**Python scripting**

**Using Python to Automate Docker**

**Install python3**

apt-get install python3 -y

python3 --version

**install docker**

curl -fsSL https://get.docker.com -o install-docker.sh

sudo sh install-docker.sh

docker --version

**Install python pip**

apt-get update -y

apt-get install python3-pip -y

pip3 --version

**Python program Interacts with docker**

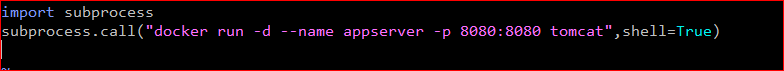
Require packager called ==> docker -py

pip install docker-py

**Use case1:**

**Create a tomcat server**

**Vim sample1.py**



If python want to interact with docker, then we have to use module called**🡺 subprocesses**

Subprocess module has to imported

**How do you import?**

**Import subprocess**

Once is subprocess is imported, we have a function called as🡪 **call** function

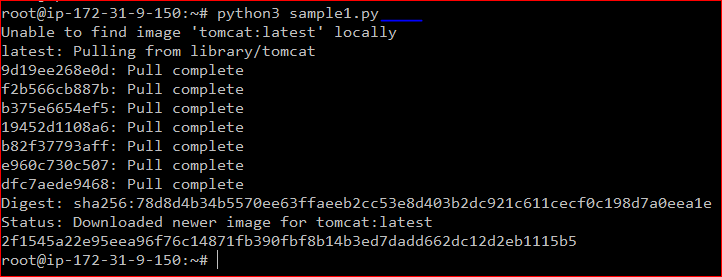
In this call function, we have to use any docker command:🡺

**Subprocess.call(“docker run -d –name appserver -p 8080:8080 tomcat”,shell=True)**

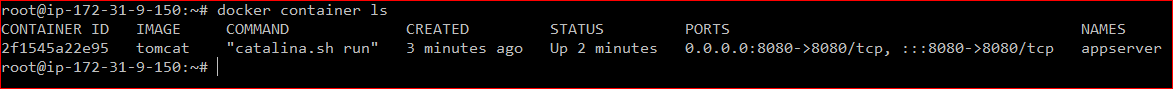
**-------------------------------------------------------------------------------------------------------------**

**Now run this python script**

**Python3 simple1.py**



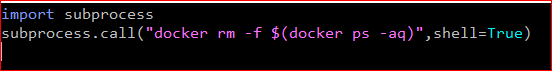
**Check the container**



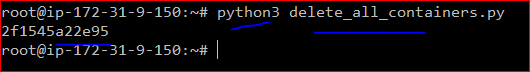
**Use case2:**

**Now we to delete all running and stopped container using python script**

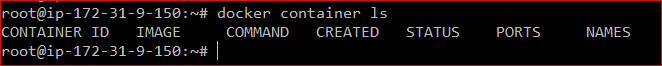
**Vim delete\_all\_containers.py**



**Run script**



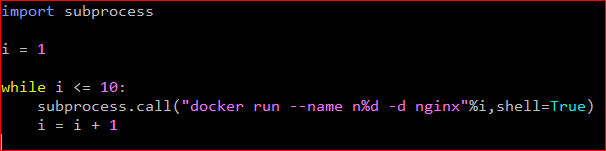
**Check the container**



**Use case3:**

**Python program create a 10 nginx container for performing load balancing**

**Vim simple2.py**



**If ‘ I’ is true**

**n%d 🡺 my program understands after ‘n’ there should be a numerical value ‘%d’**

**now give’ %i ‘**

**what ever value store in ‘I’**

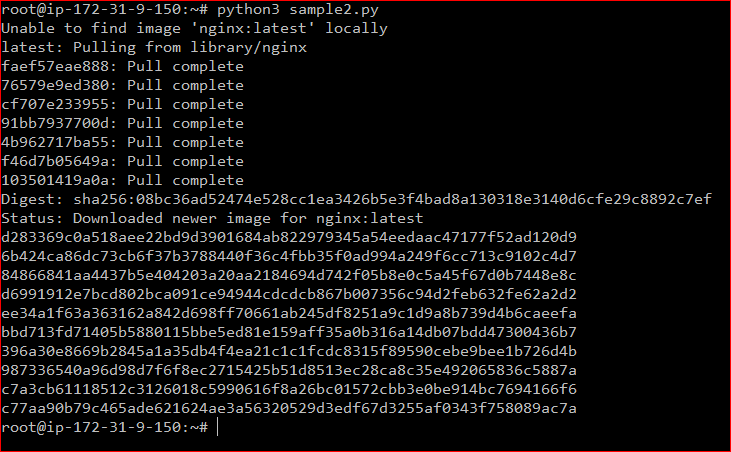
**value it will take and replace over ‘n%d’**

