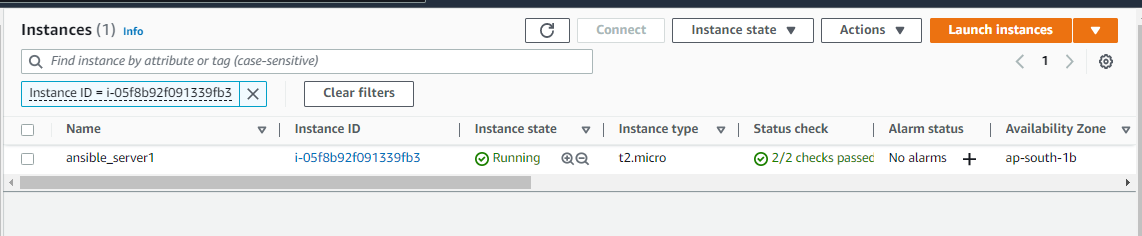
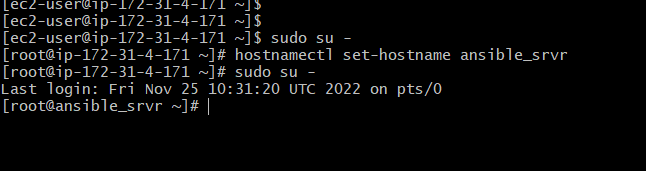
**Deploying on Docker container using Ansible Playbook and CI/CD job:**

1. **Creating set up for ansible server:**

Launch one more instance from aws account to create our ansible server.

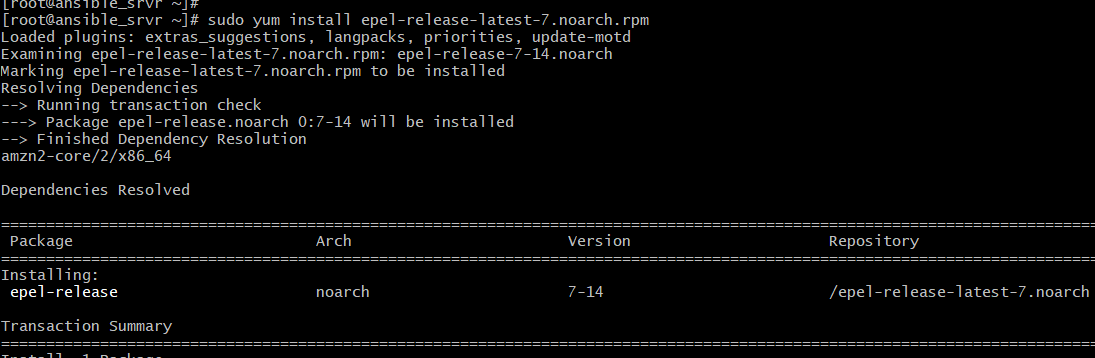


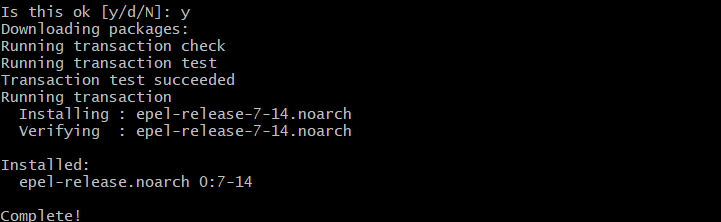
After installation take ssh in GIT Bash of ansible server login with root user change the hostname and again login with root user as shown in screenshots



Use following command to install epl release

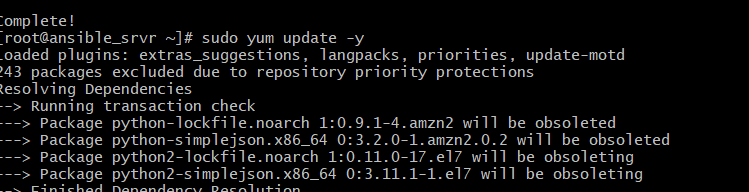
# sudo yum install epel-release-latest-7.noarch.rpm



Type y to complete the installation

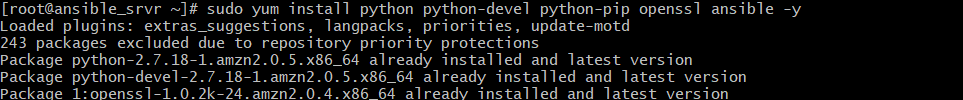
After that type following command to update our packages

**# sudo yum update -y**



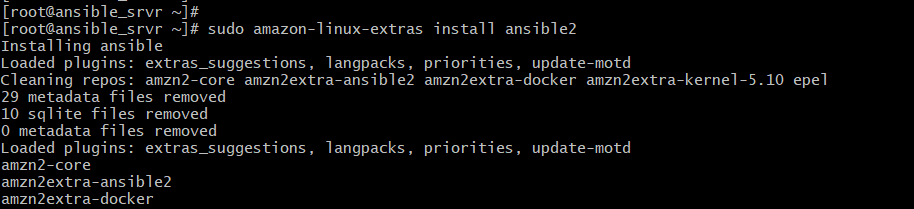
It will update the package and our yup repo is ready to install ansible.Run the following command

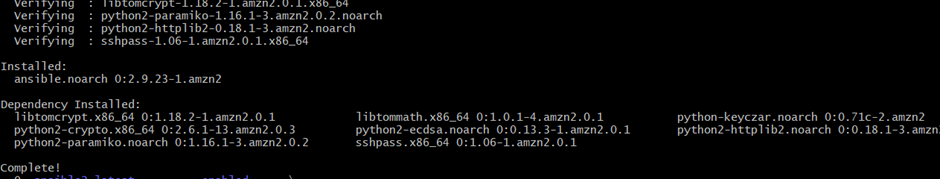
**# sudo yum install python python-dlevel python-pip openssl ansible -y**



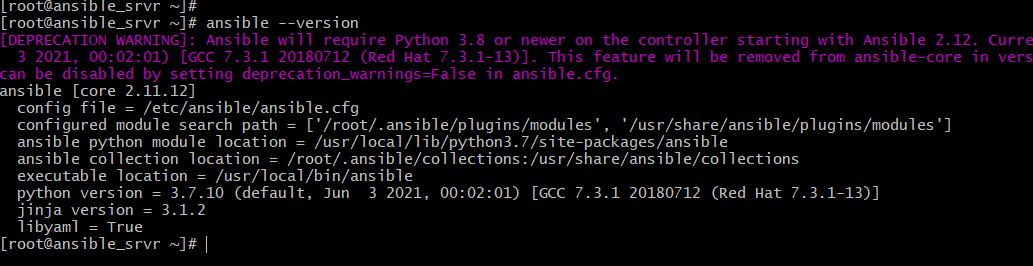
It will install all the packages. After that use following command to install ansible

**# sudo amazo-linux-extras install ansible2**

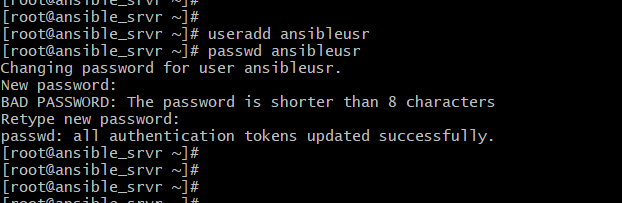




After installation check the version of ansible. Ansible is successfully installed.

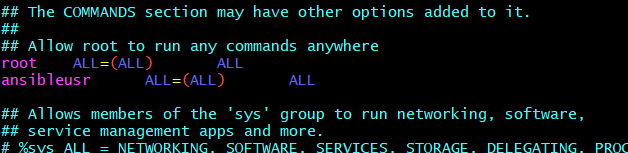


After successful installation of ansible login to root user ad ansible user and give its password

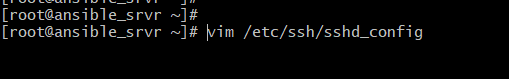


After adding the user successfully go to **/etc/sudoers** file to give all permission for ansible user as shown in below screenshots change edit the file as shown in the screenshot and save

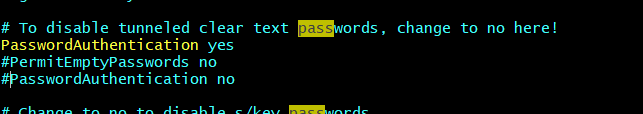




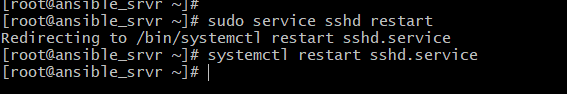
File we need to edit and that is **/etc/ssh/sshd\_config**



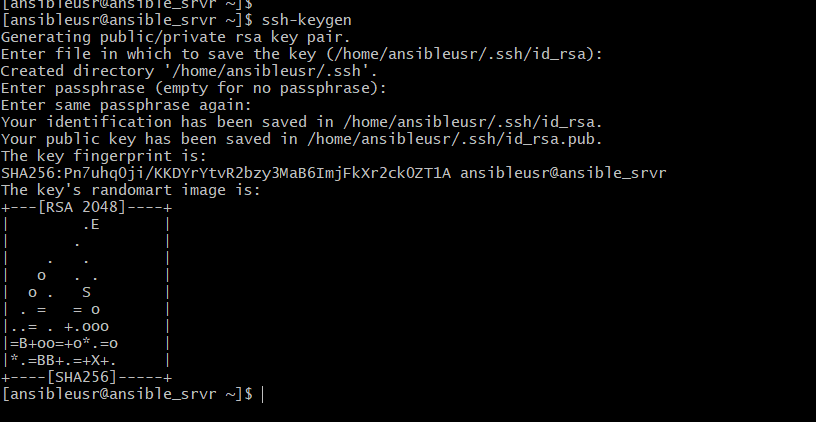
give **PasswordAuthentication** yes as shown in screenshot save the file



