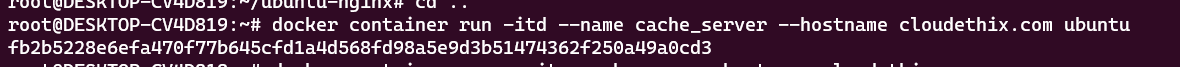
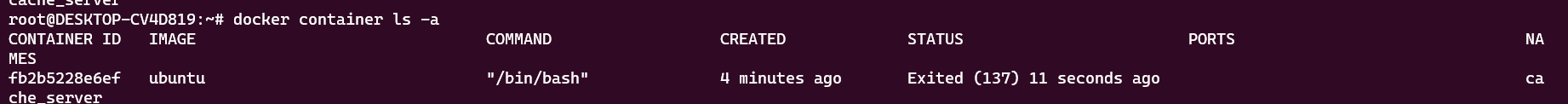
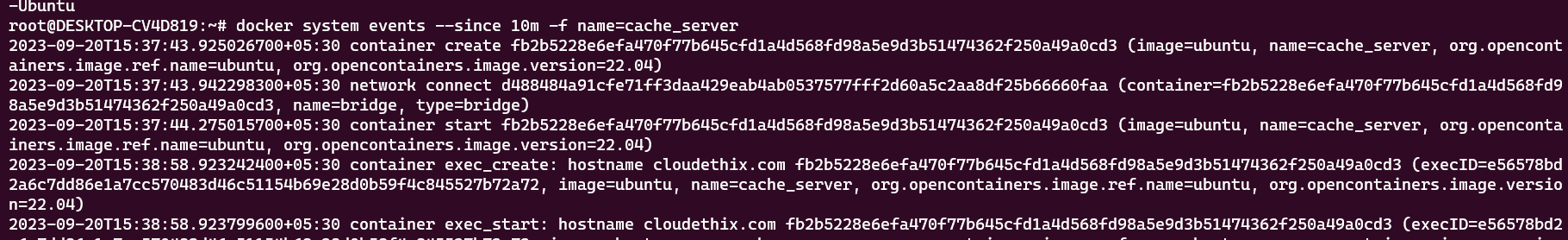
**1.Create a docker container using Ubuntu image with a meaningful name & set thehostname to the container cloudethix.com. Then stop or kill the container and check its exit code and check the log of the container for more details.**