**if I valve is 1 , it will take ‘n%d’ is 1**

**--------------------------------------------------------------------------------------**

**now run the script**

**python3 sample2.py**



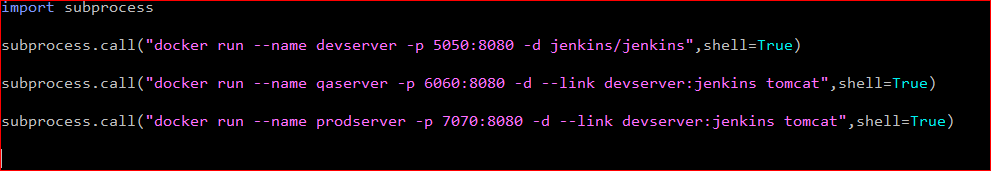
**Check the container**



**Use case4: alternative for docker compose**

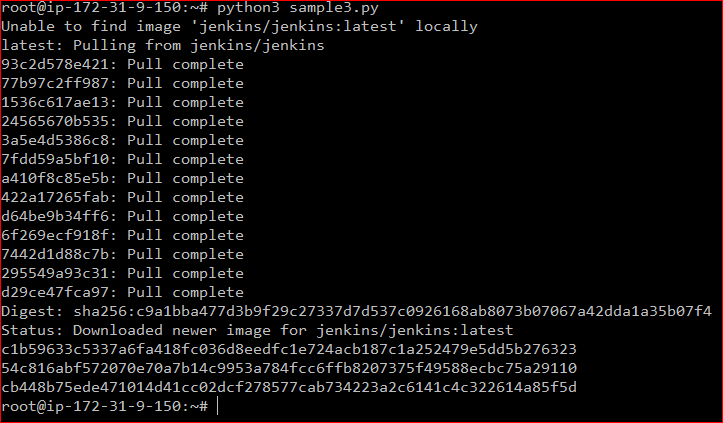
**Create a python program which will link Jenkins container with 2 tomcat containers, to implement CICD**