After that **reload** **ssh services** as shown in screenshot and **restart** the **sshd services**

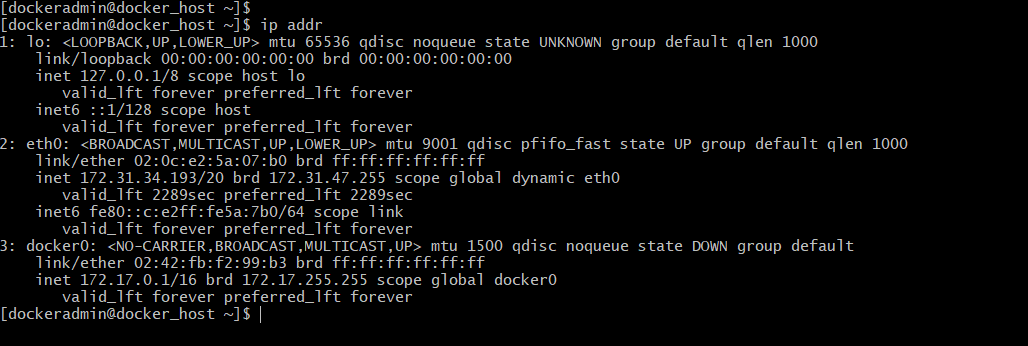


Generator ssh key on ansible server by using **ssh-keygen** enter 3 times it will create ssh key as shown in screenshot

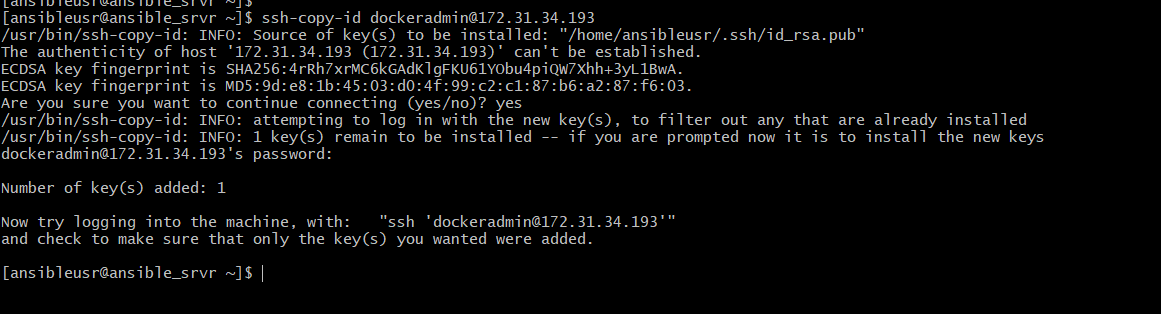


1. **Integration Docker host with Ansible server:**

Then go to the docker host which we want to integrate with ansible server login in as docker admin check it its IP address as soon in screenshot copy that IP address

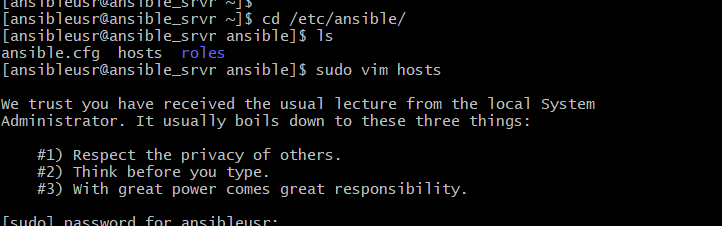


Then go to the ansible server where we created ssh key to copy ssh ID to address of docker host as shown in below screenshot



We can see that ID is created successfully now we can access docker host from ansible server near our docker host is successful integrated with ansible server

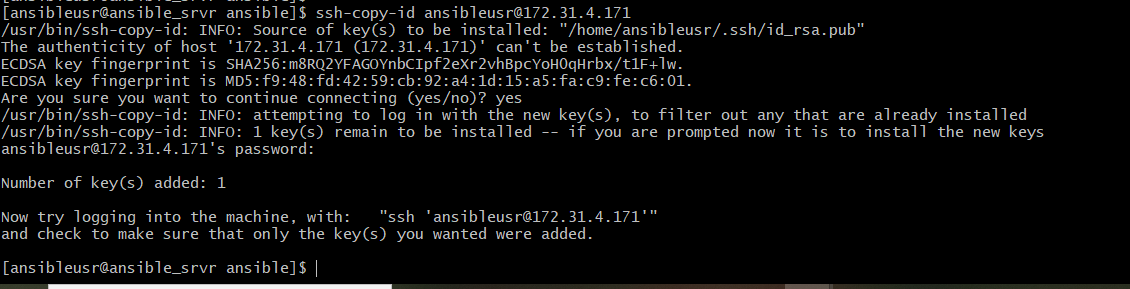
Now go to the directory where ansible is install as shown in figure



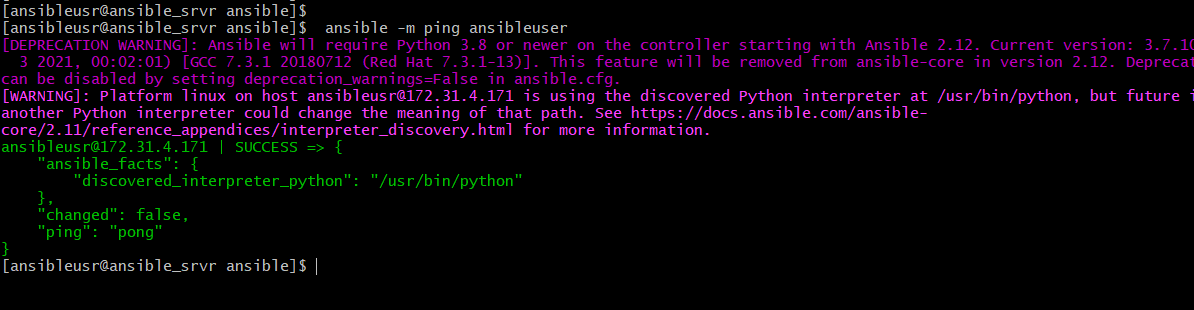
First, we will check power connection with doc ansible user so we are giving type address of ansible user as shown in screenshot and save the file



Before that first copy the access ID to the ansible user as we have created before as shown in screenshot

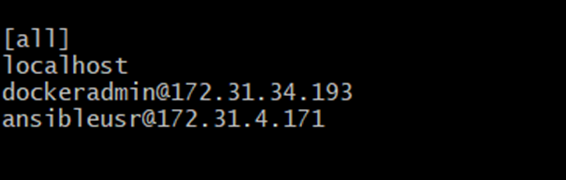


Then use ping model to connect with ansible user as shown in screenshot

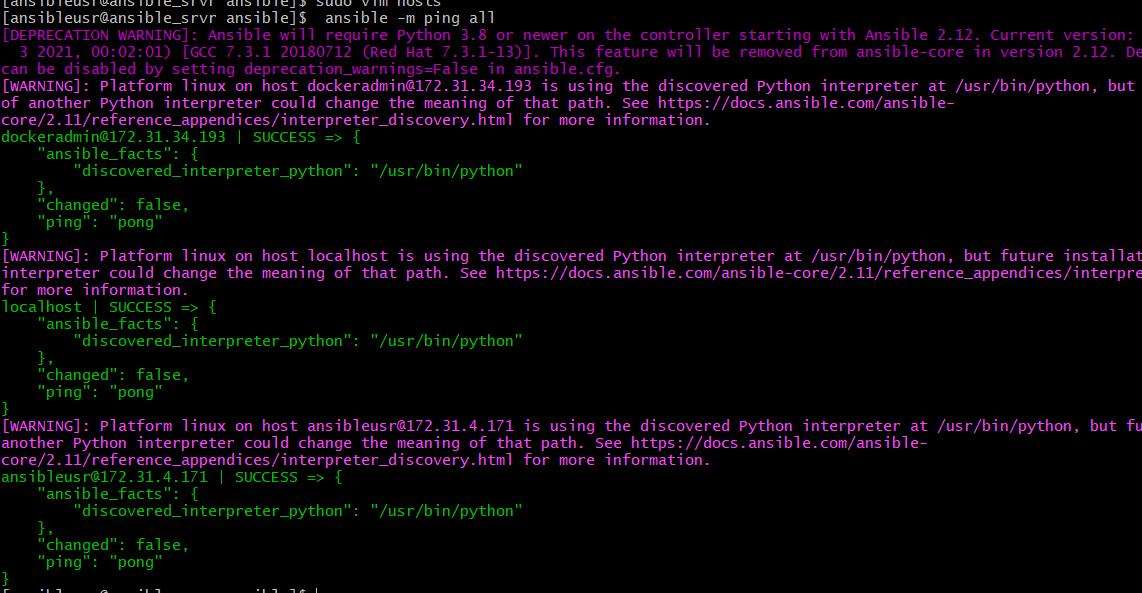


Our ansible user is get success fully connected with ansible now will check the connection for localhost and docker host.