**Answer:**



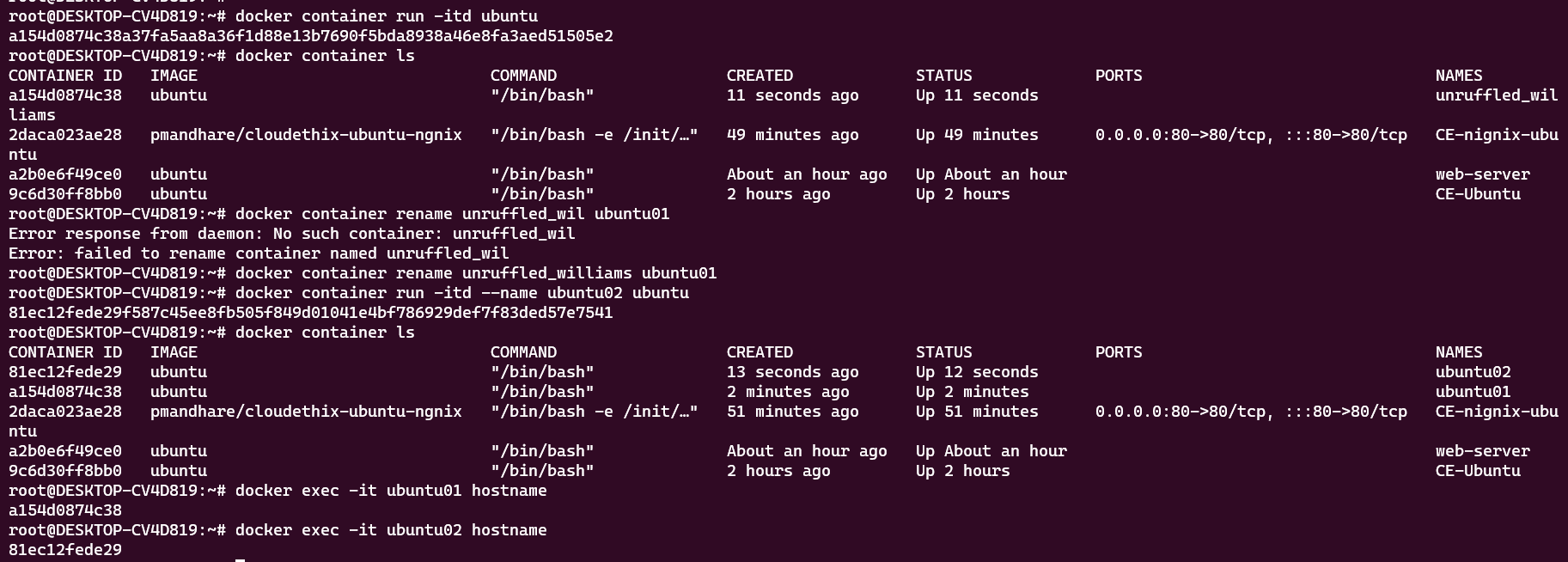


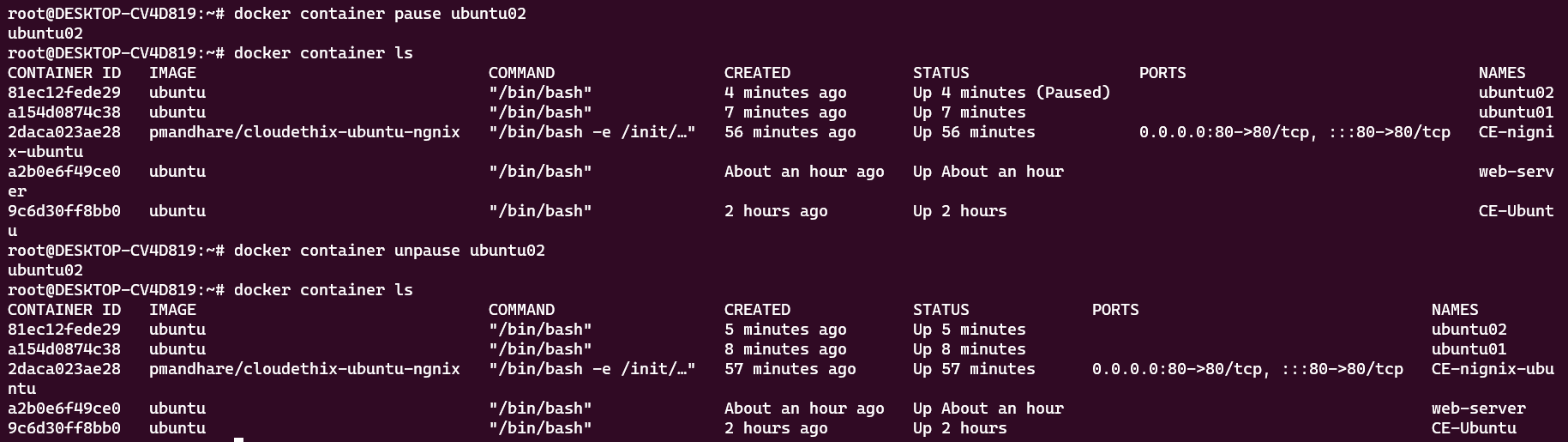


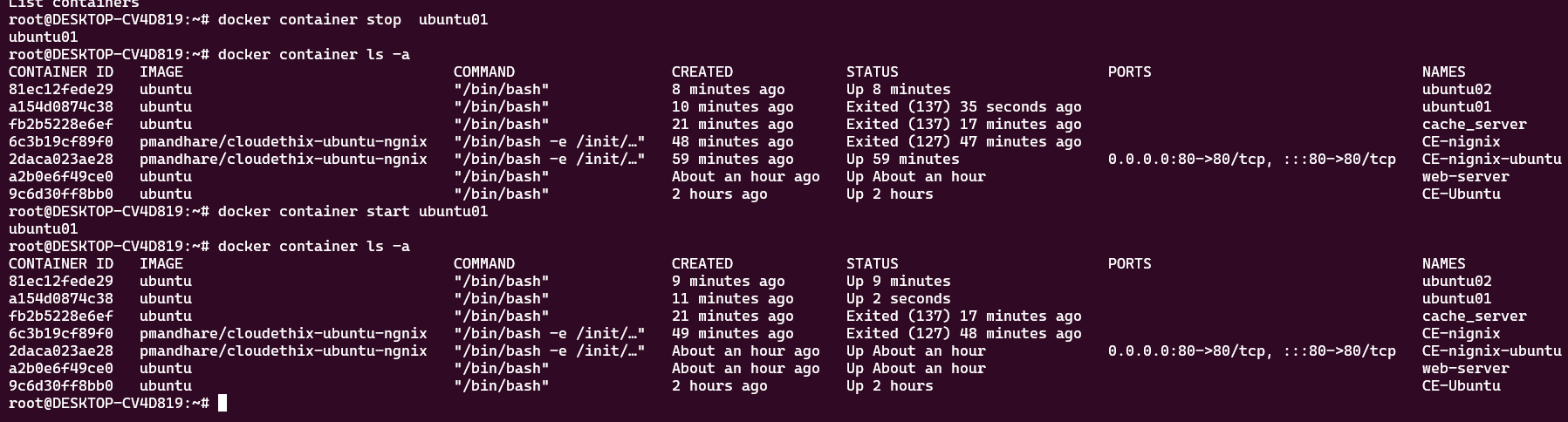


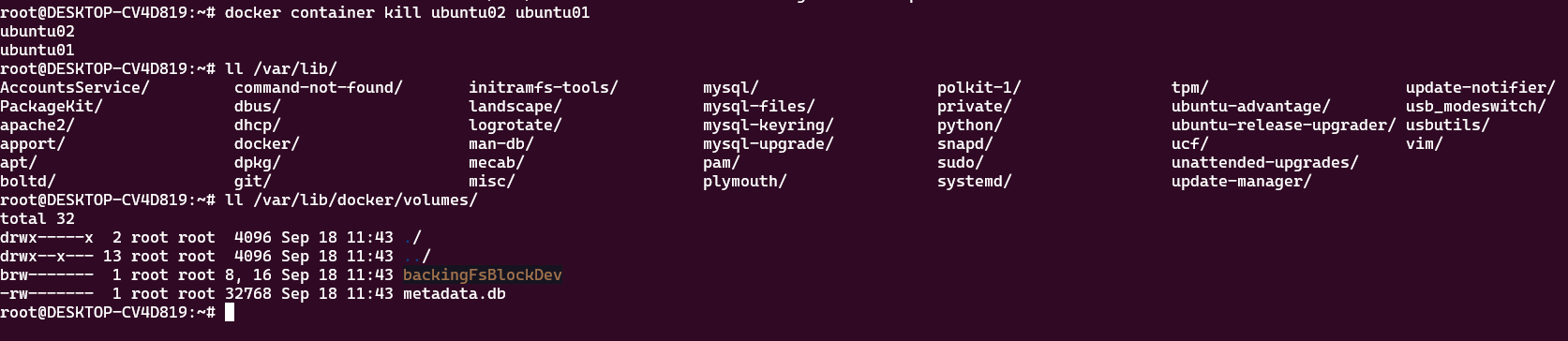
**2.Create and start an Ubuntu container with a random name. Then rename the containername to the meaningful name Ubuntu01. Then run another container named Ubuntu02and then check the hostname of both the containers. Then pause and unpause theUbuntu02 container and stop , start , restart Ubuntu01 container. Also check stat andsystem events of both the containers and then kill and delete the same. Also make suredirectories created by containers should be deleted.**

**Answer:**





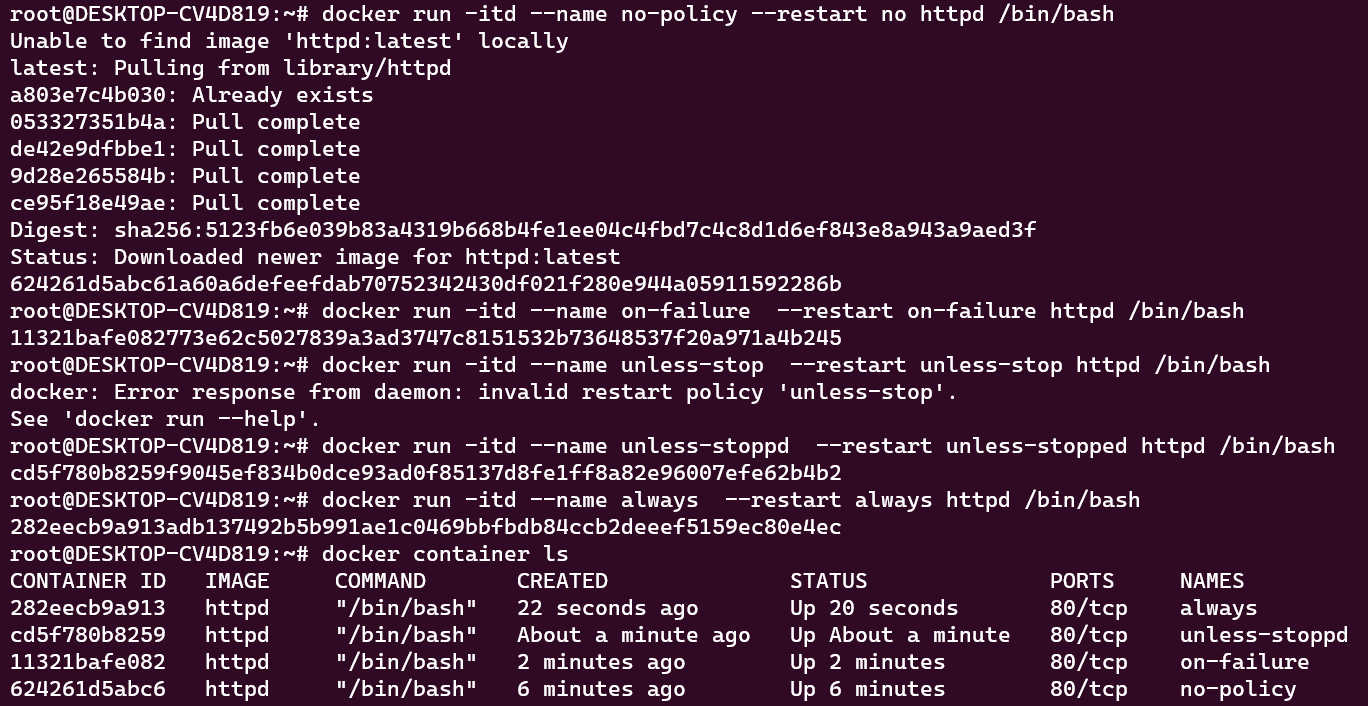




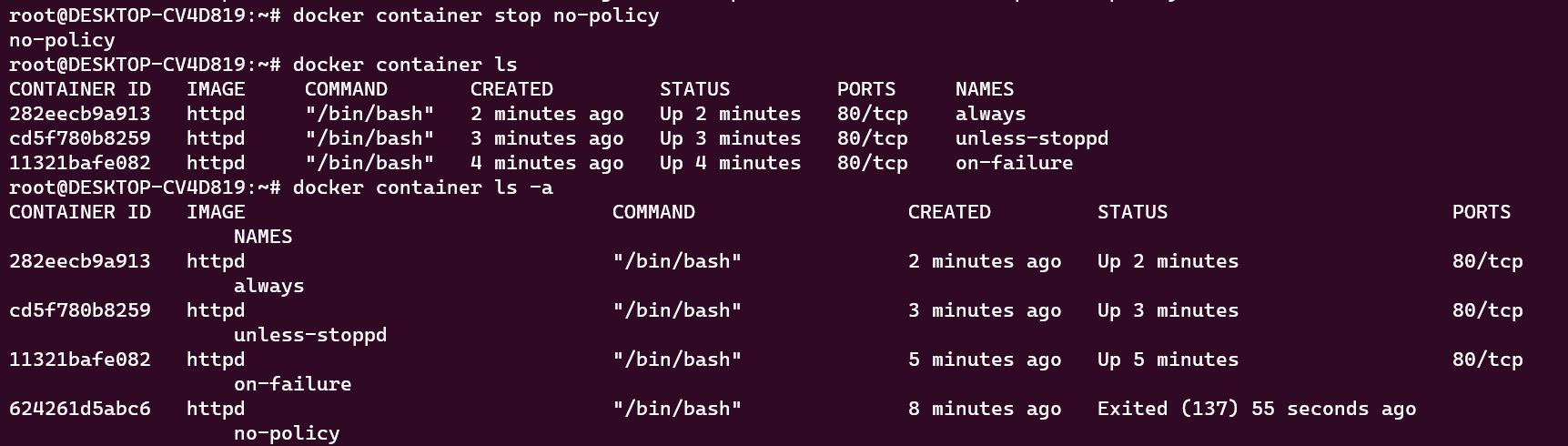
**3.Run four HTTPD docker containers with meaningful names to each container and thenapply restart policies NO , On-Failure, Always and Unless-Stopped on these 4 containersrespectively. Once applied, prove that restart policies are working.**

**Answer:**

* We ran four containers with the names and restart policies respective

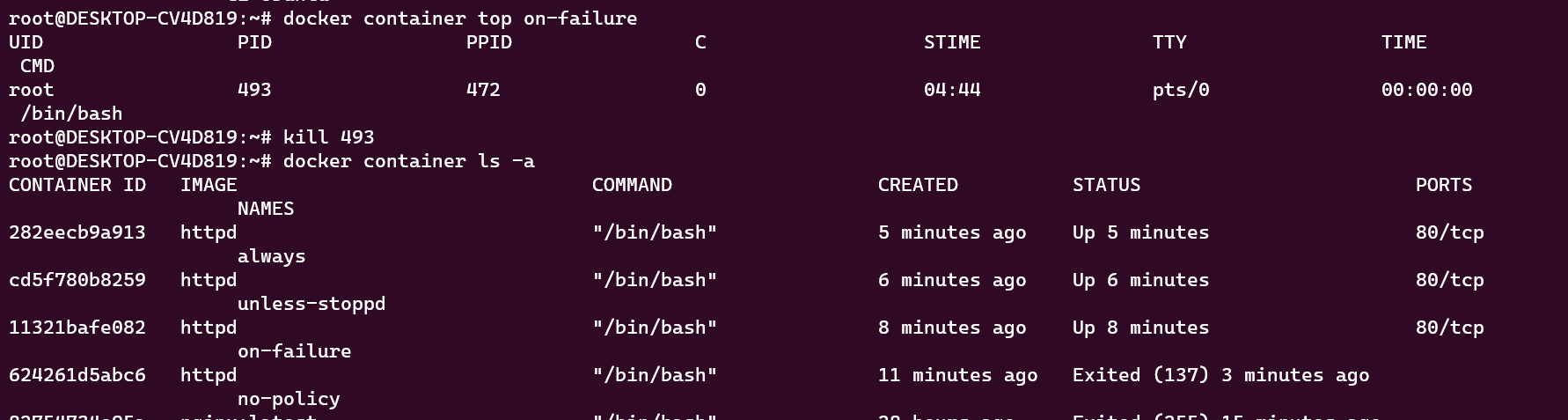


* We tested no\_policy container, and it did not start after rest

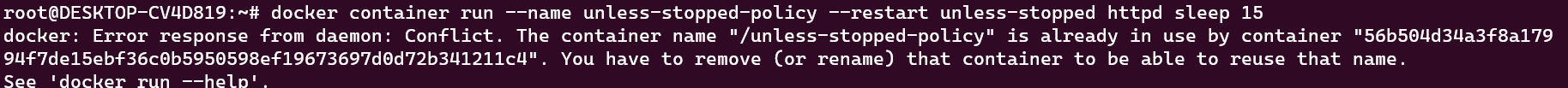


* We tested on failure container by killing the process using PID. The cotainer

restarted .