**Vim simple3.py**

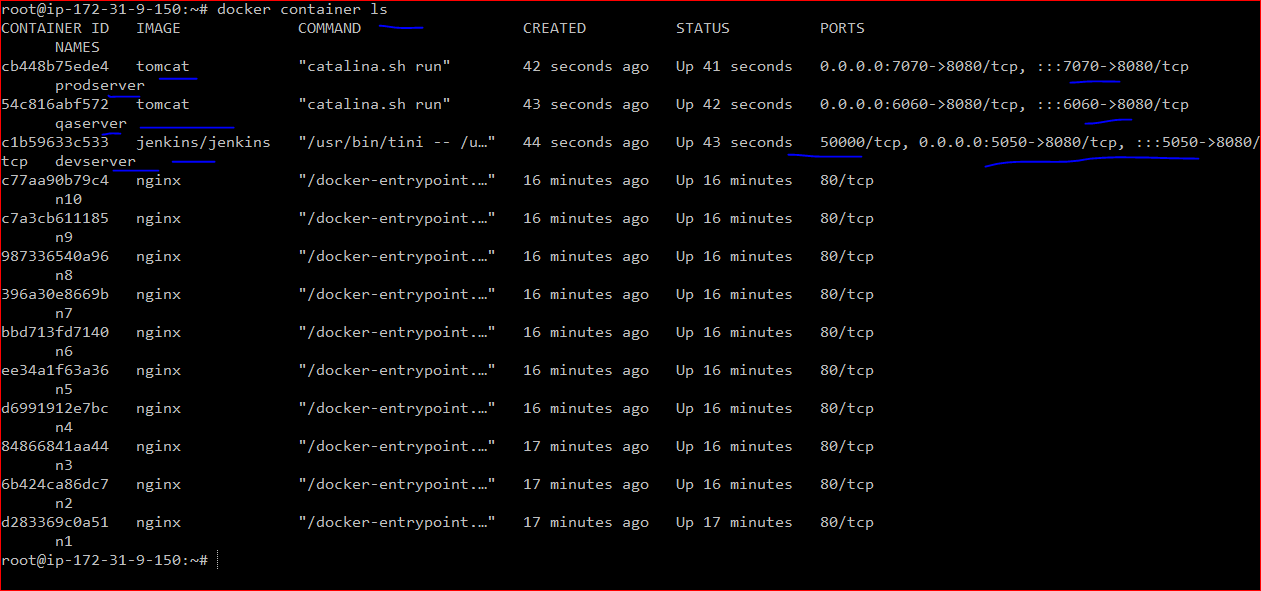


**now run the script**

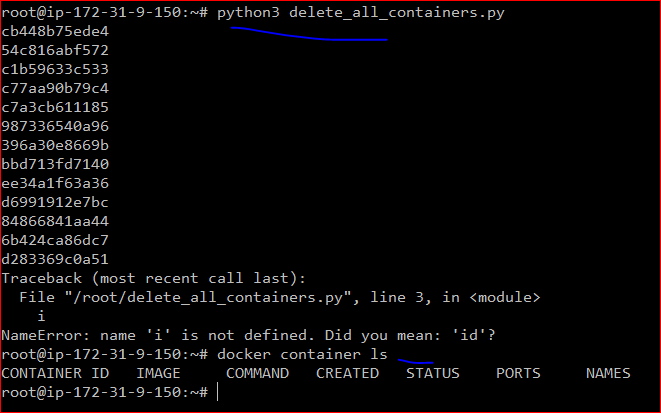
**python3 sample3.py**



**Check the container**

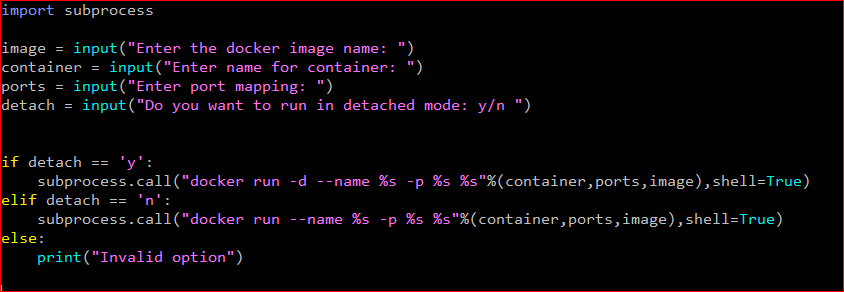


Now delete all container



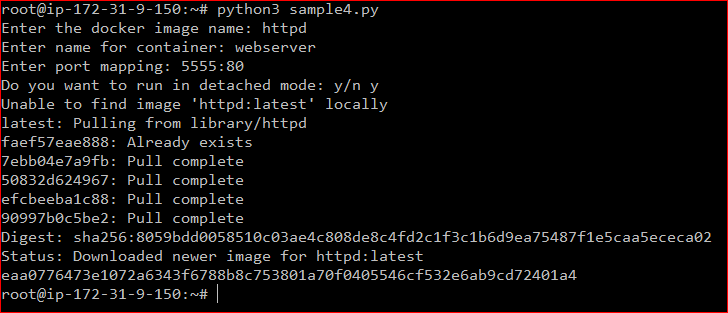
**Use case5: Enter input from the user**

