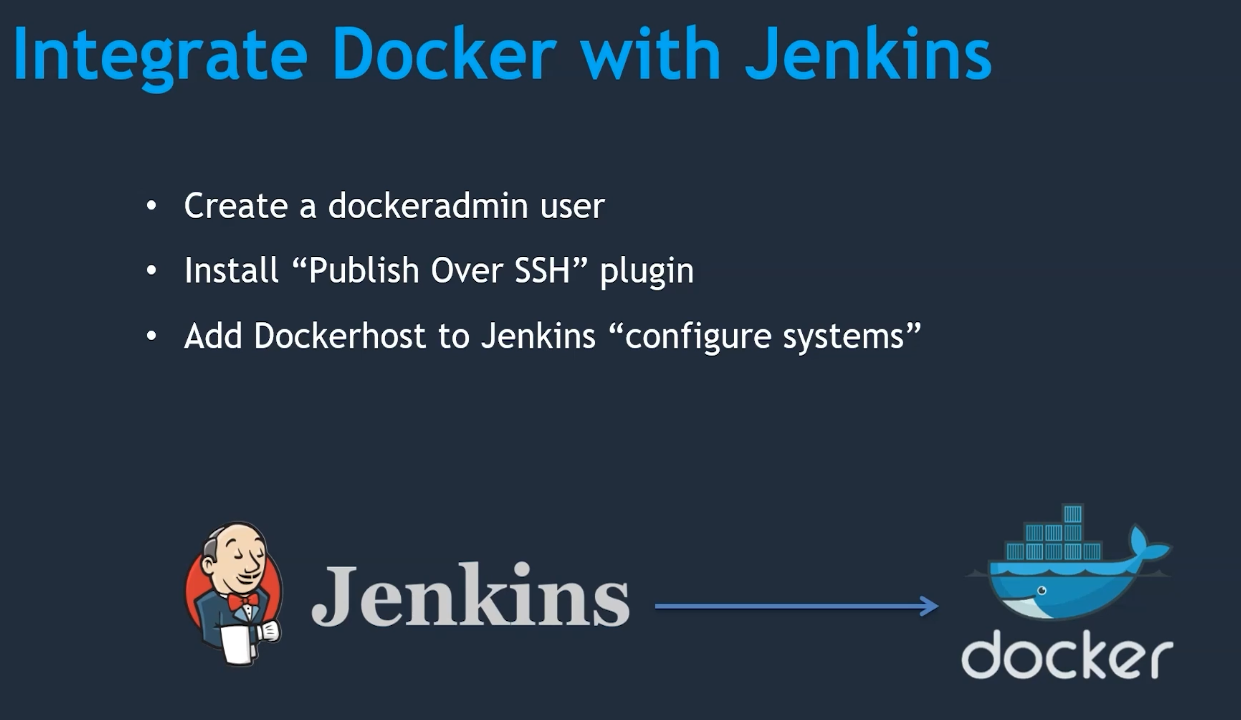
sudo apt-get install jenkins

**Access the Jenkins in Browser**

Update the Tools (Java & Maven) Path in Jenkins – Manage Jenkins -> Tools







**Create a User in Dockerhost**

cat /etc/passwd

cat /etc/group

docker group is already there.

**Create a user and add the user to docker group.**

adduser dockeradmin

passwd dockeradmin

id dockeradmin

usermod -aG docker dockeradmin



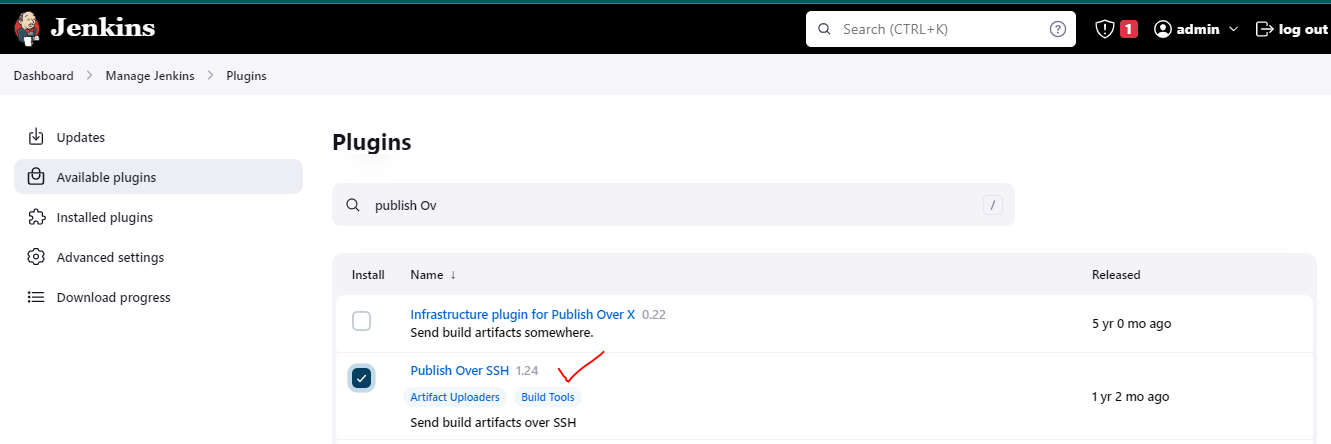
By Default, ec2 instance doesn’t allow passwordless authentication. We need to enable it.