* We tested unless stopped container by giving it command to sleep



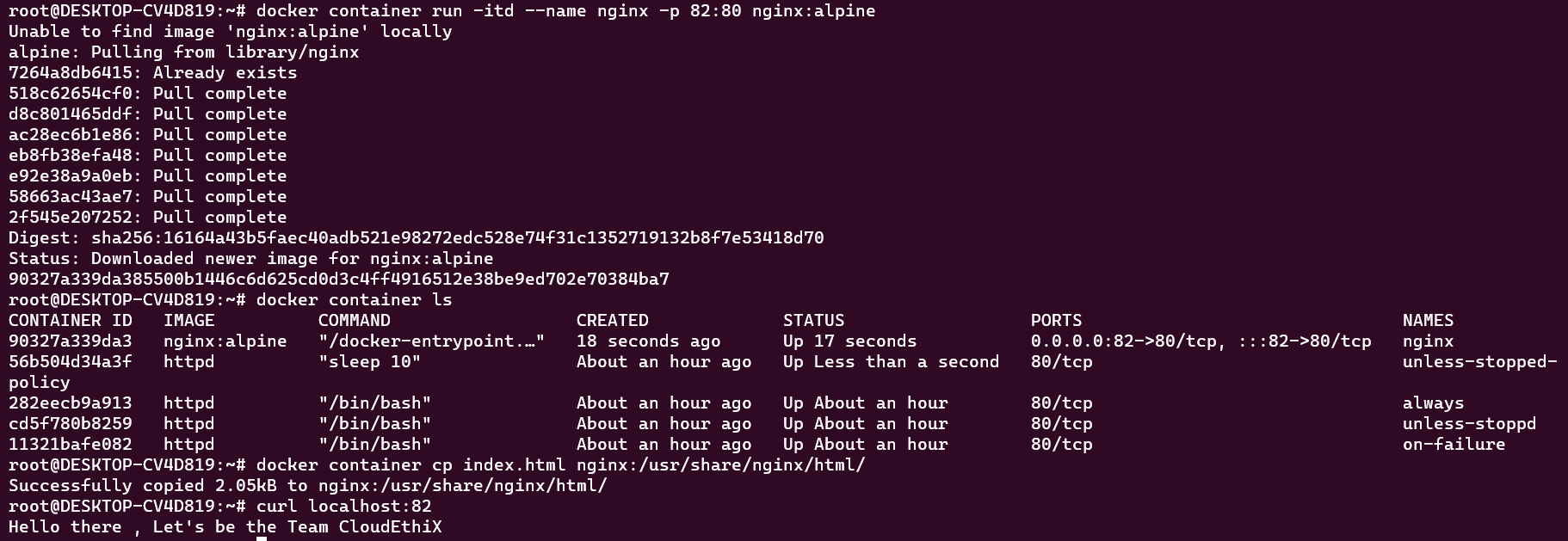
* For always restart the docker engine and check its still restarted or not

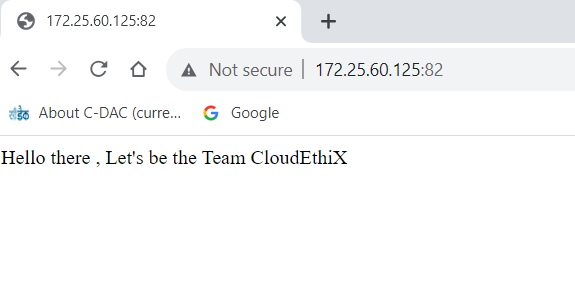
****

**4.Run an NGINX container with meaningful and expose the container on host port 80. Also create an index.html file with data "Hello there , Let's be the Team CloudEthiX" . Then copy the index.html file to the container under /usr/share/nginx/html/ location. Once copied try to access the container in the browser and check if the webpage is getting displayed properly**

**Answer:**

* Below we run an ngnix container, write an index.html file in our local system and copy it to the container. We are able to access the page

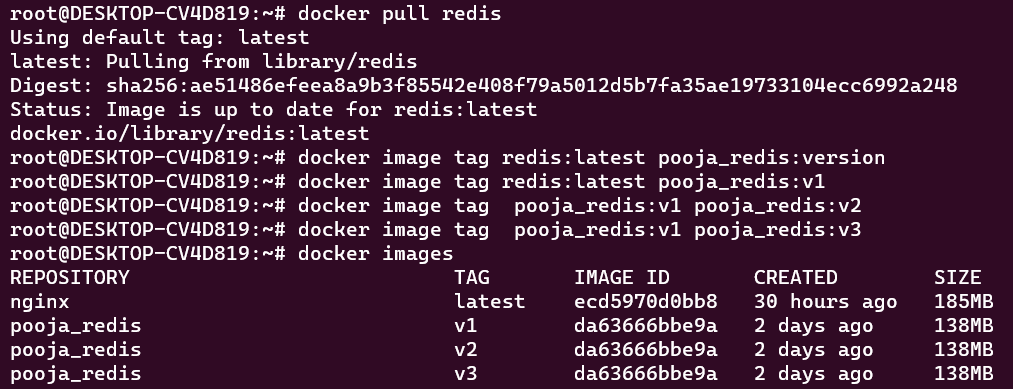




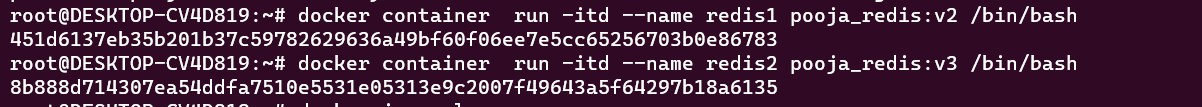
**5.Run a command to Pull the redis image. Then tag the image with yourname\_redis:version 1. From this tag create another 2 tags for yourname\_redis:version 2 & 3. Once images are tagged , run two docker containers from these tagged images with name redis01 and redis02. Then delete the image with tag yourname\_redis:version 2.**

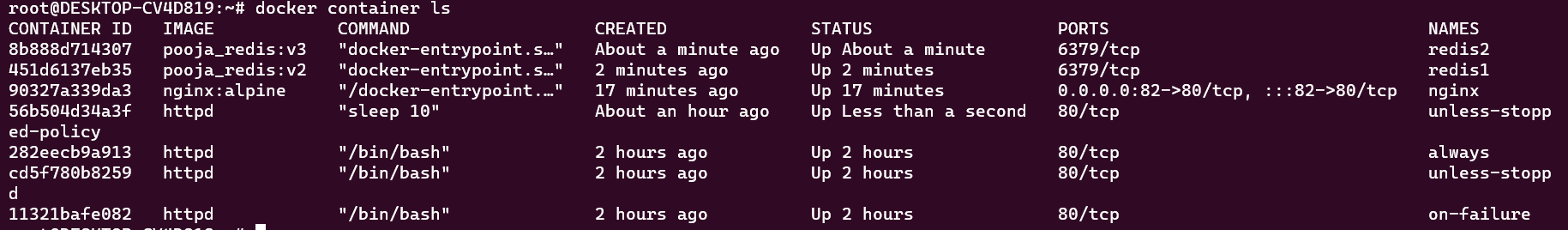
**Answer:**

* We have pulled the redis image and tagged it once. Then we have tagged 2 more tags with the first tagged im

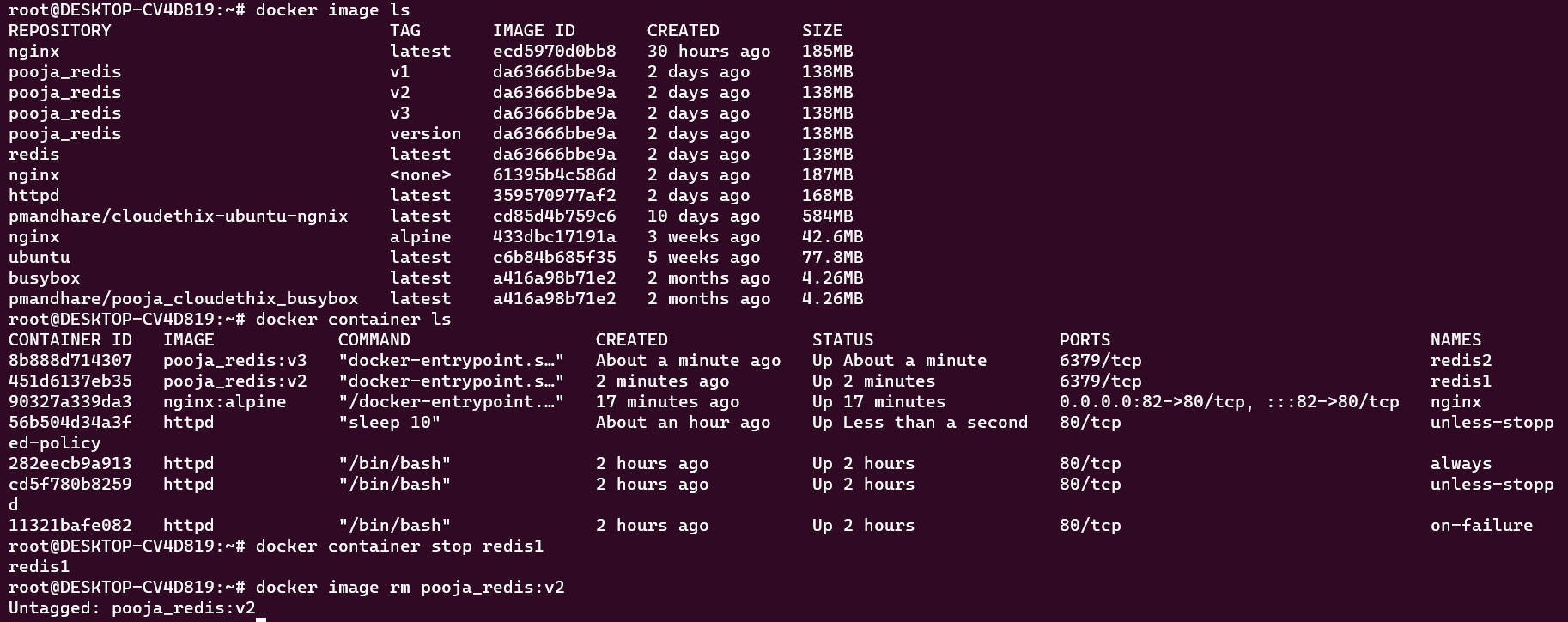


* l We have run 2 containers with the image tag





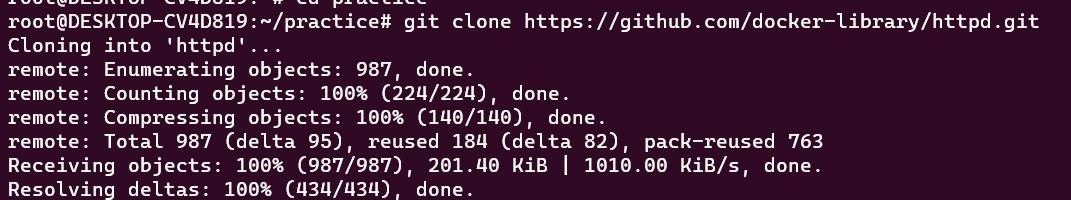
* To delete the image, we need to stop the container and then we will delete the image.