Before that we will again edit our host file go to the host file and edit as shown in screenshot

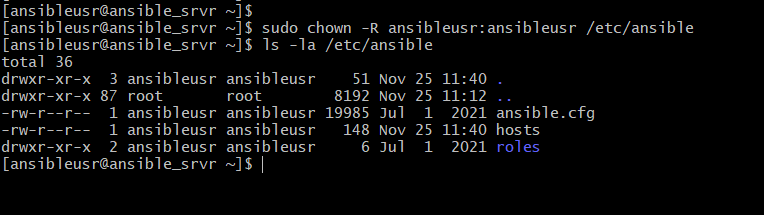


Again, use module ping to check the connection with all as shown in screenshot which is successfully connected

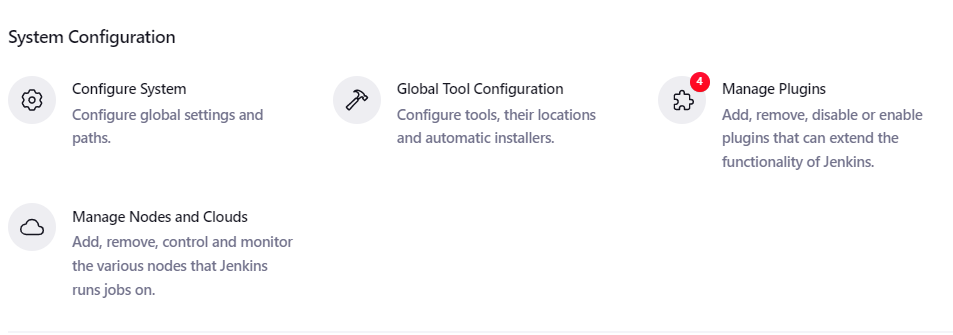


1. **Integrating Jenkins with ansible server:**

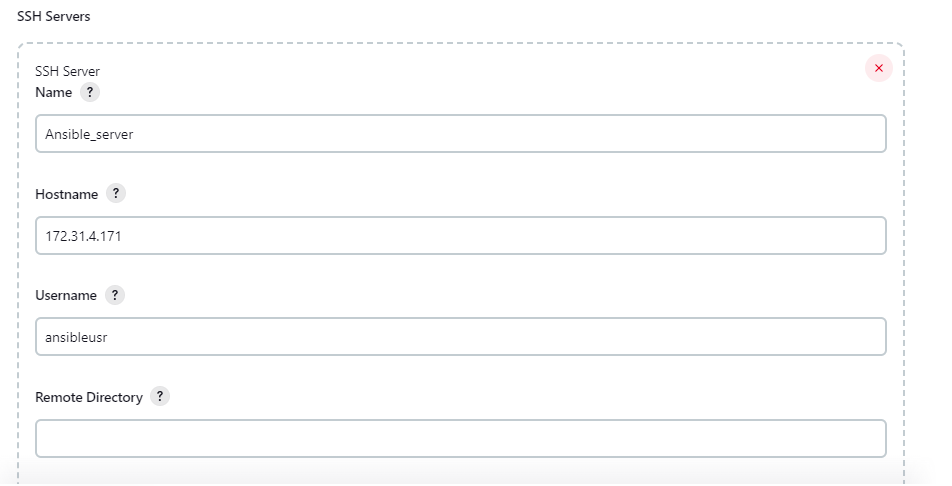
Now we will change the ownership of the the ansible directory so that we can access it by ansible user so we are using **chown** command to give access to the ansible user as shown in screenshot



After changing ownership go to the Jenkins Browser login to your Jenkins go to the manage Jenkins select configure system option as shown in screenshot

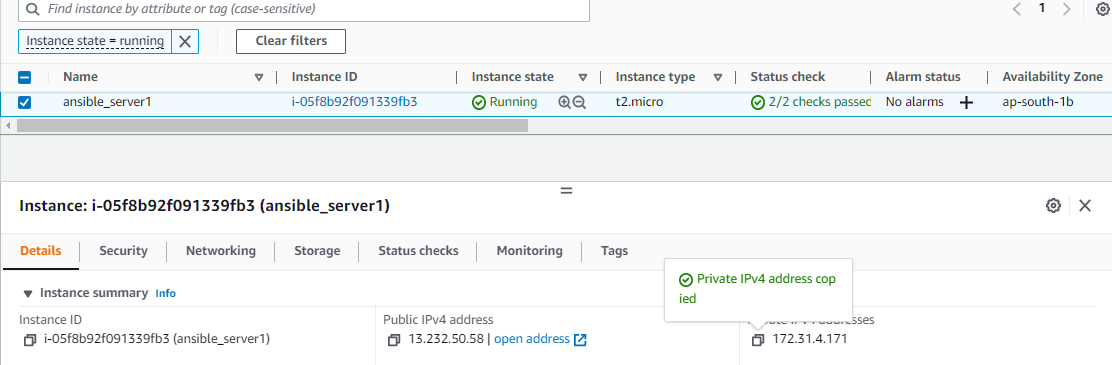


Then scroll down below in ssh server section provide the name of ssh server host address of ssh server that will be private IP address of ansible server then provide username as shown in screenshot

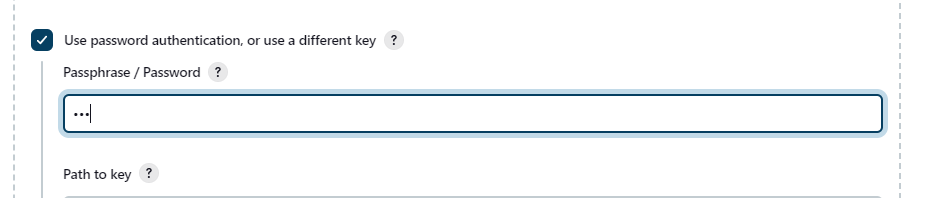


Then scroll down below choose use password authentication for or different key in that provide the password of ansible user

To copy private IP of ansible server go to a WS copy the private IP of ansible server as shown in screenshot and paste it on hostname



Provide the password of ansible user that we have created on ansible server



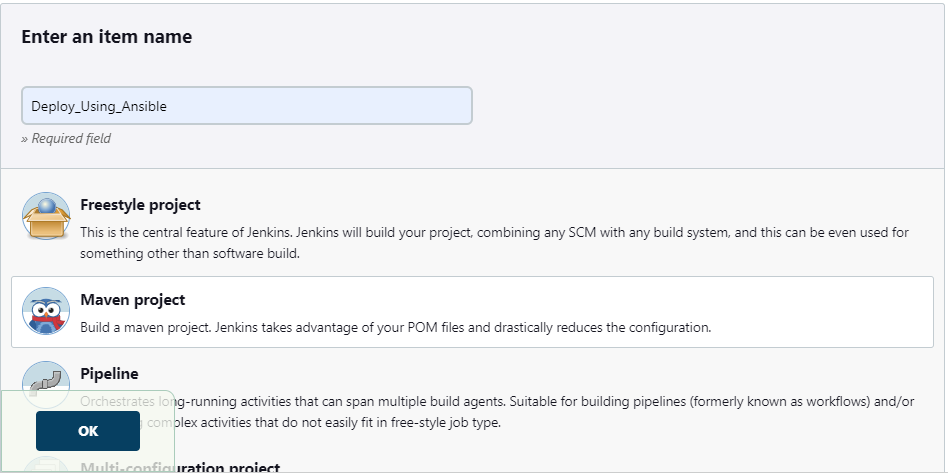
Then scroll down and test the configuration if it is get success it will show message as shown in screenshot



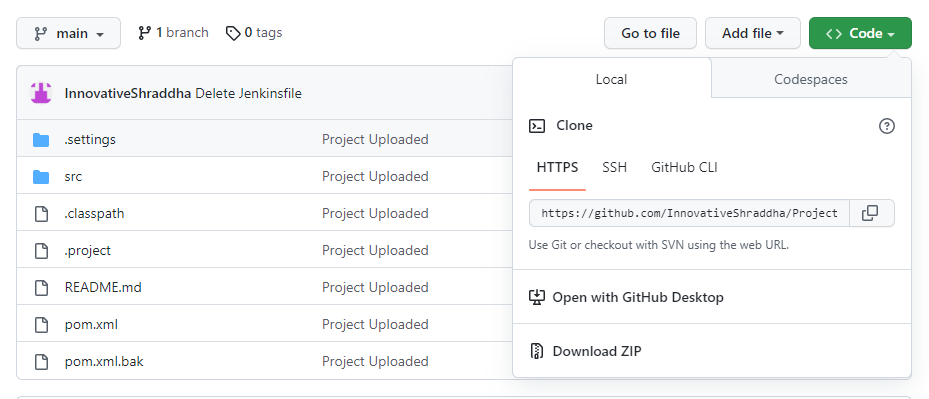
Now ansible server gates successfully integrated with Jenkins