Vi /etc/ssh/sshd\_config

**PasswordAuthentication yes**

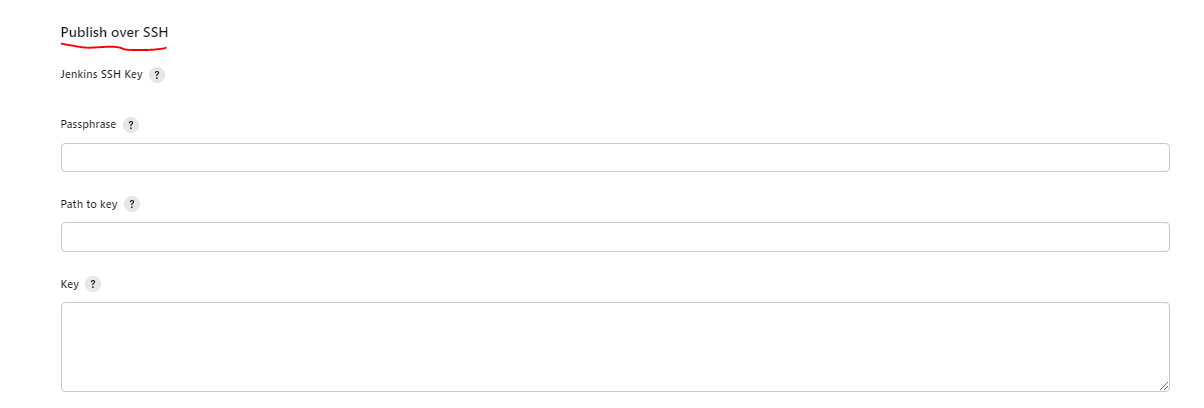
systemctl reload ssh

**Add a Plugin in Jenkins – To publish the Artifacts from Docker to Jenkins over SSh**



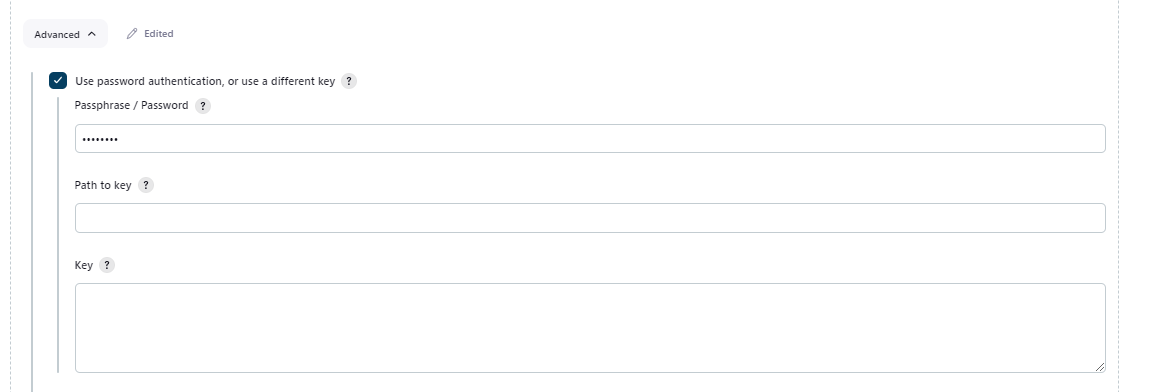
**Configure Dockerhost in Jenkins**

**Manage