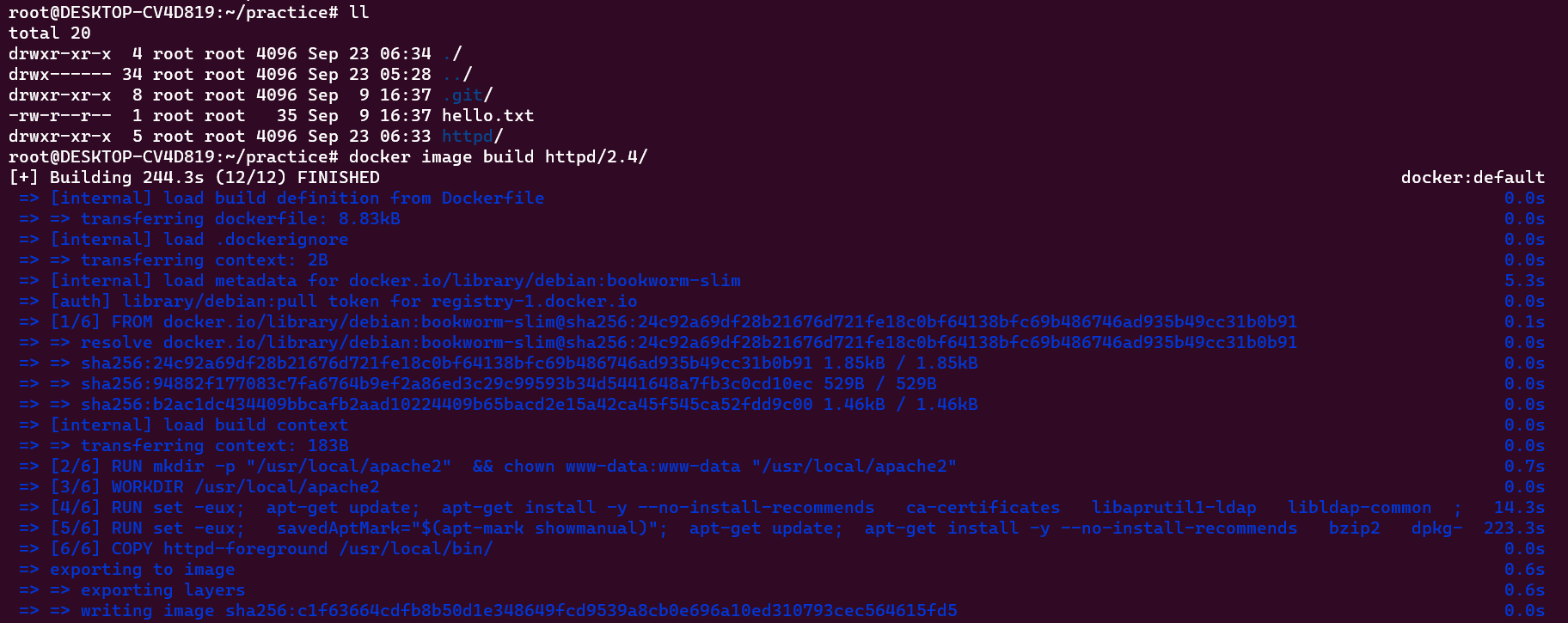
**6 . Clone the https://github.com/docker-library/httpd.git Git repo locally and build the httpd docker image from the Dockerfile from path httpd/2.4/Dockerfile. Then run & expose the Docker container from this image with a meaningful name on port 8181. Access the webpage in the browser. Once succeeded, tag the image as V1. 7.Push the redis images named version 1 & 3 to your Docker Hub repository named “yourname\_cloudethix\_redis”.**