1. **Copy .war file on ansible server using CI/CD job of Jenkins:**

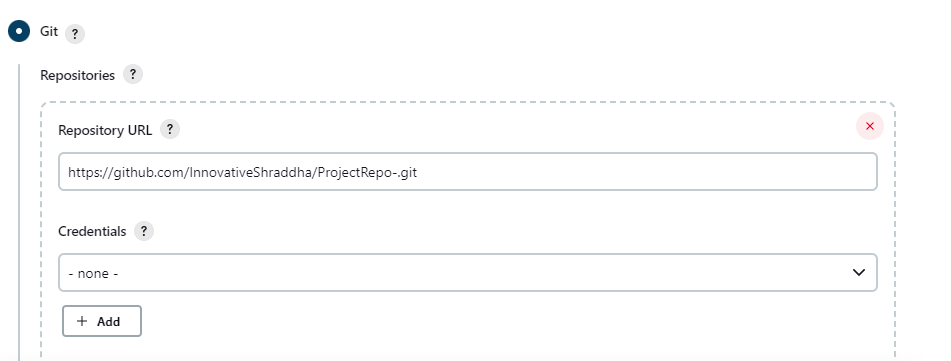
I have created a CI CD job for copying our war file on ansible server so create a job as shown in screen shot click on ok button



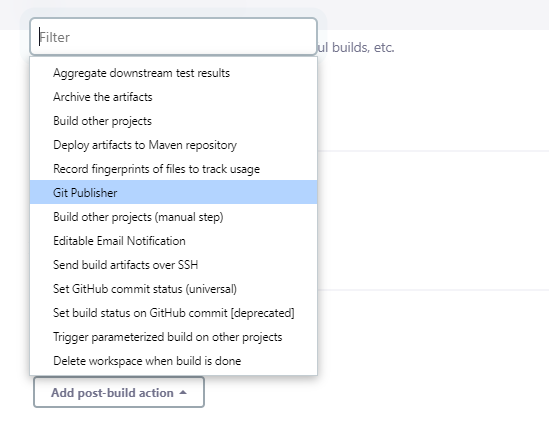
To give the path of GIT repository first copy it from account a shown in screenshot



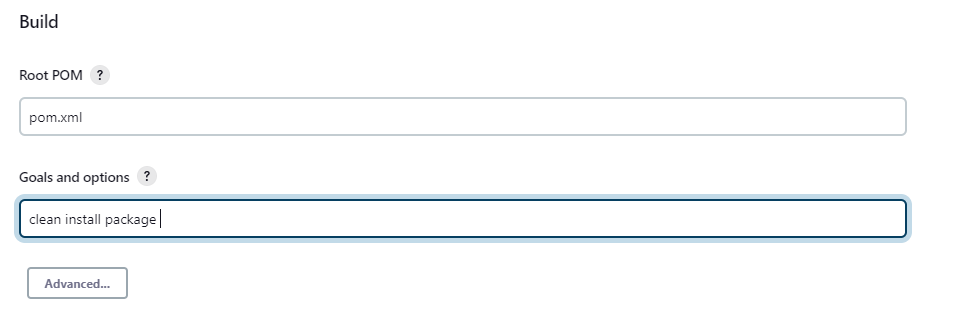
Then go to Jenkins job go to configure section in the source code management paste the path of Git repository that we have taken from GitHub as shown in screenshot



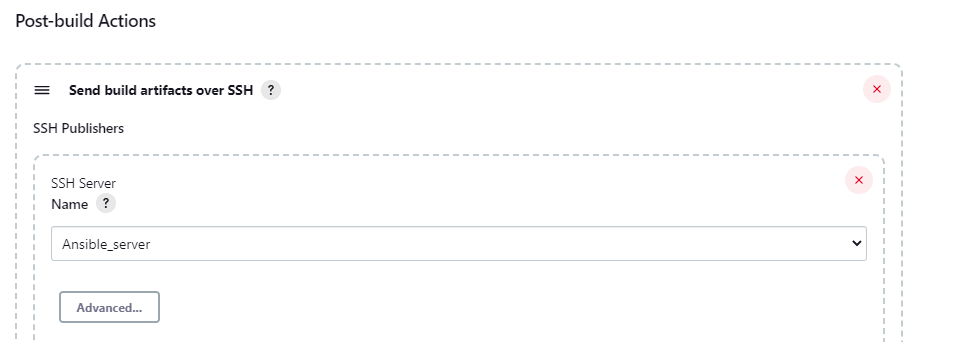


Then in the build section give the Maven command as shown in screenshot Then in the build section give the Maven command as shown in screenshot

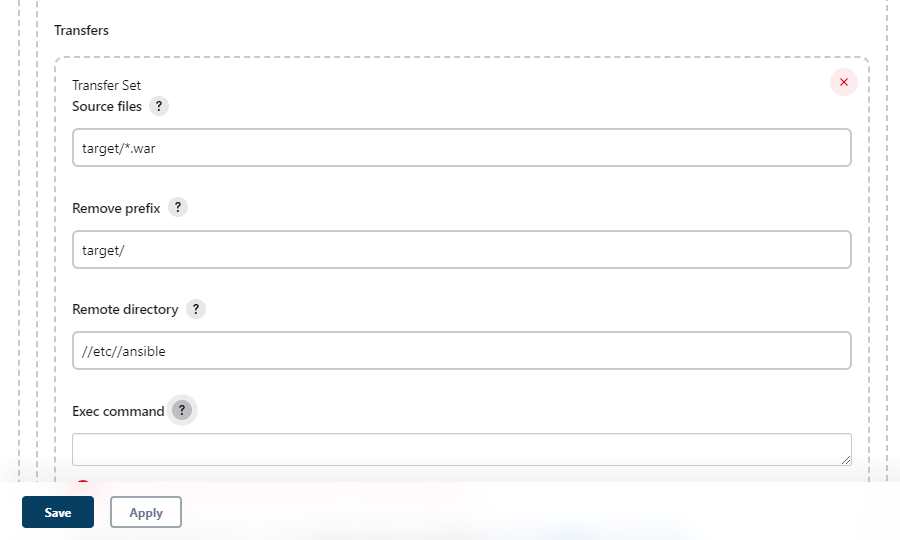


Then provide the name of ansible server selected from drop down ashen is screenshot

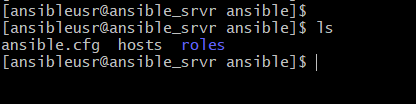


Provide the part of our war file and the remote directory there we want to copy our war file as shown in screenshot

Click on **save** button after configuration



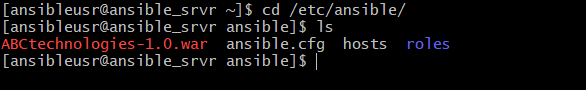
Before building the job check whether **.war** file is present or not as shown in figure



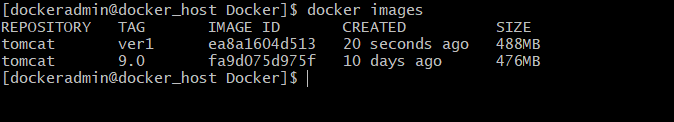
In above screenshot we can see that our war file is not copied it so we will go to our Jenkins server and build the job. I will give following output if our job gets built successfully



Then go to ansible server and check whether our file is copied or not in the below screenshot we can see our file gets copy successfully



Then go to your **Docker** host the image is created is from the previous task



**5) Pushing image to Docker Hub registry:**

There are two ways where we can deploy war file on docker container

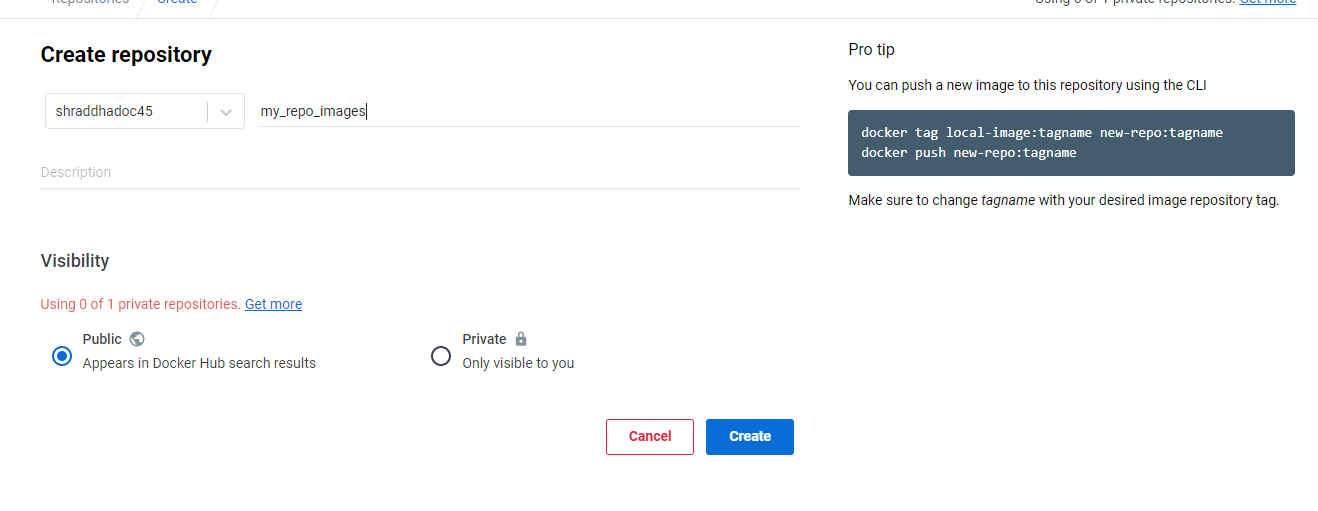
1. We will tag the build image and push it on docker hub registry

And write a Playbook to pull the image from docker hub registry

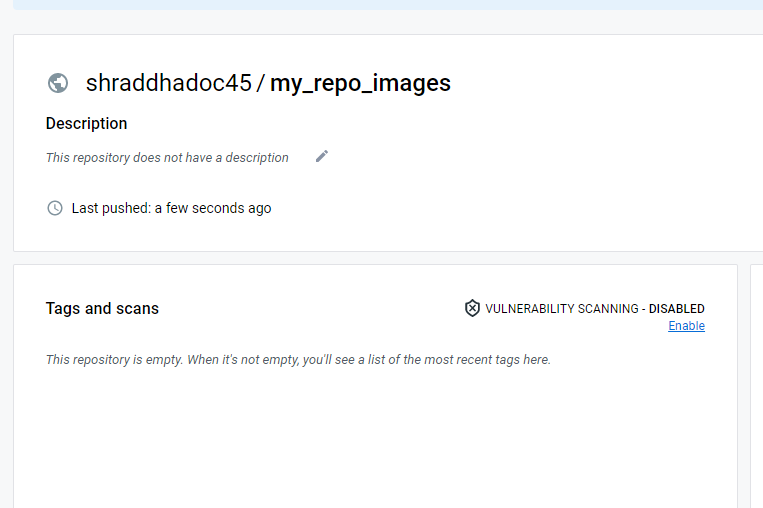
1. We will write a Playbook directly to build image as well as will create a container out of that image.