Jenkins -> Configure System/System -> Publish Over SSH**





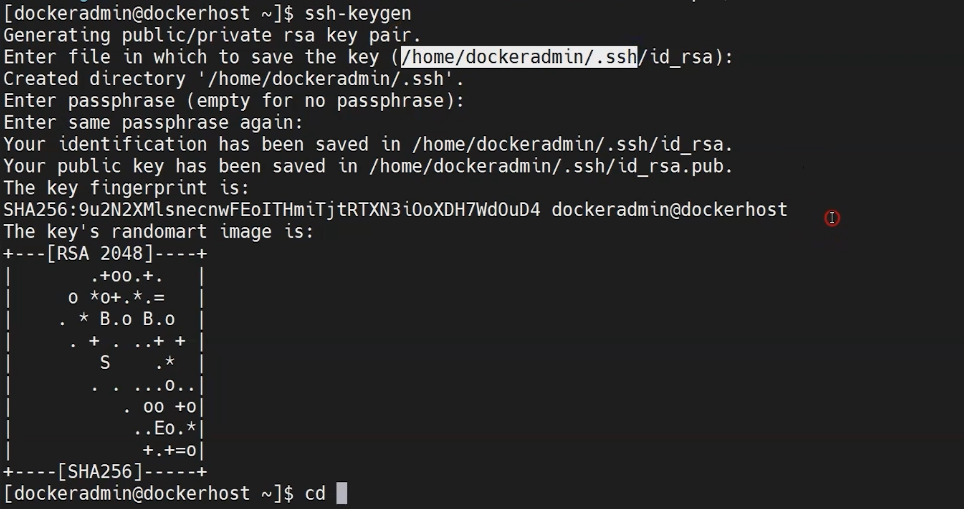
**Click on Advanced**



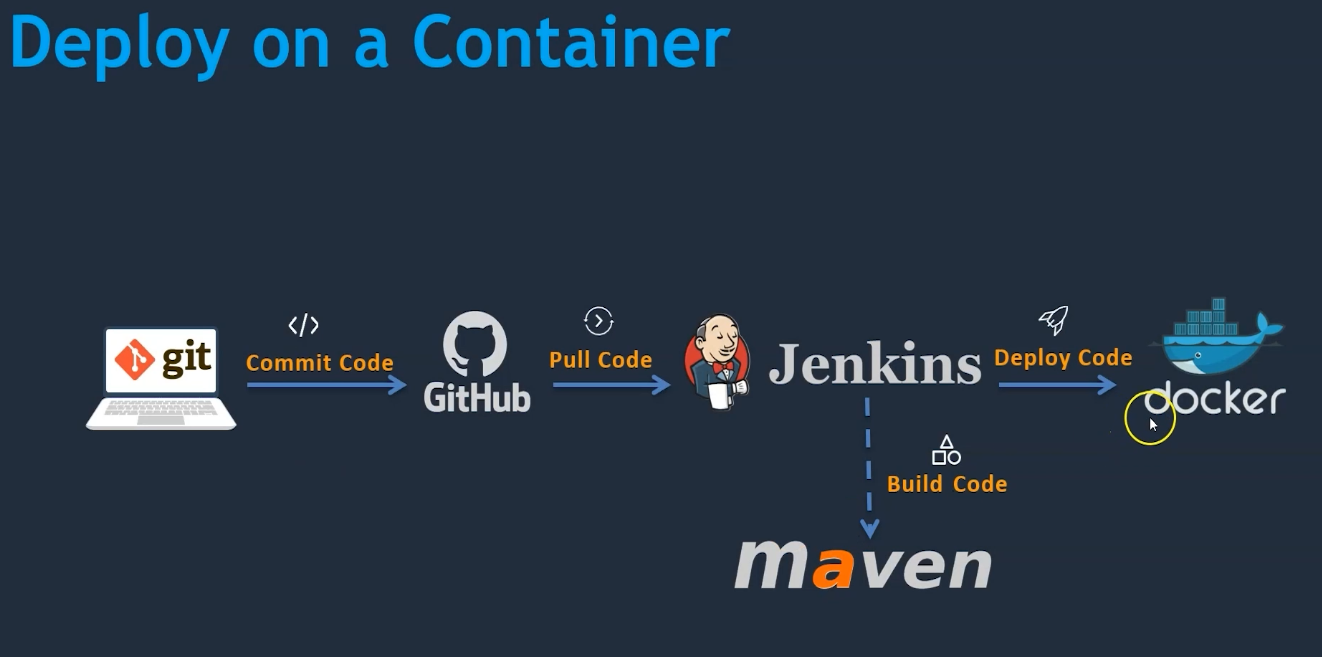
**Test The Configuration**



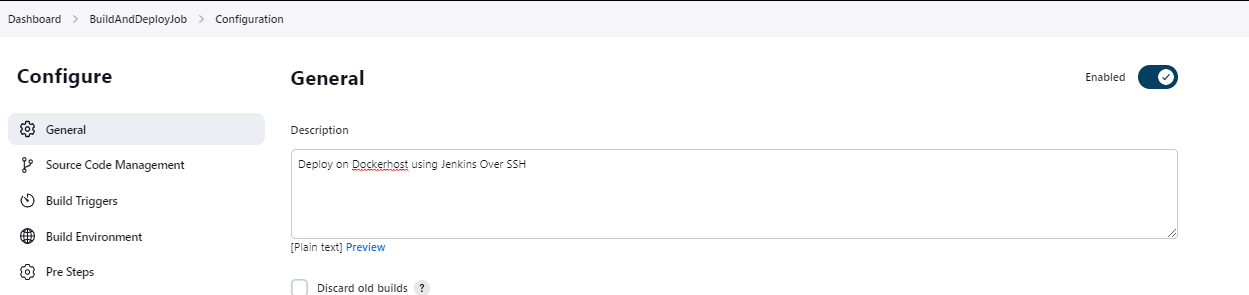
**You can generate the SSH key and use SSH key as well rather than