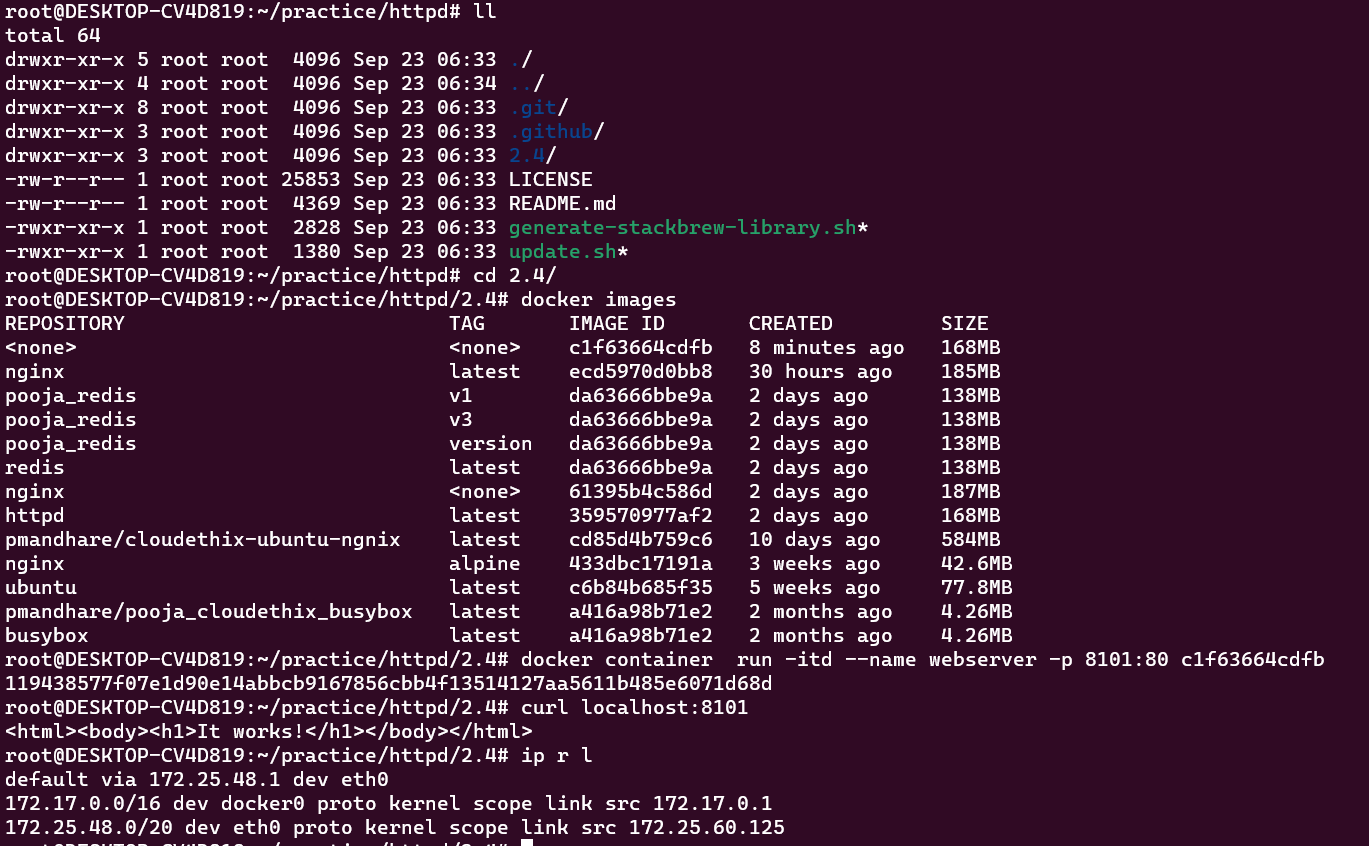
**Answer:**

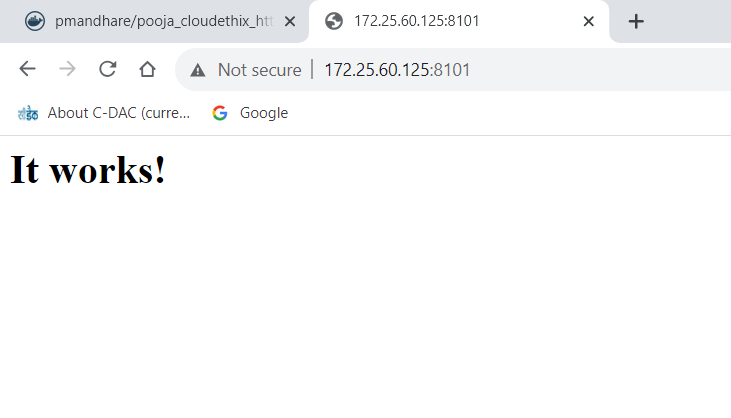
* We clone the repo and build the image

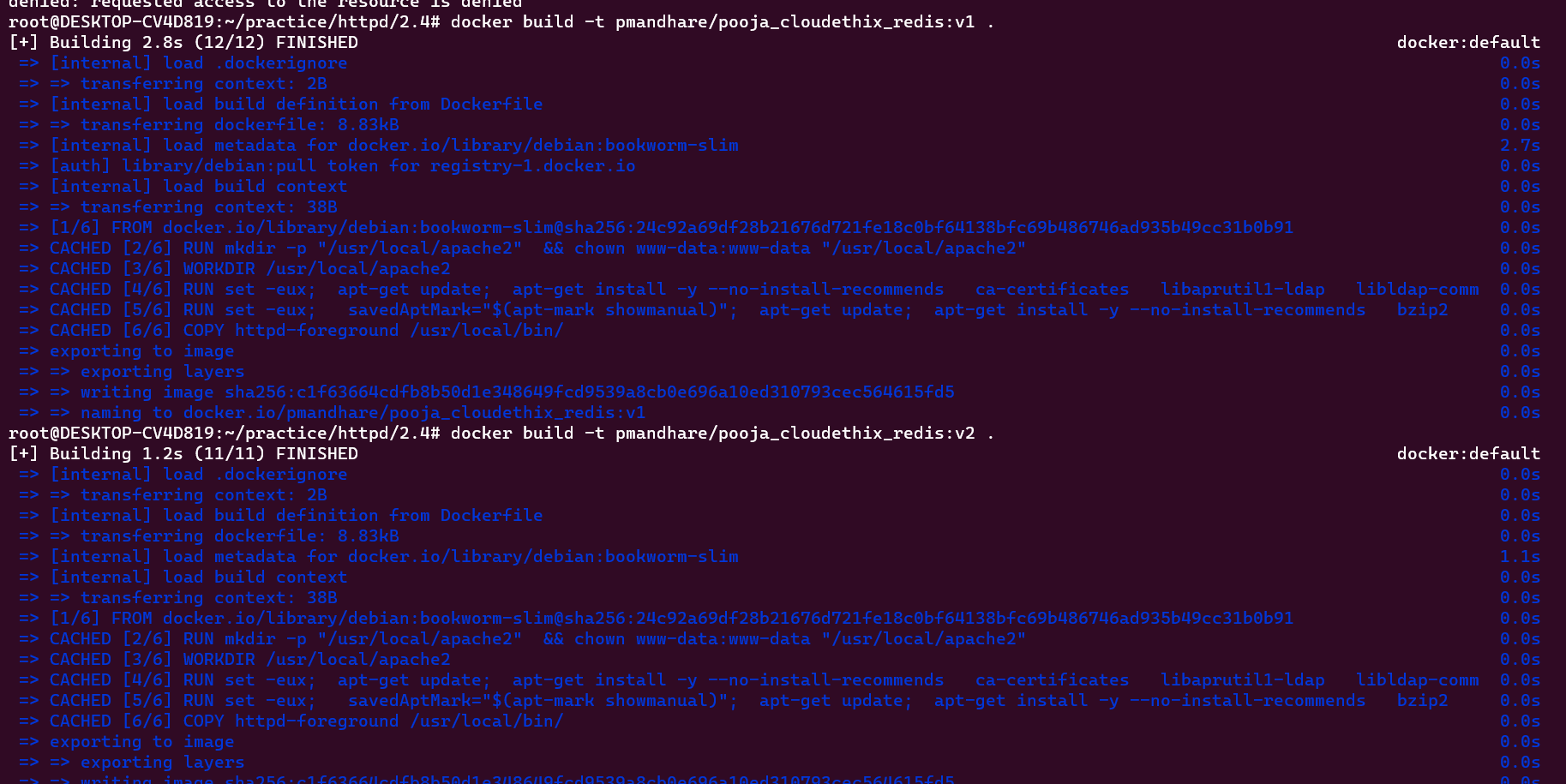


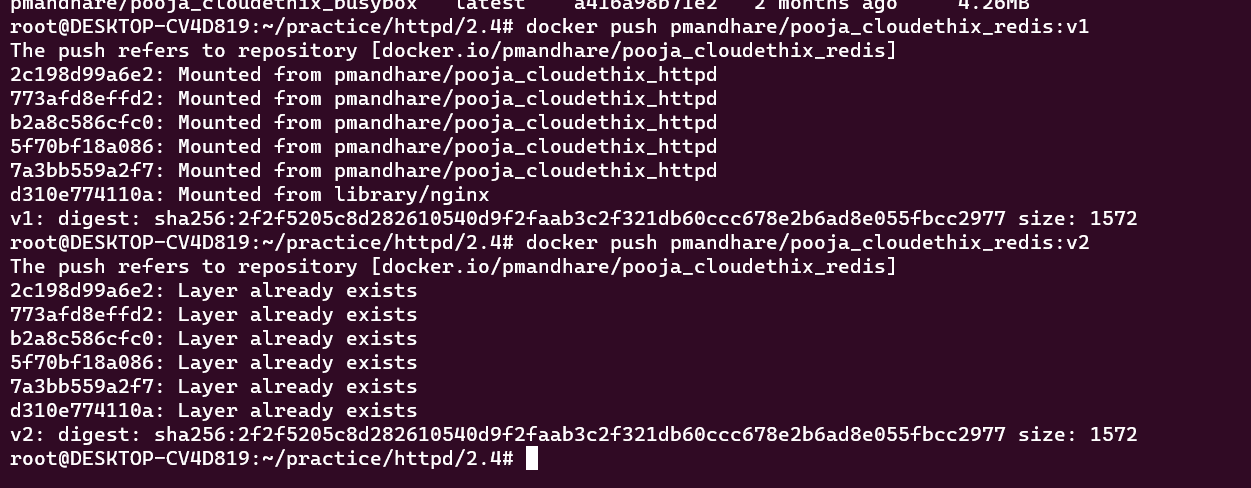
* We have tagged and run a container using the image and tagged the image as v1