**Push image to docker hub and then create container out of it**

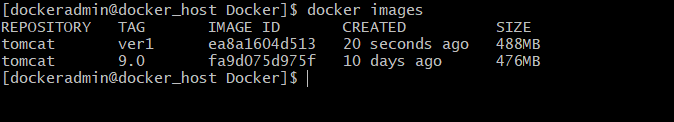
For that login to your docker hub registry after that create a new repository that will stover h images which we have push from docker server as shown in screenshot Give the name of repository and click on create button



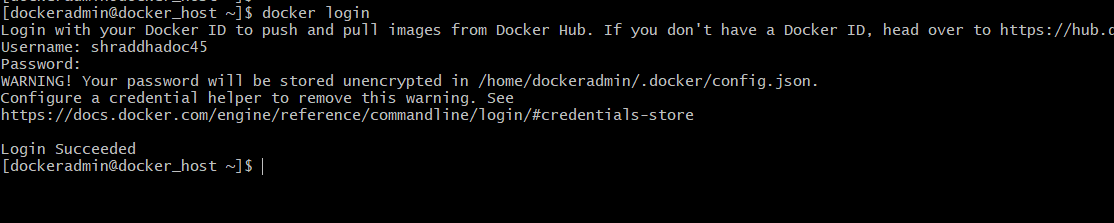
I have created a new repository which has currently no image is push as shown in screenshot



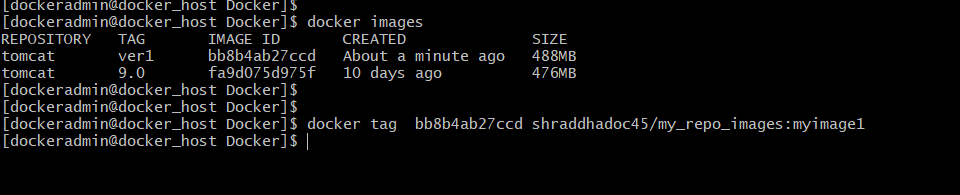
Check that is any current image present for pushing image to docker registry



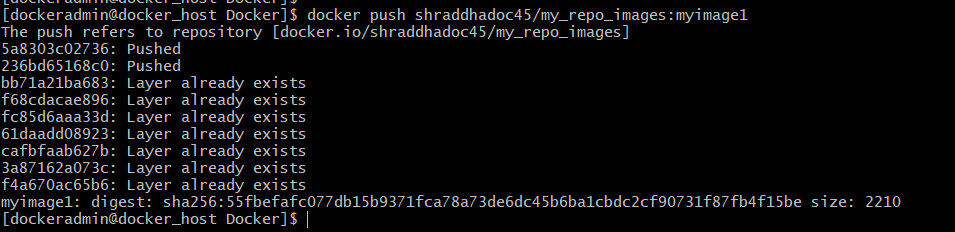
So here tomcat image is present with **version 1** tag that created in the **task 3** then and login in to docker registry from docker host serial I as shown in screenshot and provide username password that are used for login into docker hub registry



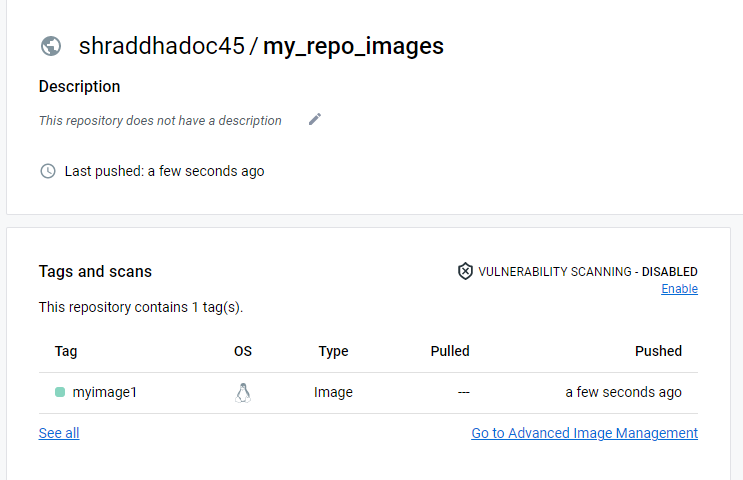
Now we will tag the image to push it on docker hub registry as shown in screenshot



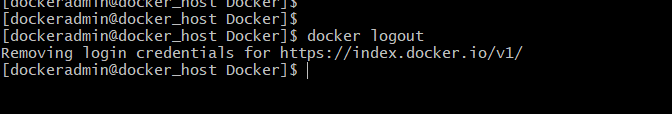
Now we will use docker push command to push image that tag before as shown in screenshot



I It is successfully push tom cat image to docker hub registry now we will check whether our image get push or not go to your docker hub repository and check it is successfully push tomcat image

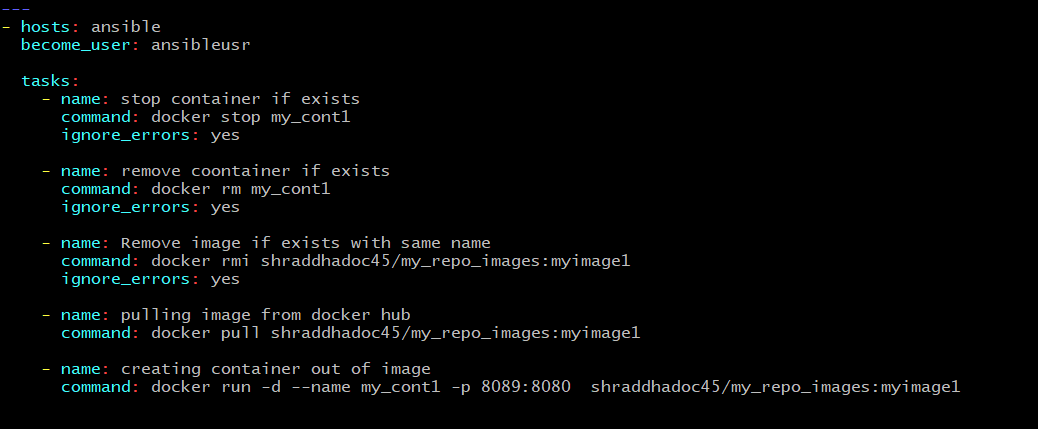


After pushing the image, we will log out from docker registry using the core host CLI as shown in screenshot.

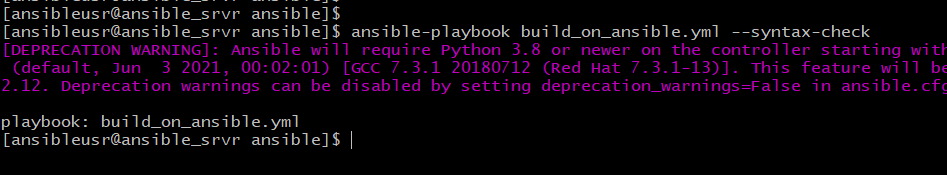


**Push image to Docker Hub using playbook**

Write a Playbook to push docker image to docker hub registry as shown in screenshot



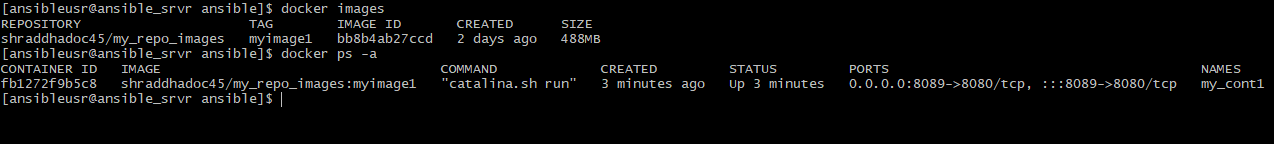
Save the file and check the syntax of the Playbook by using following command as shown in screenshot



When we run the playbook it will delete any image and container present with the same name and if not it will create successfully.