Vim sample4.py

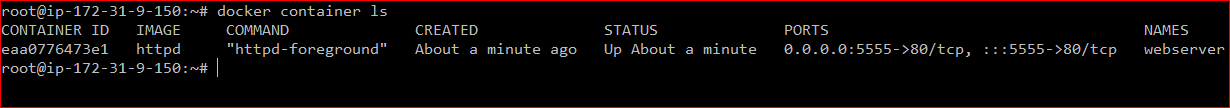


**now run the script**

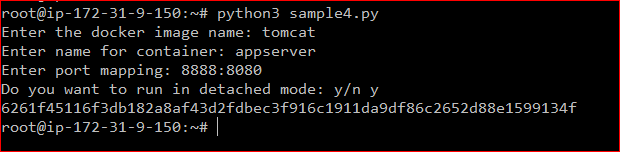
**python3 sample4.py**



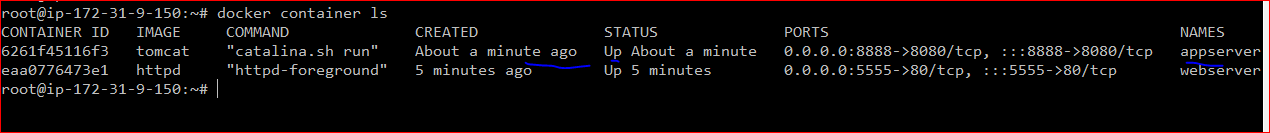
**Check the container**



I want take same program and I want to create **tomcat container**

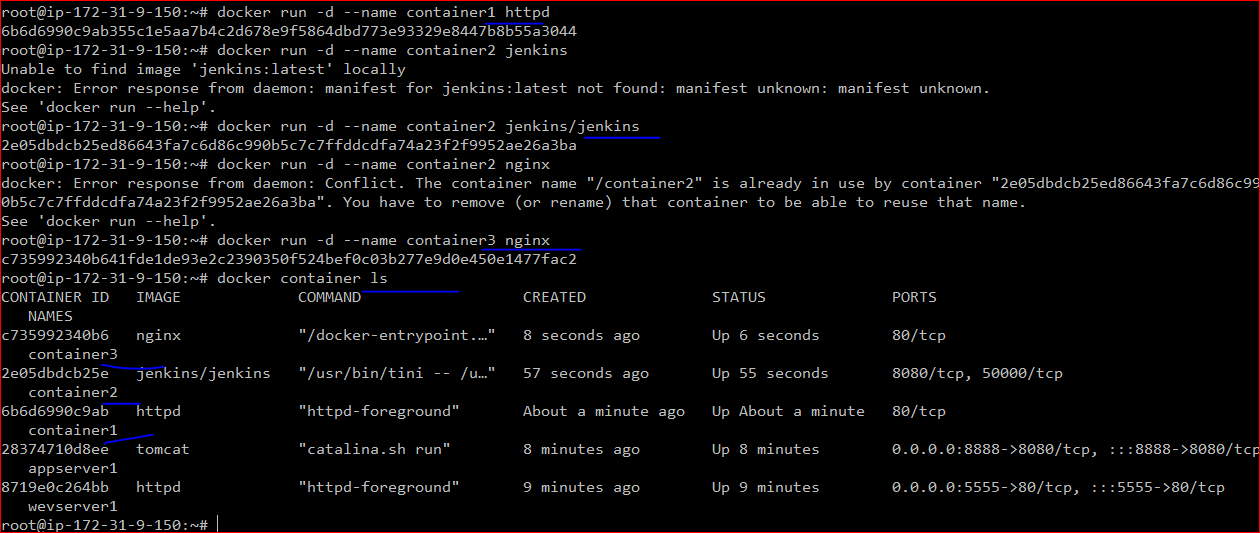


**Check the container**



----------------------------------------------------------------------------------------------------------------------

Create some container

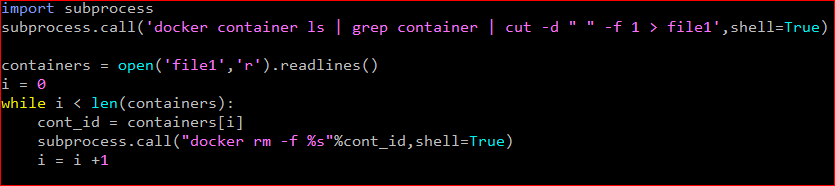


**Use case 6:**

Python program for deleting all the containers whose name start name container

Delete only those containers whose name starting with a name word called as ‘container’

**Vim simaple5.py**



**docker container ls | grep container | cut -d “ “ -f 1 > file1**

**docker container ls | grep container =🡺** this command we will capture the container id

container id is coming in the first column

now we to use **cut command** , where we use cut command , we have to use where is the space:

**cut -d “ “ -f 1 > file1**

d 🡺 Delimiter, separated by space

-f-🡪 which column we want to capture

1 is the column -🡪 only the first column we will capture

Now once the first column is captured, I want to store the data in file1

**Once the the data stored, how do you read data from a file ?**

Use open module and read the data which function call that **readlines()**

**open(‘file1’,’r’).readlins()**

**readlins()** function read the data line by line

all this information, we will store in variables called as **container**

**containers = open(‘file1’,’r’).readlins()**

containers is a variables which is going to contain all the id of containers

because multiple container ids are captured

we do **while loop** for one after another

**I = 0**

**While I < len(containers):**

**cont\_id = containers[i]**

once is done we have to delete the containers

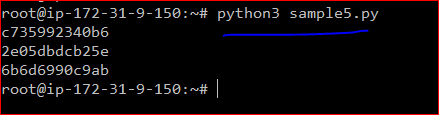
**subprocess.call("docker rm -f %s”cont\_id,shell=True)**

**I = I +1**

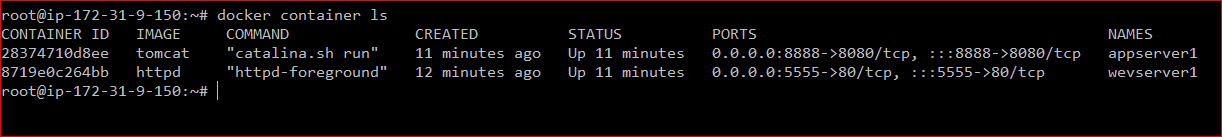
every iteration I value should be increment

-------------------------------------------------------------------------------------------------------------

Run the script



Check the container

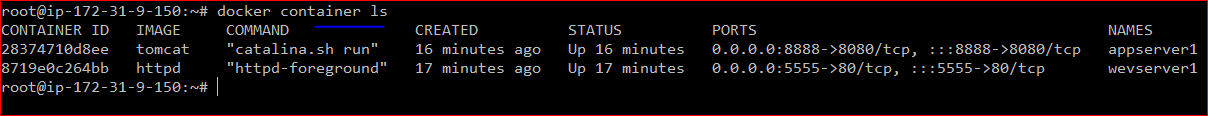


**Use case 7:**

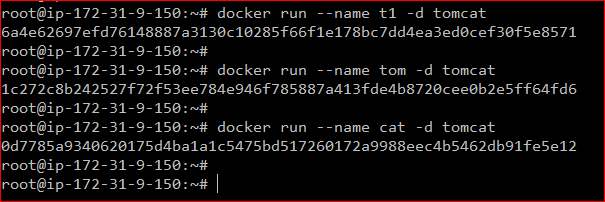
write a python program to delete all the tomcat container