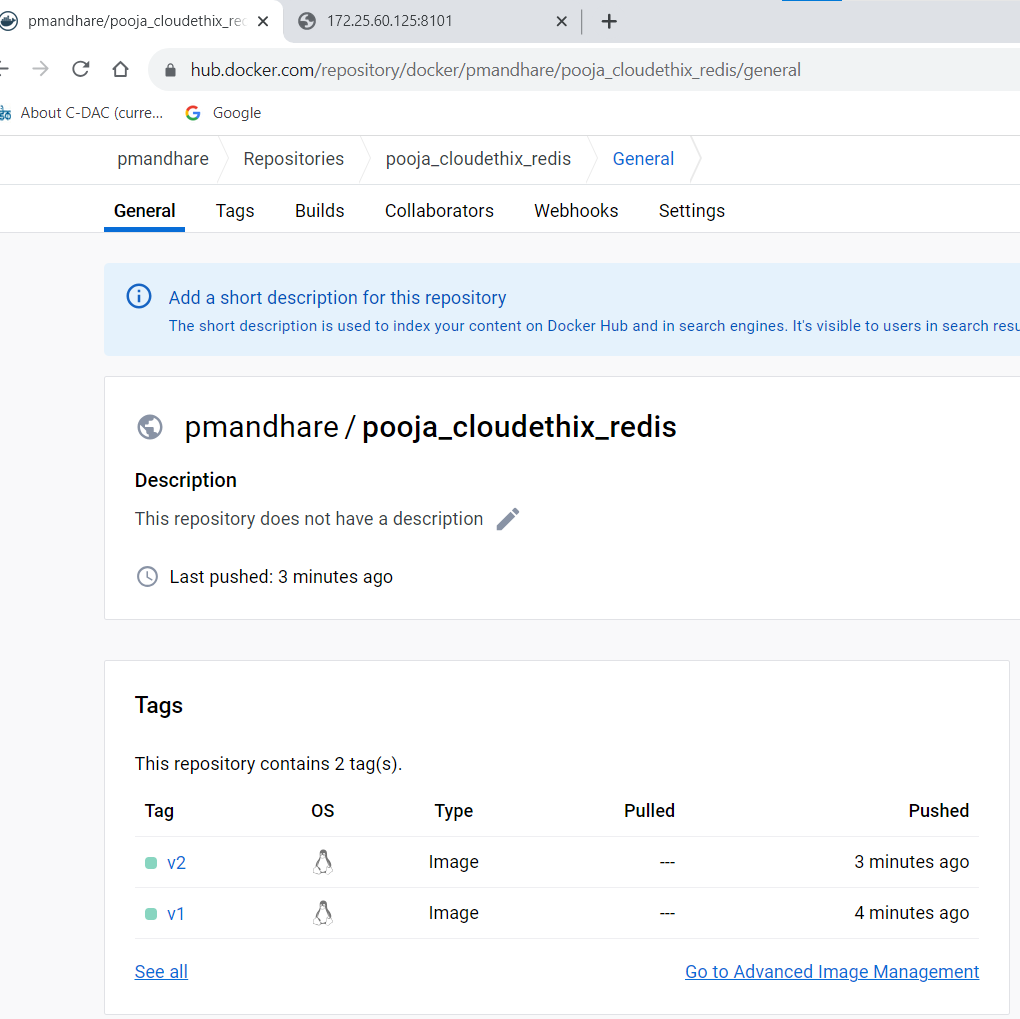




****7.Push the redis images named version 1 & 3 to your Docker Hub repository named “yourname\_cloudethix\_redis”.**

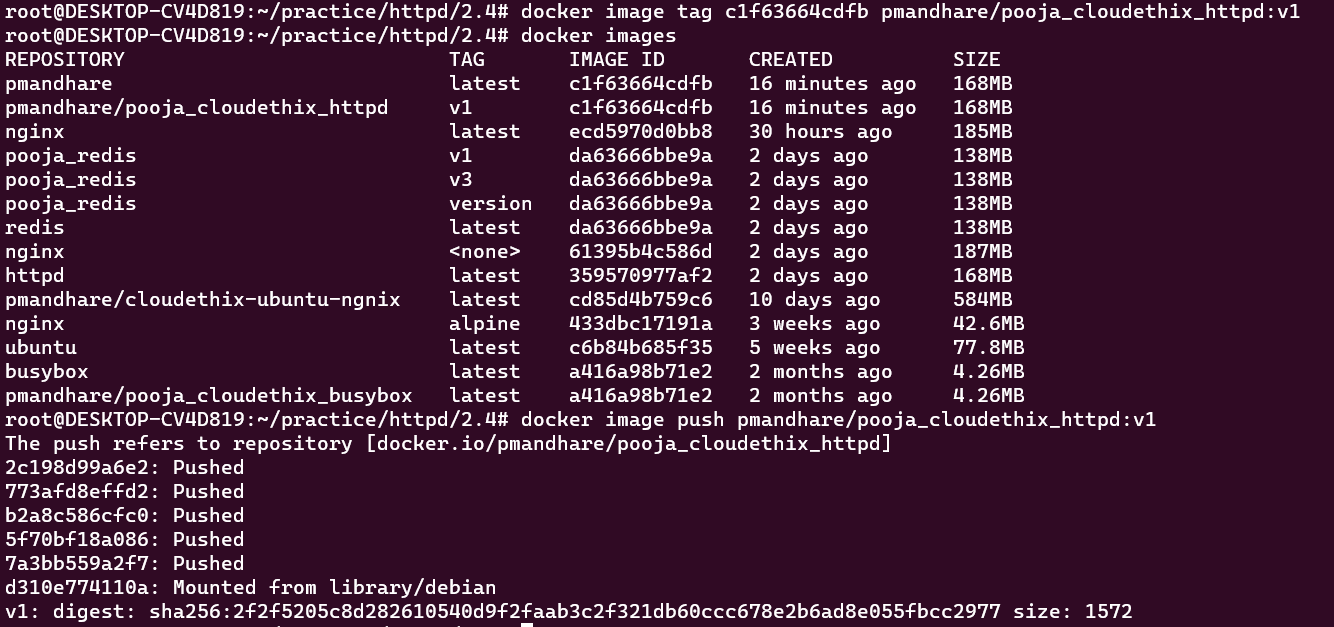
****

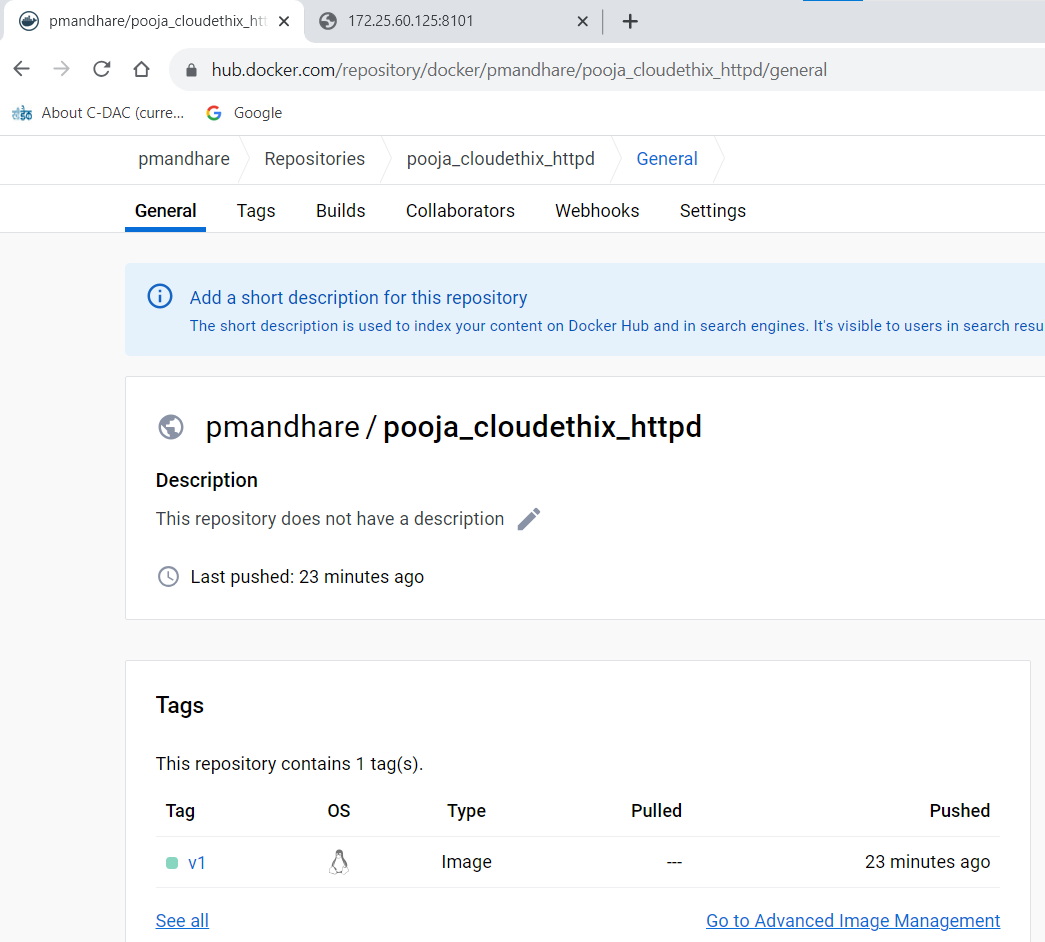
****

****

**8. Create a Docker Hub Account & generate security token. Also create a Docker Hub repository named yourname\_cloudethix\_httpd. Then login to the Docker Hub and then push the above httpd tagged image to your Docker**

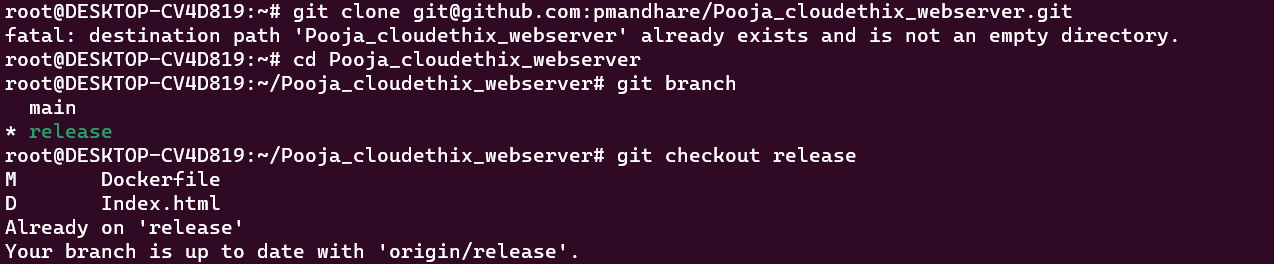
* *We have a docker hub account. Now we will push this image*

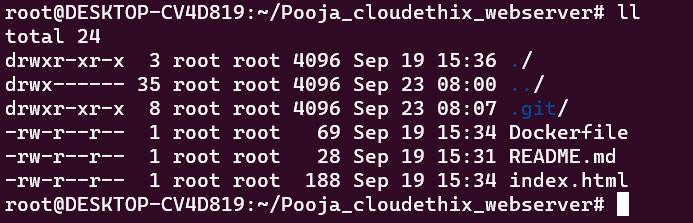
****

****

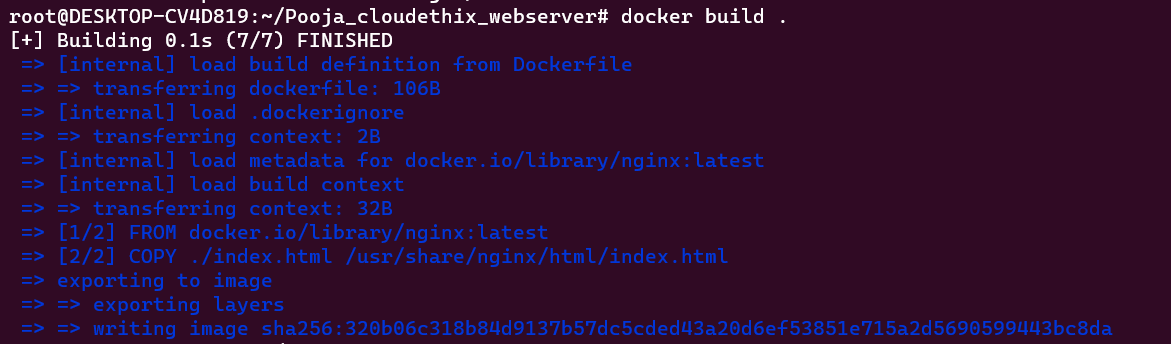
**9.Create a remote GIT repository and Add Dockerfile and index.html file as given in the documentation under Build NGINX Image section. Clone the repository locally , and create a release branch. From the release branch build the docker image with some meaningful tag. Once the docker image is built , run a container from that Image & expose it to host port 8383. Check the webpage in the browser. Once succeeded , push the same image to your Docker Hub repository named “yourname\_cloudethix\_nginx”. Once Everything is tested , push the release branch to your remote git repository & merge it by creating PR.**