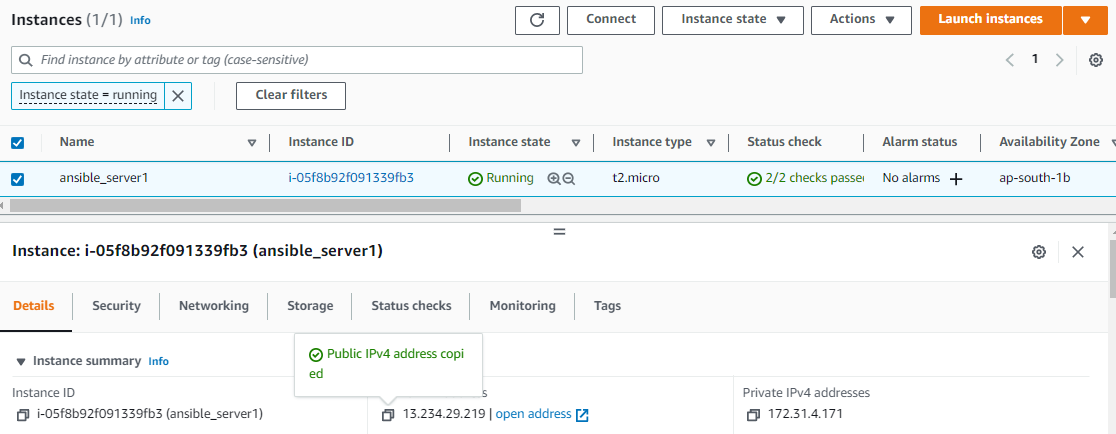
In the following screenshot we can see that image in which get pull successfully from docker hub registry and created one container out of it according to Playbook.



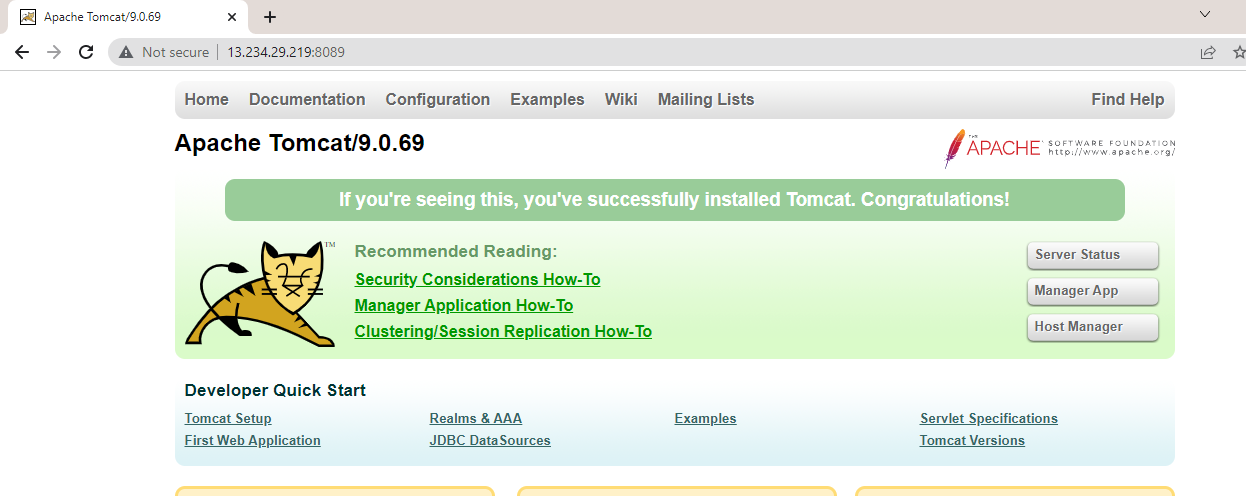
So this is a first scenario where we can to pull the image form Docker hub registry and create a container on it

If you want to check that if we want to check deployment our application is successful or not

Go to ec2 dashboard from that click the public copy the public IP ansible server as shown in following screenshot



And paste it in the browser with the port number 8089 it will show tomcat server page as shown in screenshot



Now if you want to access your application just type to your application name by using this URL

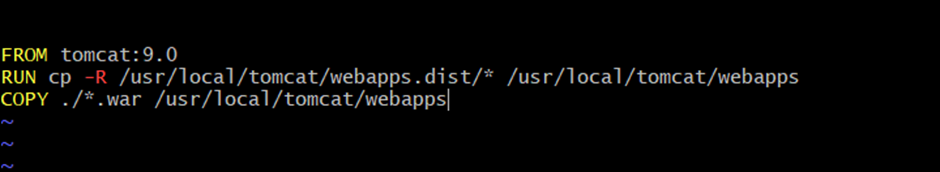
And we can see that our application deployed successfully as shown in screenshot

<http://13.234.29.219:8089/ABCtechnologies-1.0>

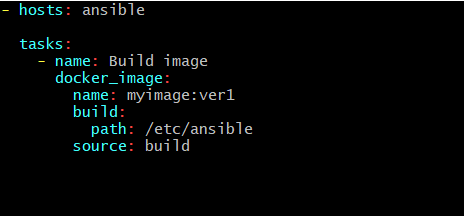


**6) Create an Container using DockerFile on Ansible Server:**

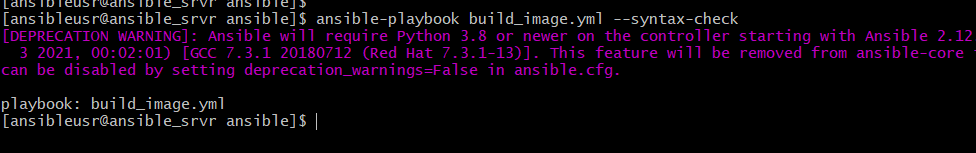
Now the second method is deploy on docker host using ansible server for that we need to tu first build image from dockerfile so first write dockerfile to create and image



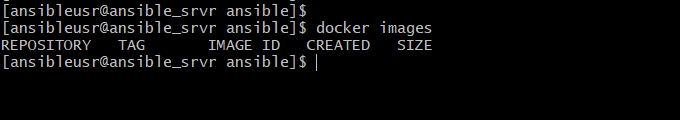
Save dockerfile and write a play book to build image from dockerfile as shown in the following screenshot



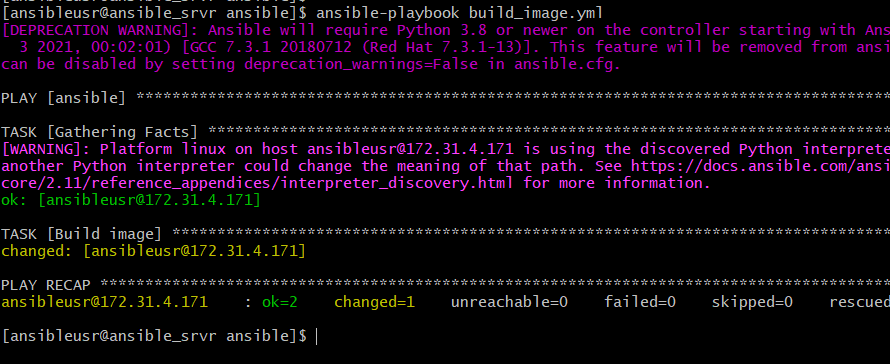
Now check the syntax of ansible Playbook as shown in screenshot



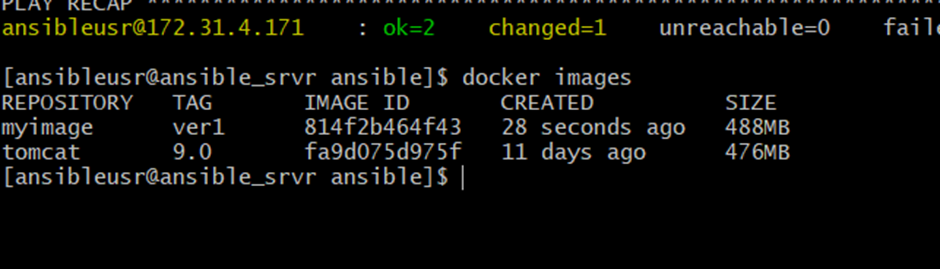
Before I run to our Playbook I have checked that is there any docker images created on my machine as shown in screenshot



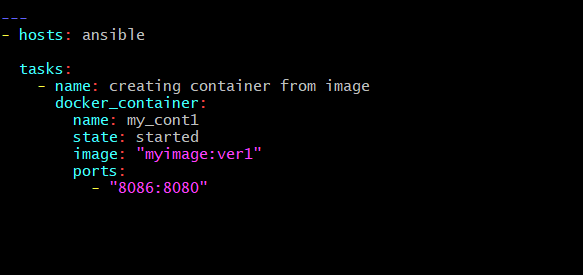
So there is no image created so we can run our Playbook and it will created successfully as shown in figure



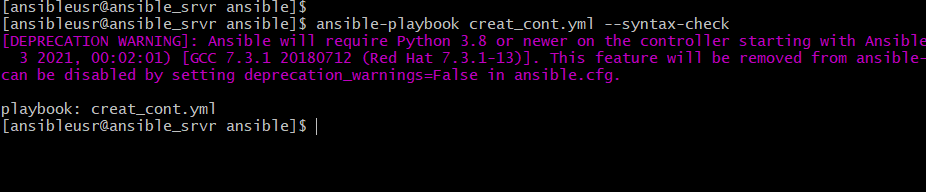
We can see that our image get created successfully from dockerfile