**We have to create a multiple container**

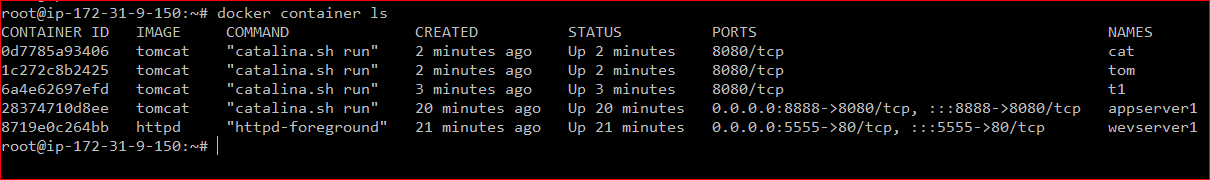
**Check the container**



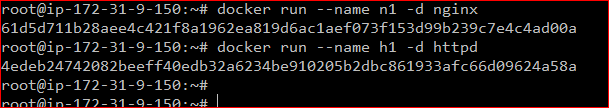
**Create a 3-tomcat container**



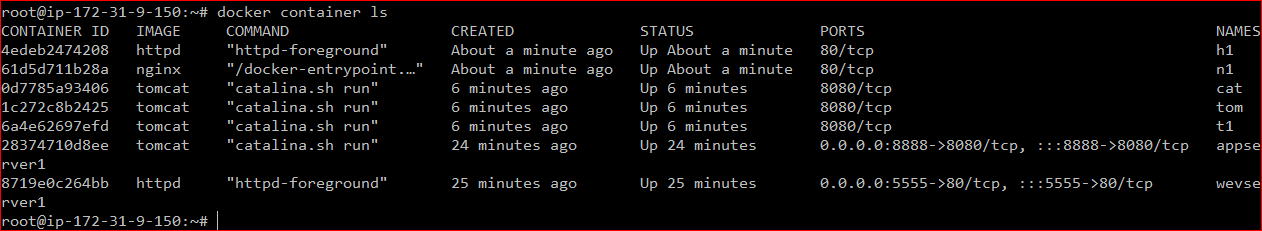
**Check the container**



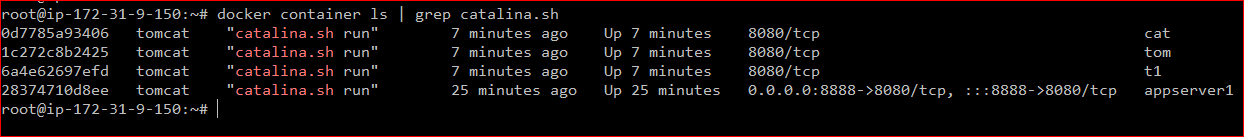
**Create httpd and nginx container**



**Check the container**

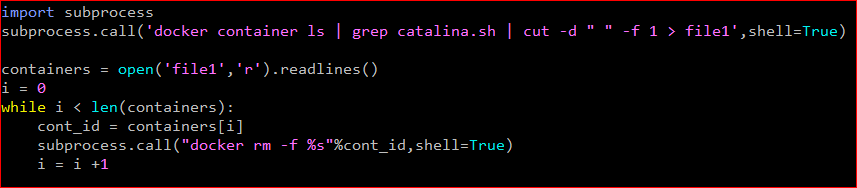


**Now we have to grep catalina.sh**

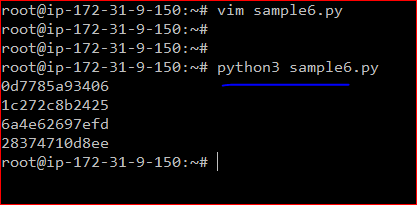


**Now write a python program for delete tomcat container**

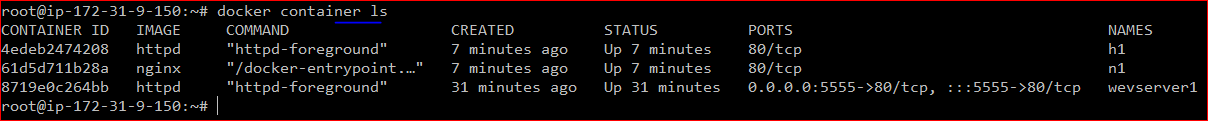
**Vim simple6.py**



**Run the script**



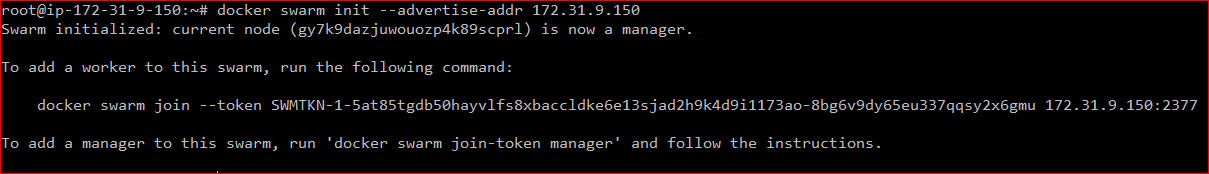
**Check the container**

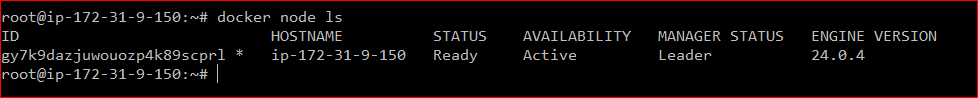


**Use case 8: docker swarm**

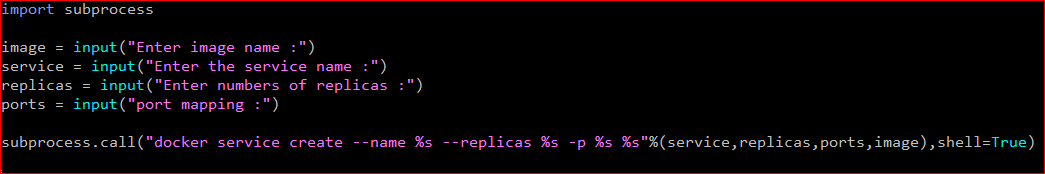
Python program for interactively deploying service in docker swarm

**Install docker swarm**

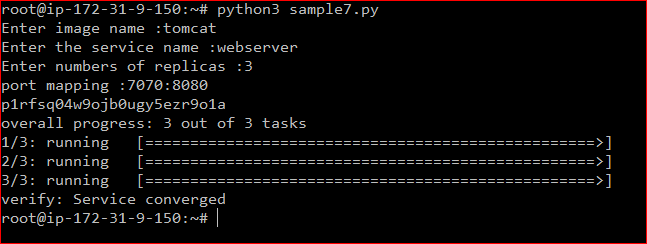




**Vim sample7.py**



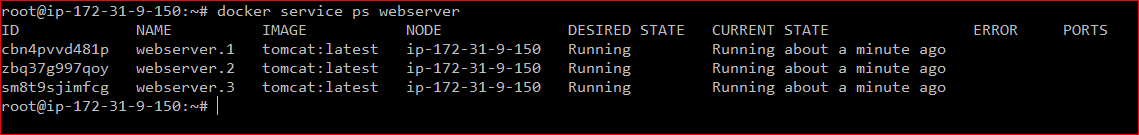
**Run the script**



**Check**

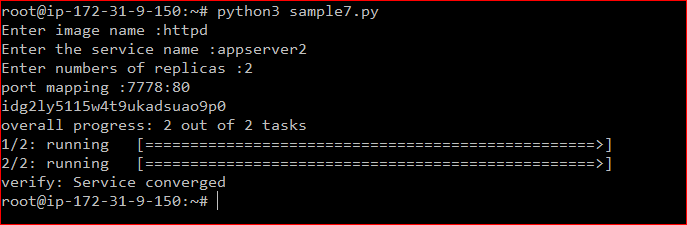
We have 3 replica , all are running on the same machine

Because we don’t have any worker node