password**

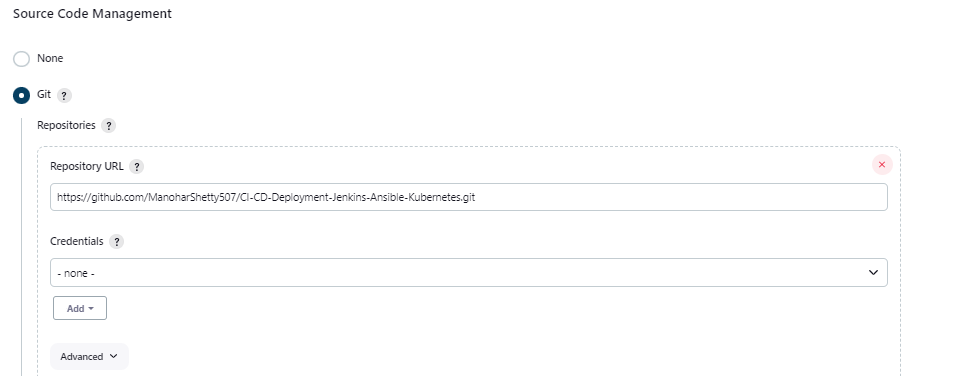


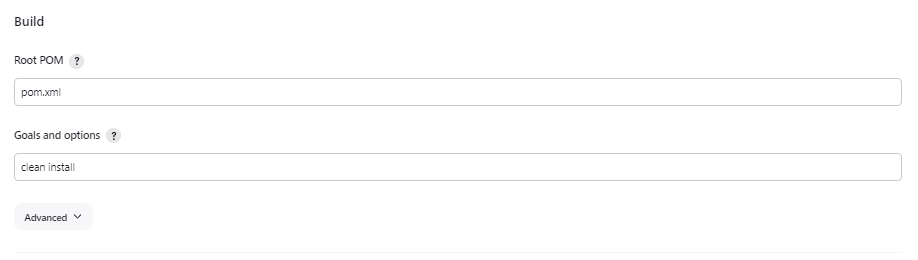
**Deploy to a Container**

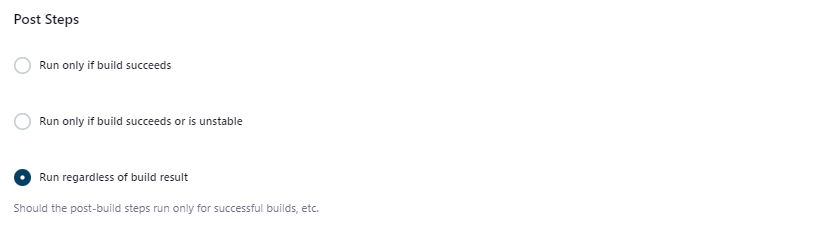


* **Pull the Code from Github**
* **Build using Maven**
* **Upload the Artifacts to Dockerhost**
* **Copying inside the container**