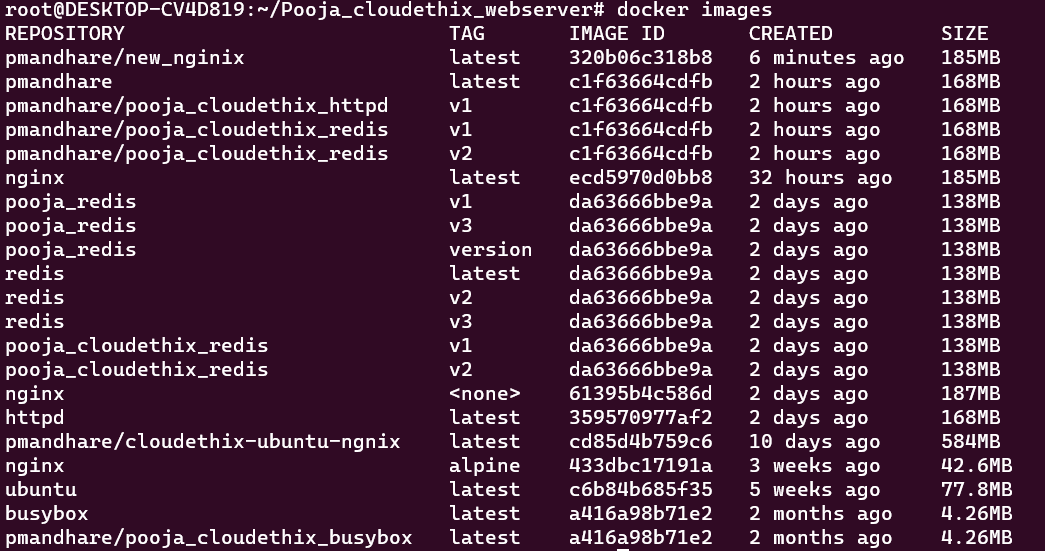
**Answer:**

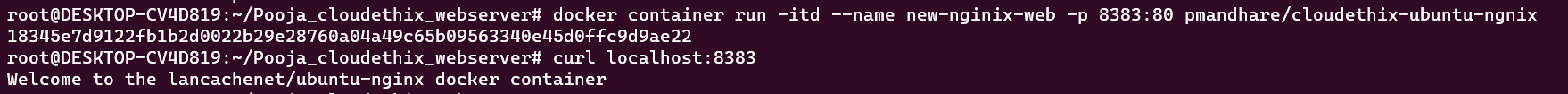
****

****

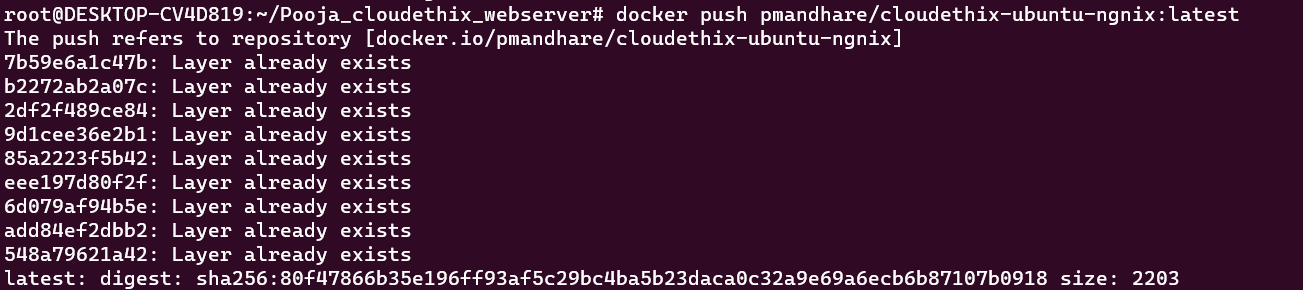
* *Image build and run the container from expose it to host port 8383*

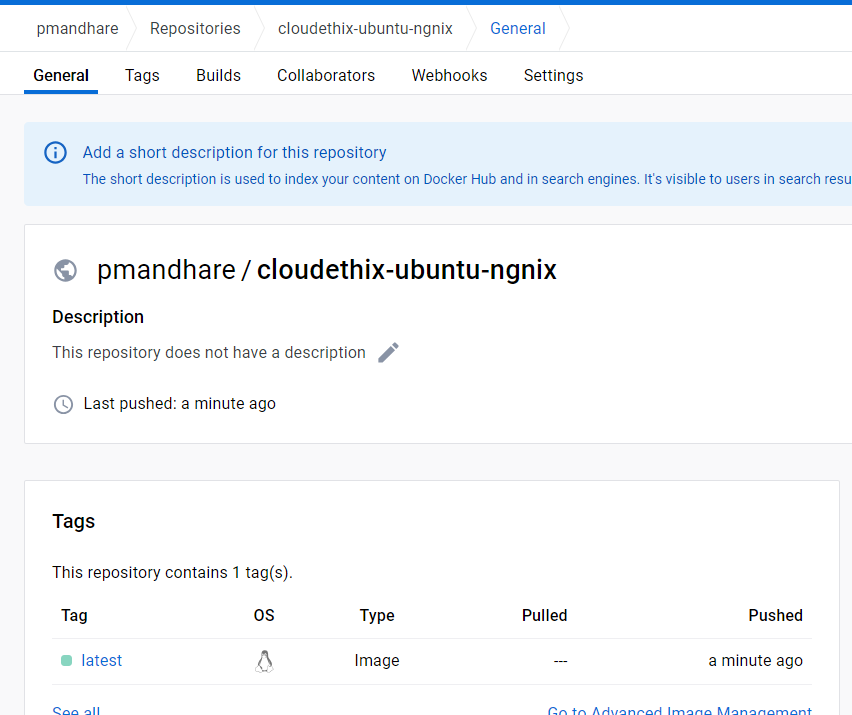
****

****

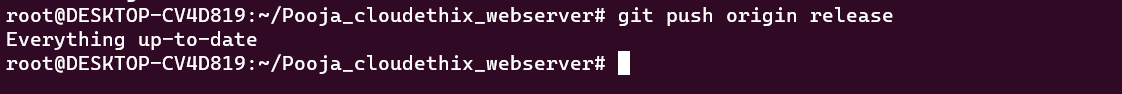
****

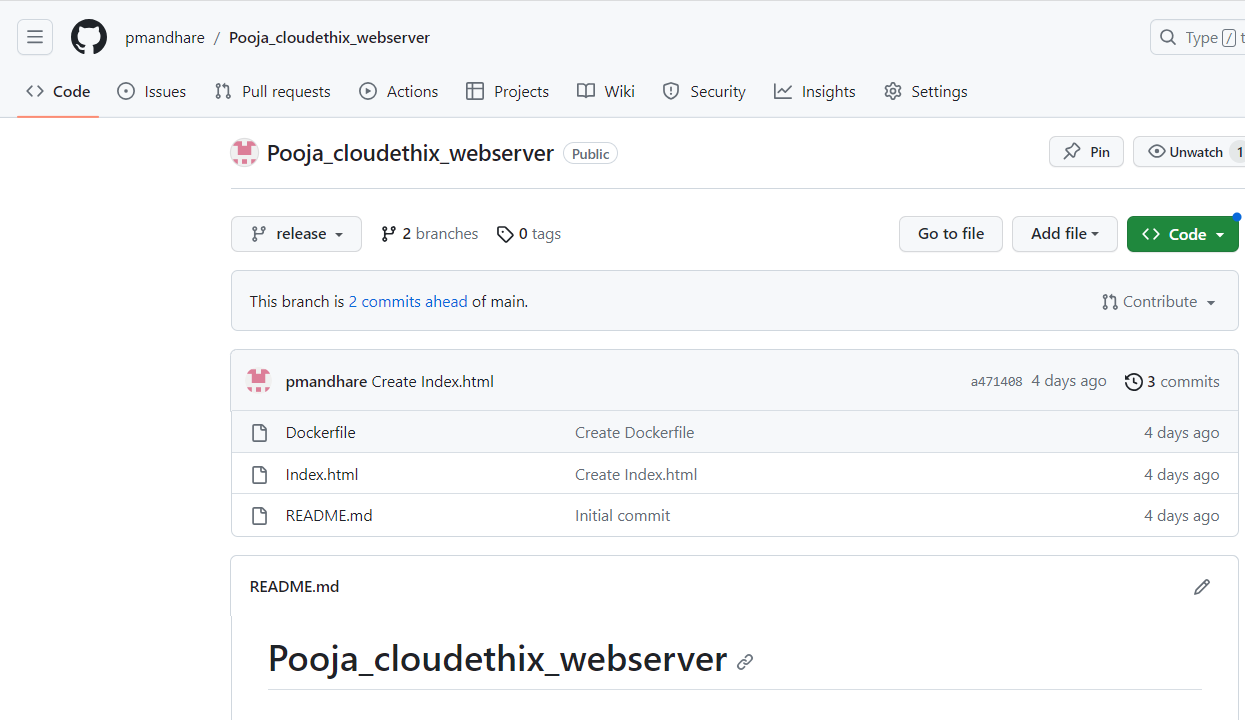
* *Pushed the same image pmandhare/cloudethix-ubuntu-ngnix:latest to Docker Hub repository named “cloudethix\_ubuntu\_nginx”.*