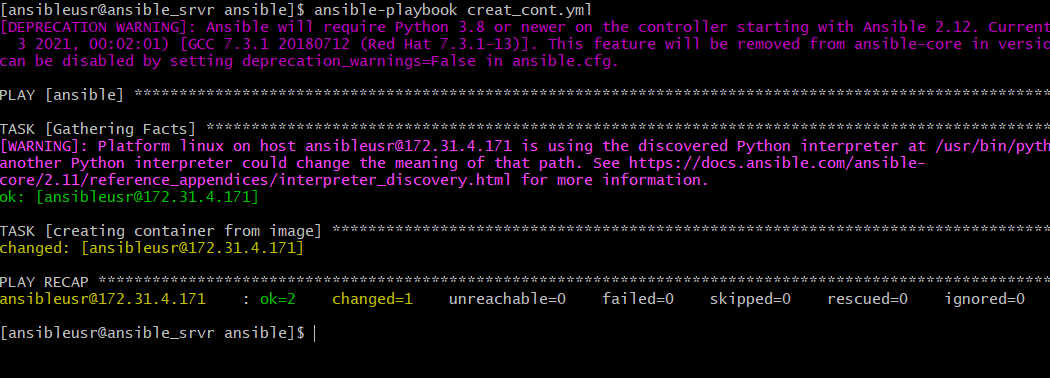
Now write a Playbook for creating container out of the image we created before as shown in figure



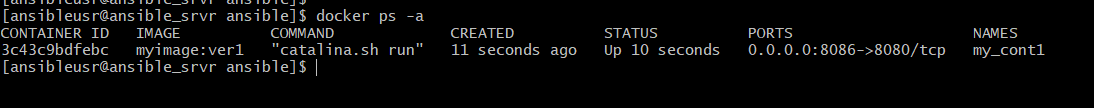
Save the file and check the syntax of Playbook as shown in figure



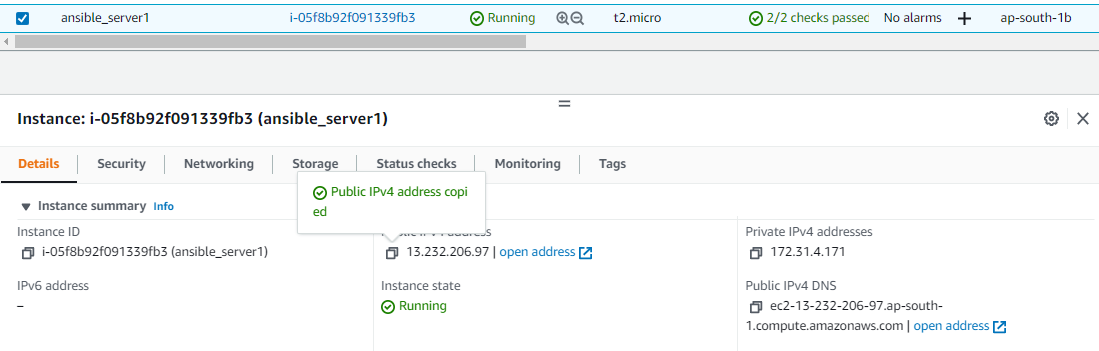
Run the Playbook it will create a container if the image is created before otherwise it will give error I have already created one image so it will create container out of that image



We can check that our container gets created or not by using docker PS command



A container is created successfully and application deployed on container to check that go to your a w ec2 dashboard copy the public IP from dashboard as shown in figure

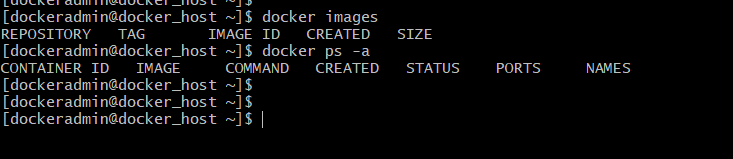


Paste the URL as shown in following figure along with port number 8086 and then give application name

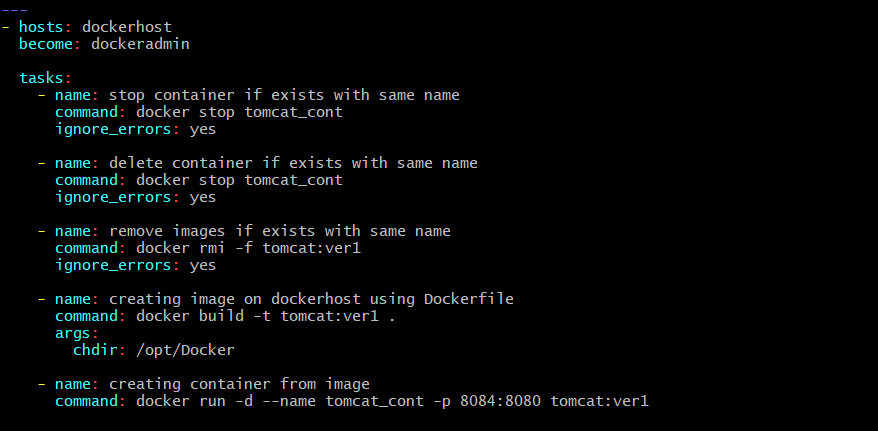
 **7) Deploying image on Docker container using Playbook:**

Up till now I have written Play Books for deploying our application on ansible server

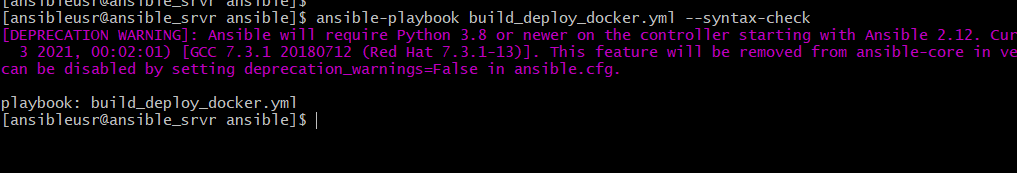
Now I will deploy on docker host using ansible Playbook for that go to docker host and check whether any image or container is present or not



Here we can see that there are no images and container created on docker host so so again go back to ansible server and write a play book as shown in figure



Save the file in **/etc/ansible** directory and check the syntax for playbook as shown in figure

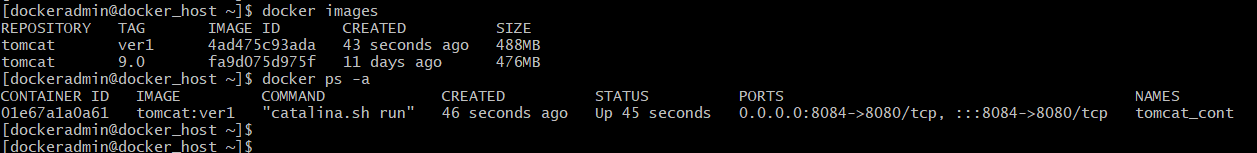


I have already configured the host file and given the private IP of docker host and check the connection before

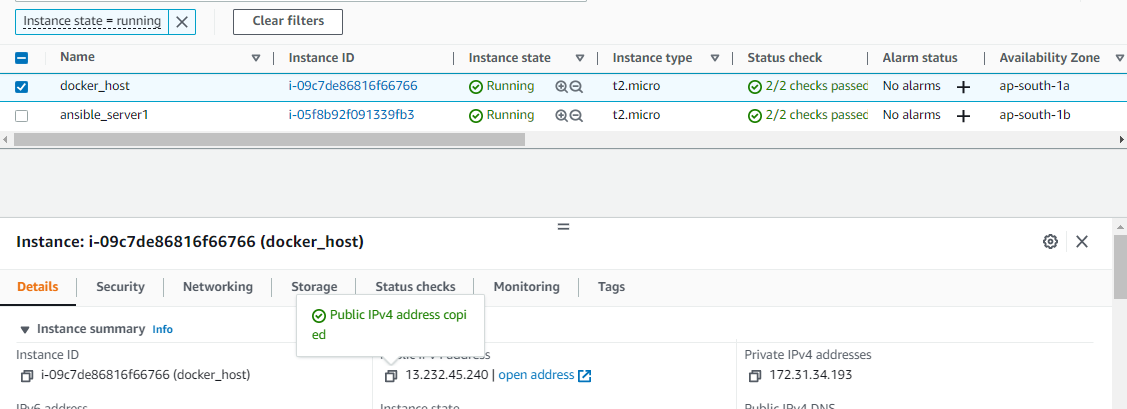
Now run the Playbook as shown in figure it will check whether the images for container created if they are present it will delete and create a new image and container on docker host into **/opt/Docker** directory



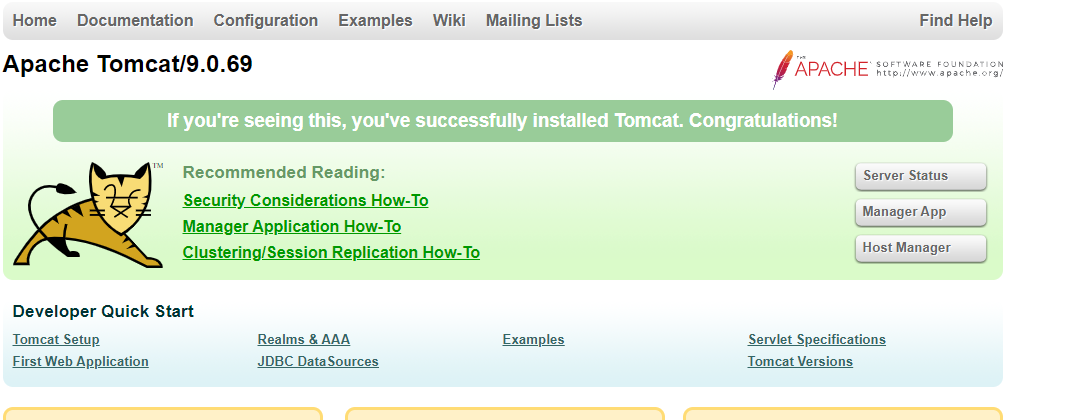
Now go to docker host and check image and container gets created as shown in figure