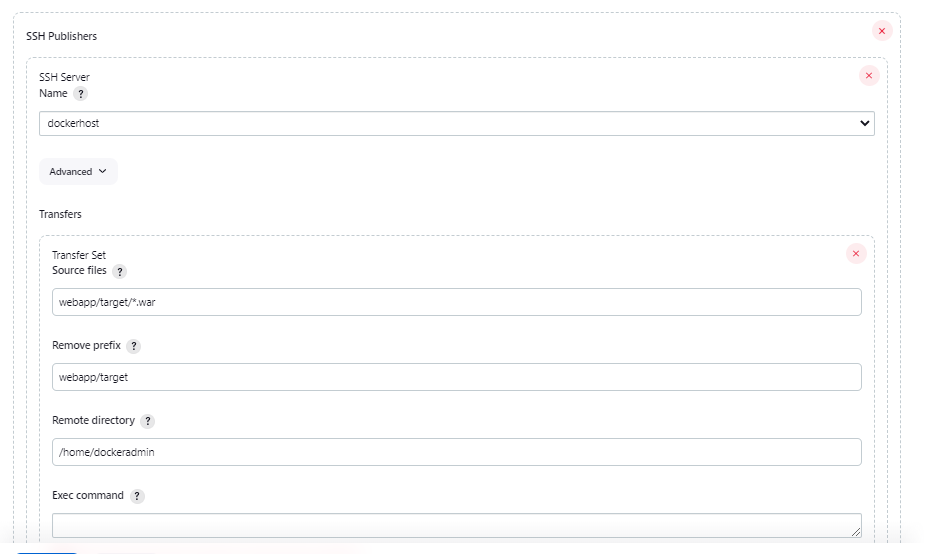




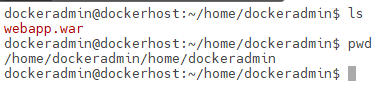




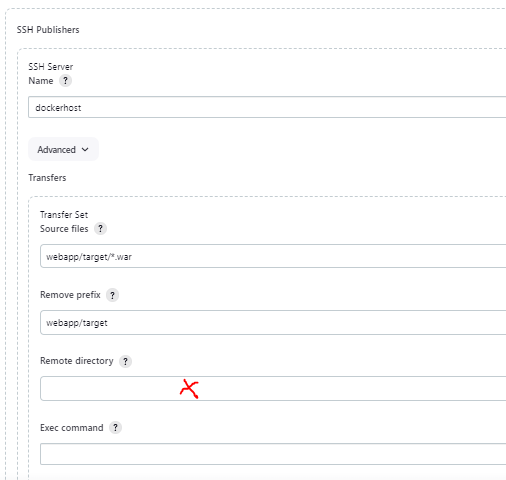
**This will only show after we install plugin**

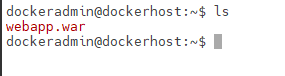


**Login to Docker Server and Check if the Artifacts are Copied**



**Remove home Path in Jenkins Configuration – As it is reprinting same path**





**Create a Dockerfile to Copy the Artifacts from local dockerhost to Container.**

* Login to Dockerhost - Lets create Docker folder in /opt and set permissions for dockeradmin