****

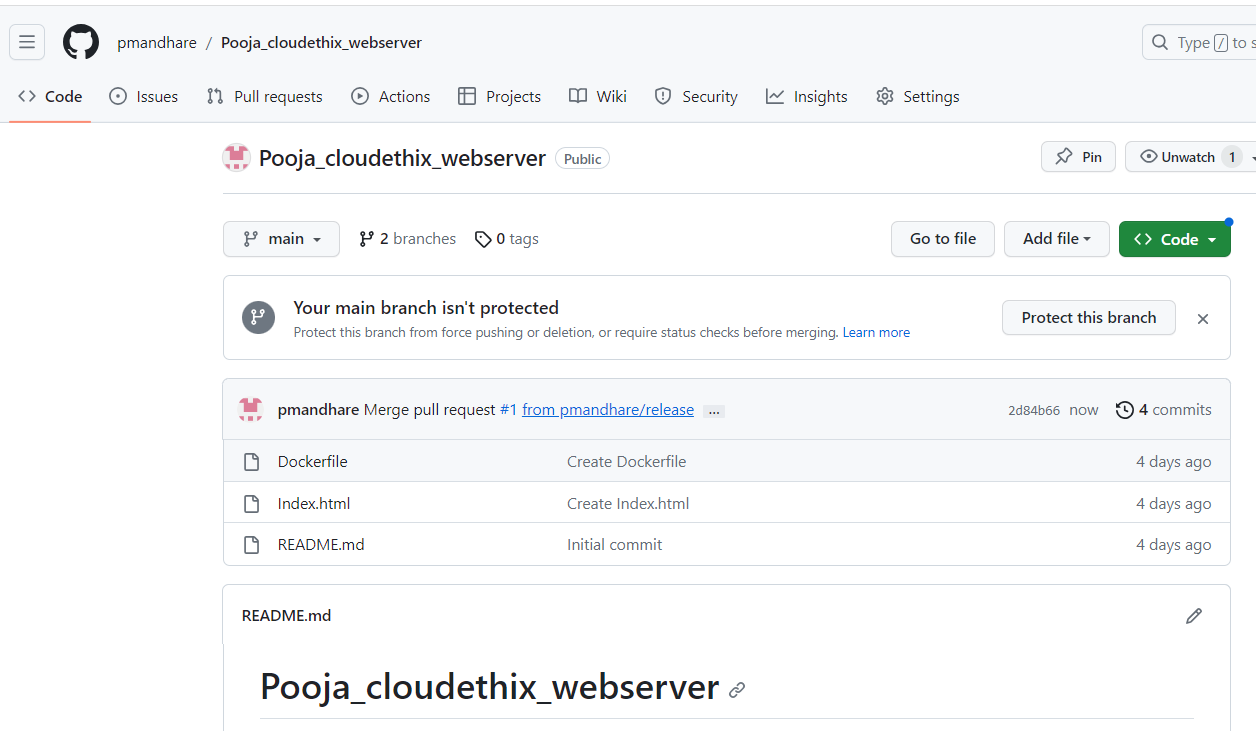
****

* *pushed the release branch to remote git repository*

****

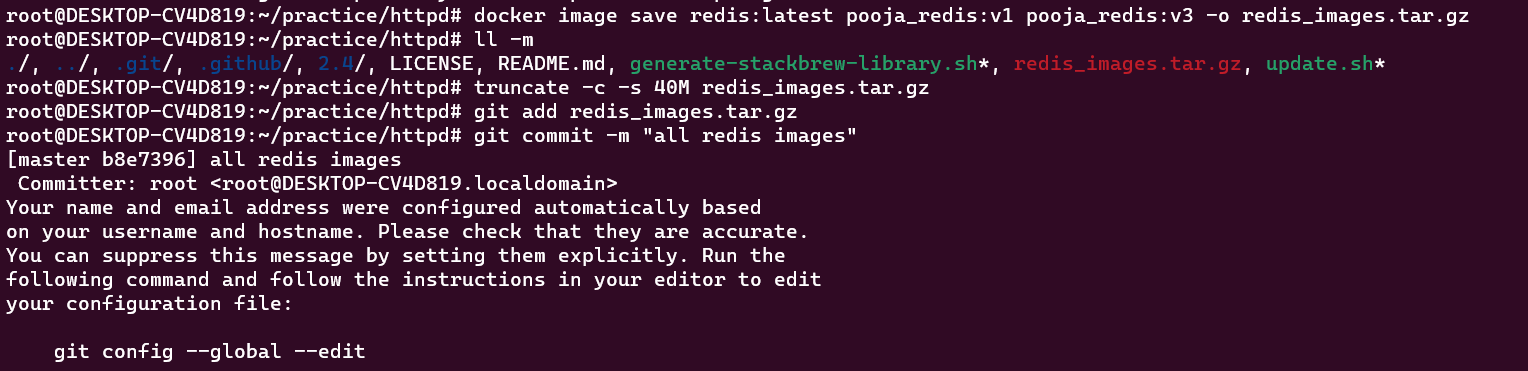
****

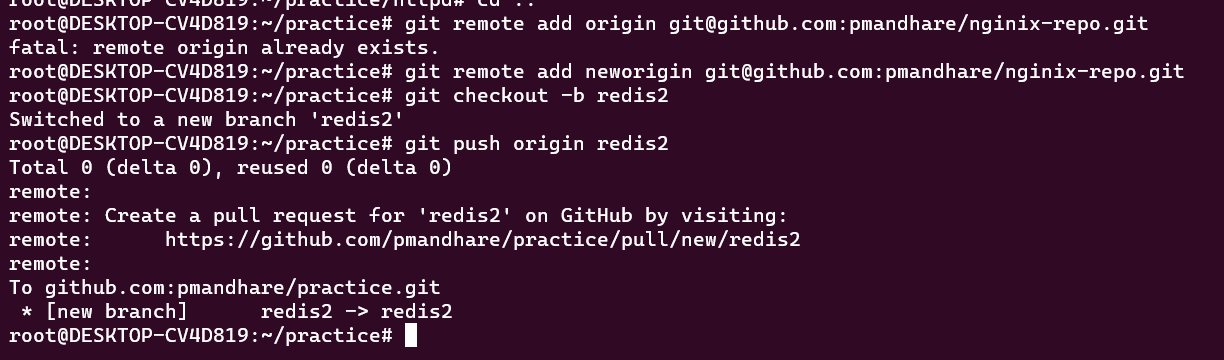
* *Mearge release branch with main branch by creating PR*

****

**10ss. Save all the local redis images in the form of a .tar file in the master branch of your local repository. Then delete all the redis images from the local system. Then push your master branch to remote. Then Load the redis images from the tar file to the local system. Check if all the redis images are loaded properly.**

**Answer:**

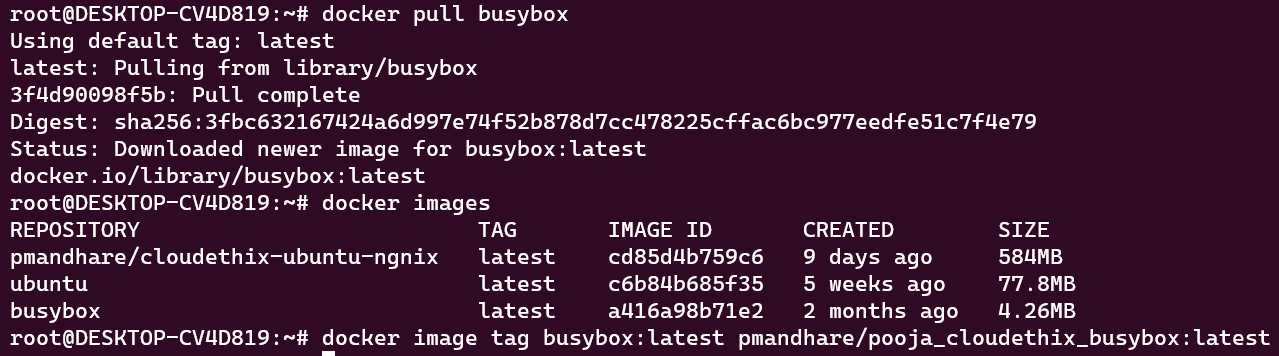
****

****

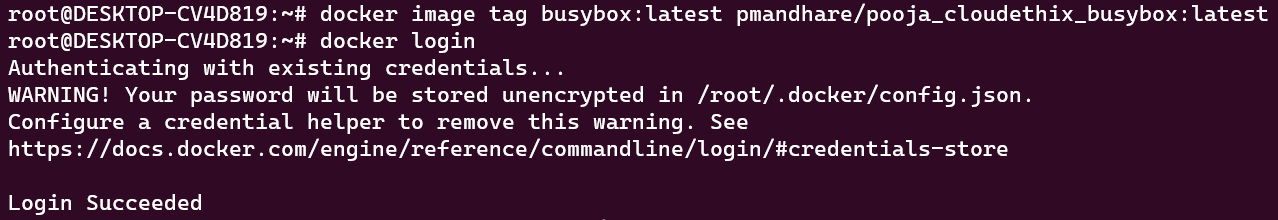
**10.Pull busybox image on your local system. Then tag that image and push it to Docker Hub repository “yourname\_cloudethix\_busybox”. Export docker image from nginx container & create a .tar file then create the docker image by importing the tar file with a meaningful name. Once Image is imported , tag it and push it to the “yourname\_cloudethix\_busybox” repository of docker hub.**

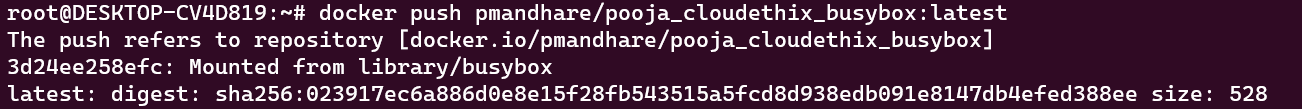
**Answer:**

* We have pulled the busbox image.

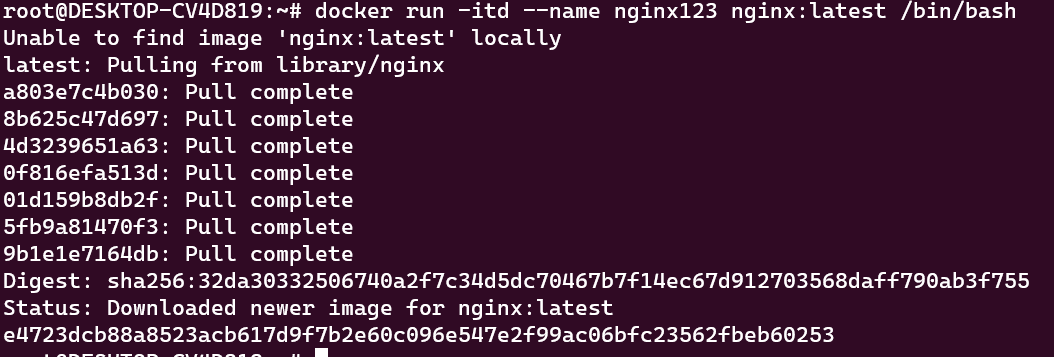
****

* Busybox image is tagged and push to docker hub

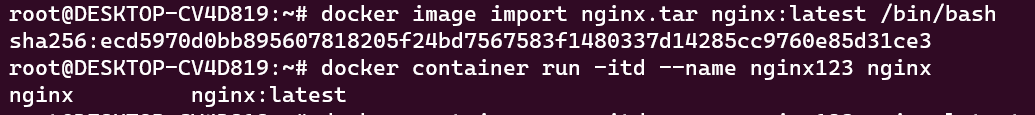
****

****

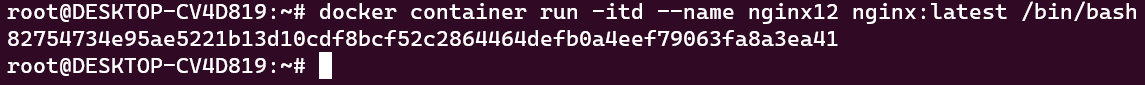
* We have run a container with ngnix image and exported the image in a tar file and imported the image again.

****

****

****

* We are able to run the container

****