Now go to ec2 dashboard and copy the public IP of docker host as shown in figure

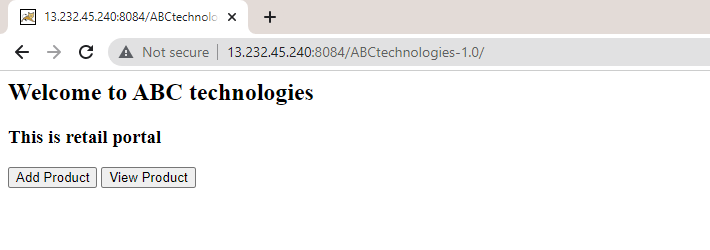


And paste it on URL with port number 8084 it will open Tomcat home page as shown in figure



Now if you want to access just give the the your application name with slash as shown in figure

<http://13.232.45.240:8084/ABCtechnologies-1.0/>



**7) CI/CD Job to deploy on Docker host:**

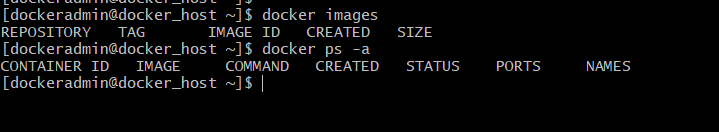
Now we want to write CI CD job to deploy application on docker host so so we will first delete our images and container using following commands on docker host

Now we want to write CI CD job to deploy application on docker host so so we will first delete our images and container using following commands on docker host

**docker stop $(docker ps -a -q)**

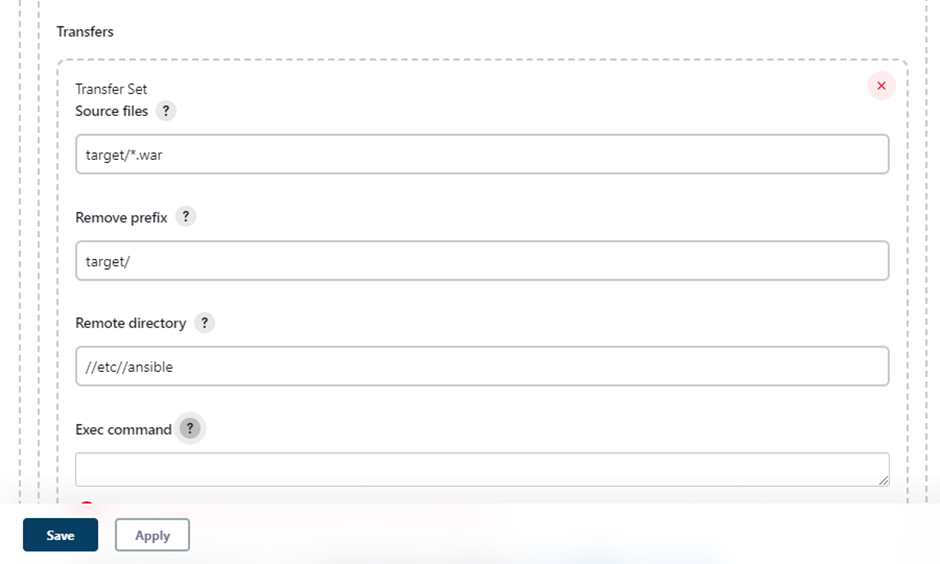
**docker rm $(docker ps -a -q)**

**docker rmi -f tomcat:ver1**

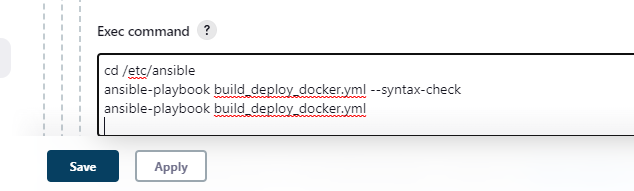


Now go to Jenkins server and configure the job we have created before for copying our war file on ansible server

In that job go to **configure section >> post build actions**



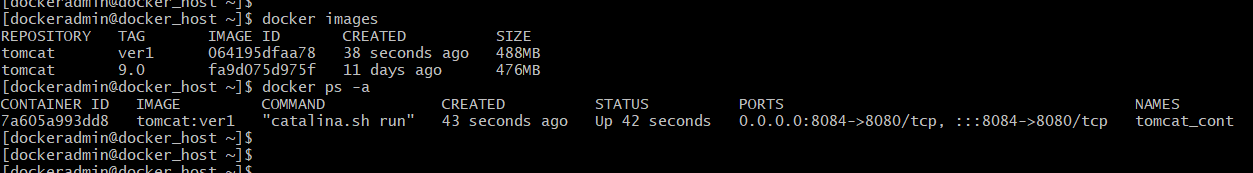
Give the command to run Playbook as shown in figure save the file



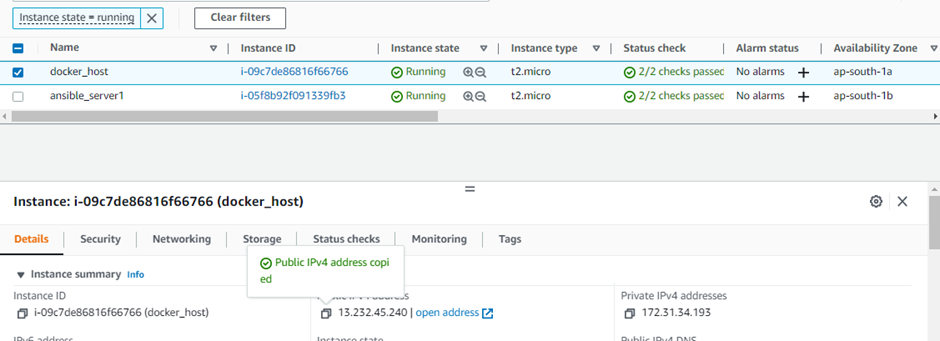
If you build the job it will give the following output as shown in figure it means that our build get successful



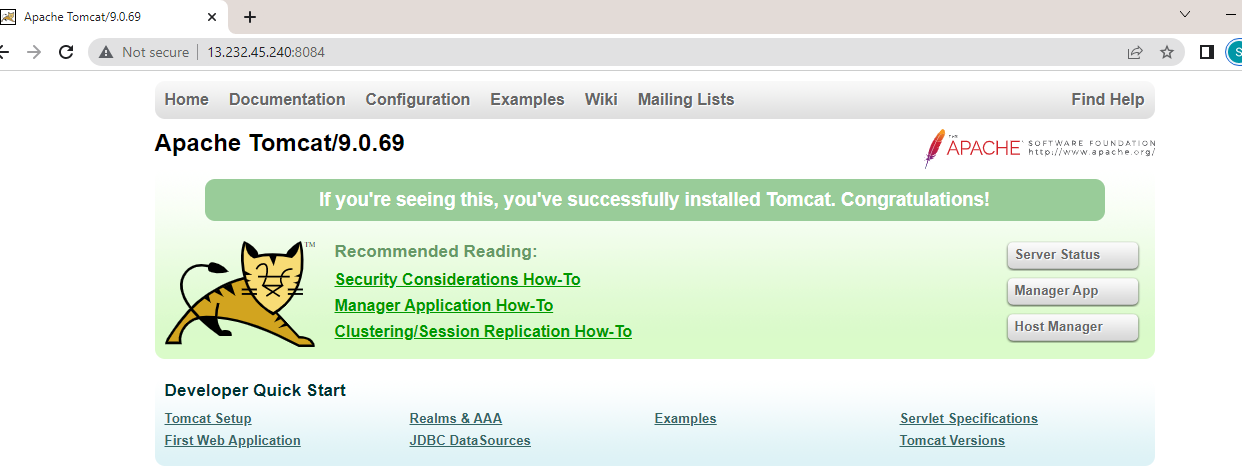
Now go to docker host and check whether image and container gets created as shown in figure



Now copy the public IP of docker host from ec2 dashboard as shown in figure

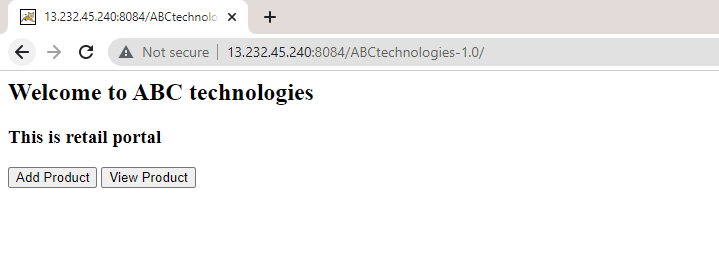


Paste the URL with port number 8084 shown in figure



Access your application using following URL

<http://13.232.45.240:8084/ABCtechnologies-1.0/>



We can see that application deployed on docker container successfully