cd /opt

mkdir docker

chown -R dockeradmin:dockeradmin docker

**Move the Dockerfile and Give Permissions to dockeradmin**

mv Dockerfile /opt/docker/

cd /opt/docker/

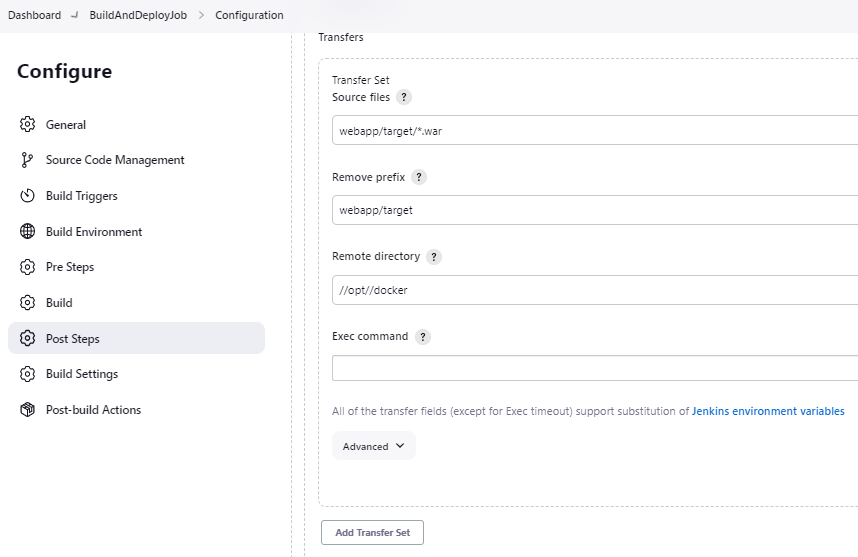
ls -ld

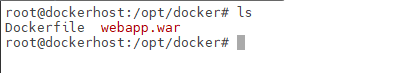
ls -l

chown -R dockeradmin:dockeradmin /opt/docker

ls -al

**Update the Path – Storing of war file in the /opt/docker**





**Upload the War file to container and run it in the container**

* **Update the Dockerfile**

**Original File**

FROM tomcat:latest

RUN cp -R /usr/local/tomcat/webapps.dist/\* /usr/local/tomcat/webapps

**Updated**

FROM tomcat:latest

RUN cp -R /usr/local/tomcat/webapps.dist/\* /usr/local/tomcat/webapps

COPY ./\*.war /usr/local/tomcat/webapps

**Build the Docker Image**

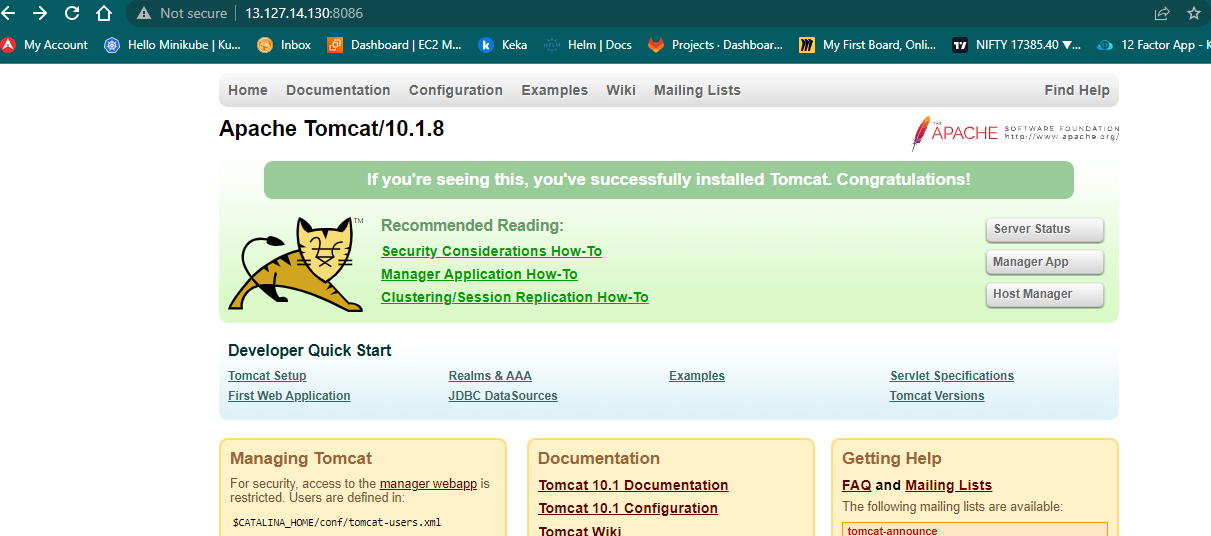
docker build -t tomcat:v1



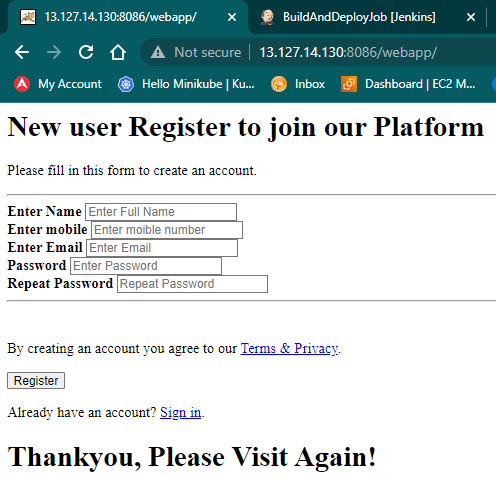
docker run -d --name tomcatv1 -p 8086:8080 tomcat:v1