**----------------------------------------------------------------------------------------------------**

So, I can use the same script, and I can deploy another service also



**Automatic Jenkins using python**

**Create an AWS instance and install Jenkins on it**

**Install python pip**

apt-get update -y

apt-get install python3-pip -y

pip3 –version

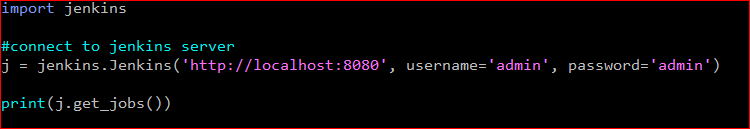
**install python Jenkins**

pip install python-jenkis

**Use case 9:**

**Python program to connect to the Jenkins server and display all the available job,**

Vim sample1 .py



We have to use Jenkins module

**import Jenkins** 🡪 this is Jenkins module,

this Jenkins module has all the method which are Necessary automating the different activities of Jenkins

**first a fall, my python program should connect to the Jenkins server, to do that, will use:**

Jenkins.jenkins(“location of Jenkins is running”, “username”,”passwd”)

**Ex: jenkisns.jenkins(“http://localhost:8008”,”admin”,”admin”)**

With this code connection between python program and Jenkins server will establish

And once it is done, I will store the information variables is called as**🡪 ‘j’**

Which means whenever I **used ‘j’**, it understand, i talking about the connectivity between our python program**(http://localhost:8008)** and Jenkins based on credential **(”admin”,”admin”)**

Now once the connection is established

**I want to display the job, which are available**

**Print(j.get\_jobs())**

**‘J’** object, there is a method called as **get\_job()**

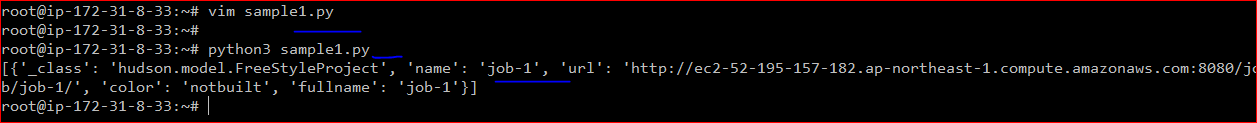
All these methods are present in Jenkins modules

This will display the all the job which are present

------------------------------------------------------------------------------------------------

Now run the script:

Python3 simple.py



-------------------------------------------------------------------------------------------------------------------------------

**Use case 10:**

**Python program to build or execute a Jenkins jobs**

Imagin that there is a Jenkins job and I want to run that Jenkins job do my python program rather than running it manually

Go to Jenkins dashboard

Create a job

Click on new item=🡺 enter the name 🡺 Dev1

Click on free style project

Select source management

Selet git

Enter url (maven url)

Apply and save

It will down load the code from git repo

-------------------------------------------------------------------------------------------------------------

**Now we to create one more job**

Click on new item=🡺 enter the name 🡺 Dev2

Click on free style project

Select source management

Select git

Enter url (functional testing)

Apply and save

**Now we to create one more job**

Click on new item=🡺 enter the name 🡺 Dev3

Click on free style project

Select source management

Select git

Enter url (functional testing)

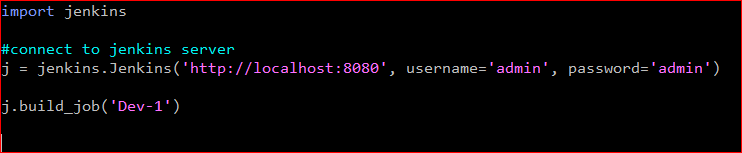
Apply and save



Now I want to run the job from python program

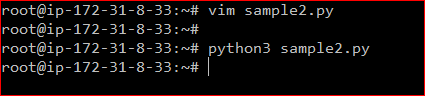
Now execute the job

Vim sample2.py



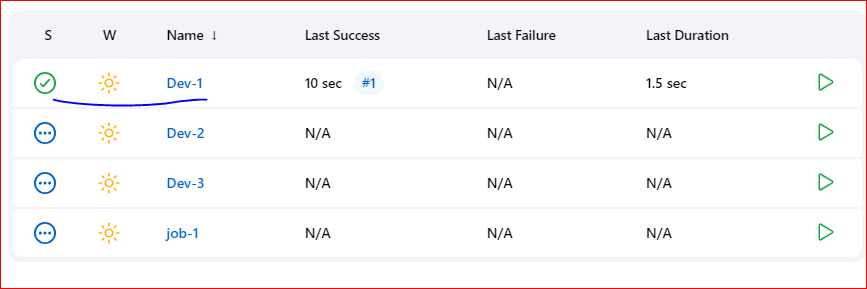
We have to use **build\_job(‘Dev1’)**

Now run the script



Go back to the dash board of Jenkins

Refresh the page



Rename the job name Dev1,Dev2 ,Dev3

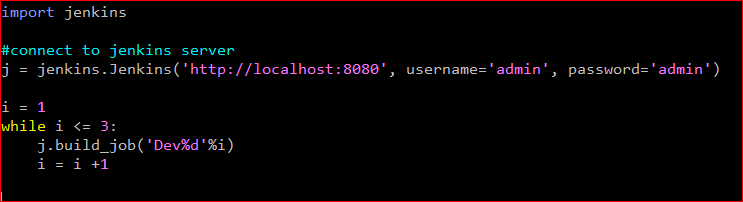
**Use case 11:**

**Python program to execute all the jobs :**

I want to run all job using python script

We have a 3 jobs so I want to run all jobs

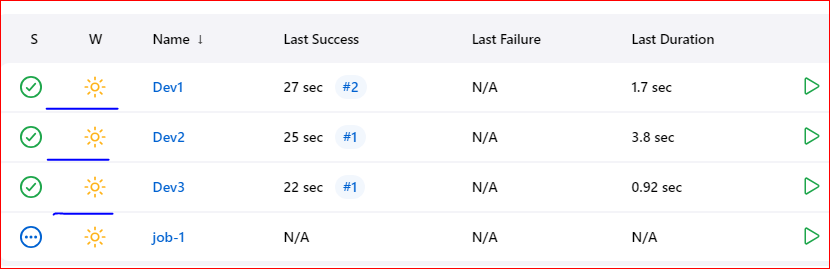
**Vim sample3.py**



**Now execute the program**



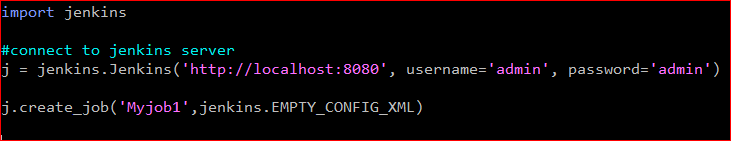
**Go back to Jenkins dash board and refresh the page**