**Access in the Browser**



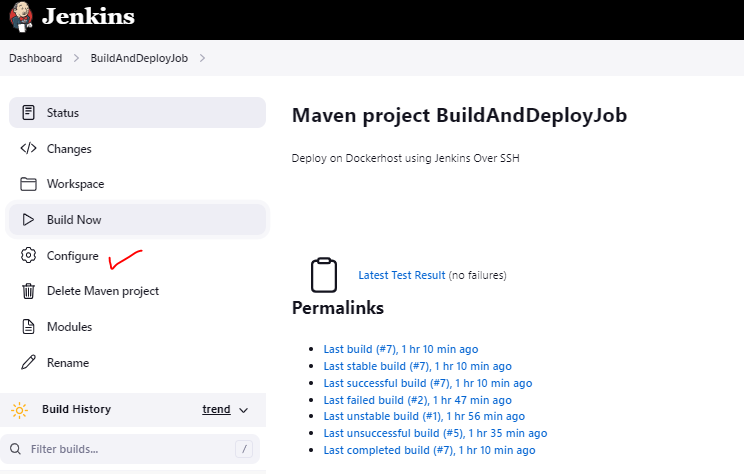
**Access the Application**



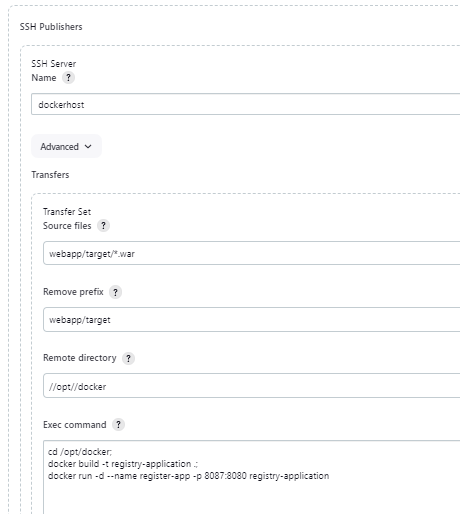
Now, this is manually created by logging into the docker.

**Automate the Deployment Process through Jenkins**

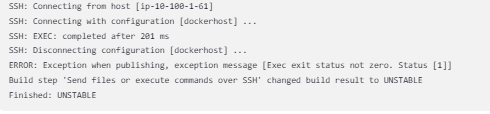
Once the code is copied – Jenkins Job itself should build an image and Create a container



Update the configuration



**SSH Issue – User dockeradmin – lacks permission of root**



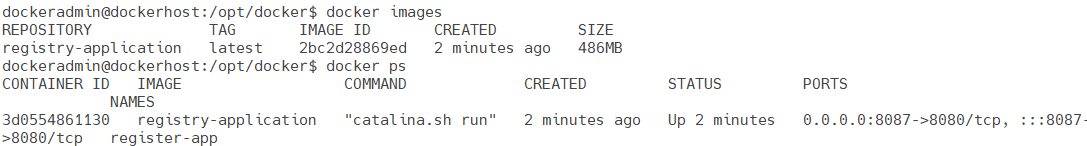
**Add user to sudoer group**

usermod -aG sudo dockeradmin

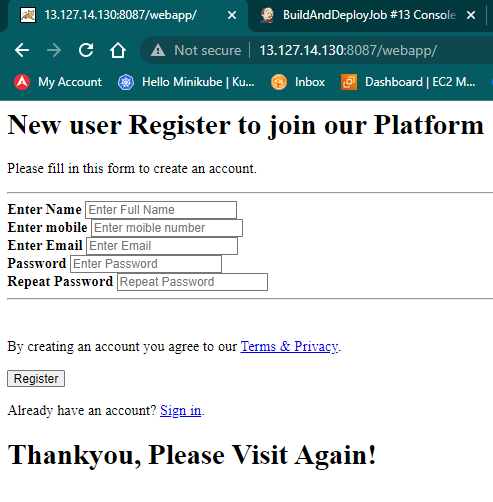
**Check if artifacts are pushed to dockerhost by Jenkins**



**Check if Image is created – Container is running**



**Check if Application Accessible**



What if a container is running and another container is getting created with the same name

**This can be managed by Ansible**