**Use case 12:**

**Python program to create Jenkins jobs**

**Vim sample4.py**



Creat a job we have to use

Create\_job(‘job name’, ‘path of XML file provided by development team’)

Ex: now I am creatin a empty job

j.create.job(‘Myjob1’,jenkins.EMPTY\_CONFIG\_XML)

**--------------------------------------------------------------------------------------------------------**

Run the script:



**Go back to Jenkins dash board and refresh the page**

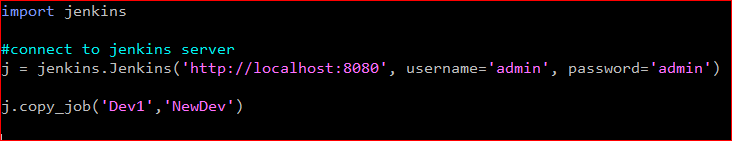


Myjob1 got created

**Use case 13:**

**Python program to take a backup of a Jenkins jobs**

**Vim sample5.py**



We have to use

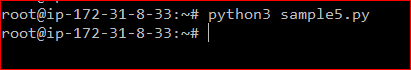
**Copy\_job**(‘jobname’, ‘give the new job name where we to save the job’)

Ex: copy Dev1 job and save the job in NewDev1

j.copy\_job(‘Dev1’,’NewDev1’)

-----------------------------------------------------------------------------------------------------------

Run the script:



**Go back to Jenkins dash board and refresh the page**



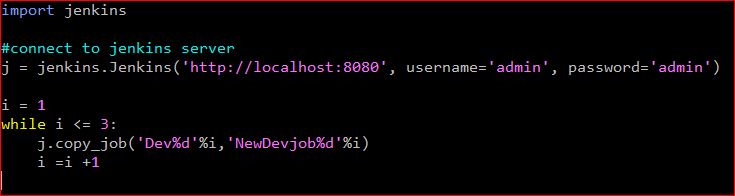
If you open Dev1 and NewDev1 both contain same data

**Use case 14:**

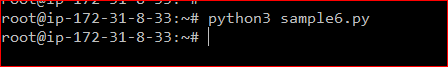
**Python program to take a backup of a multiple Jenkins jobs**

**Vim sample6.py**

I want to take back up of Dev1,Dev2, and Dev3 jobs to copy in NDev1,Ndev2,NDev3



Run the script



**Go back to Jenkins dash board and refresh the page**

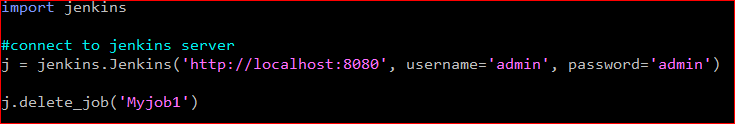


**Use case 15:**

**Python program to Delete a Jenkins jobs**

**I want to delete Myjob1**

**Vim sample7.py**

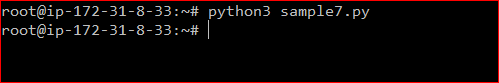


**We have to use**

**delete\_job(‘job name’)**

**-------------------------------------------------------------------------------------------**

Run the script



**Go back to Jenkins dash board and refresh the page**



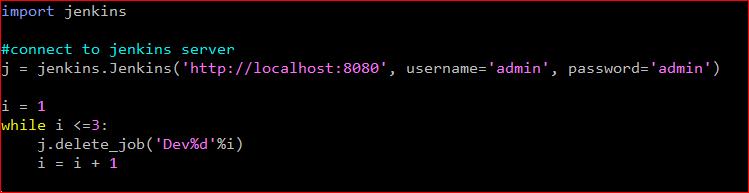
‘Myjob1’ is deleted

**Use case 16:**

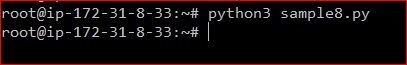
**Python program to Delete a multiple Jenkins jobs**

**Vim sample8.py**

**We have to delete 3 jobs , Dev1,Dev2, Dev3**



Run the script



**Go back to Jenkins dash board and refresh the page**

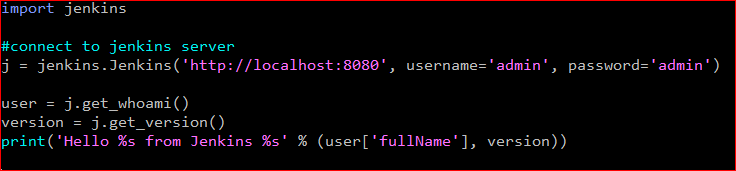


**Dev1, Dev2, Dev3 jobs deleted**

**Use case 17:**

**Get version of Jenkins**

**Vim sample9.py**



**Execute**

