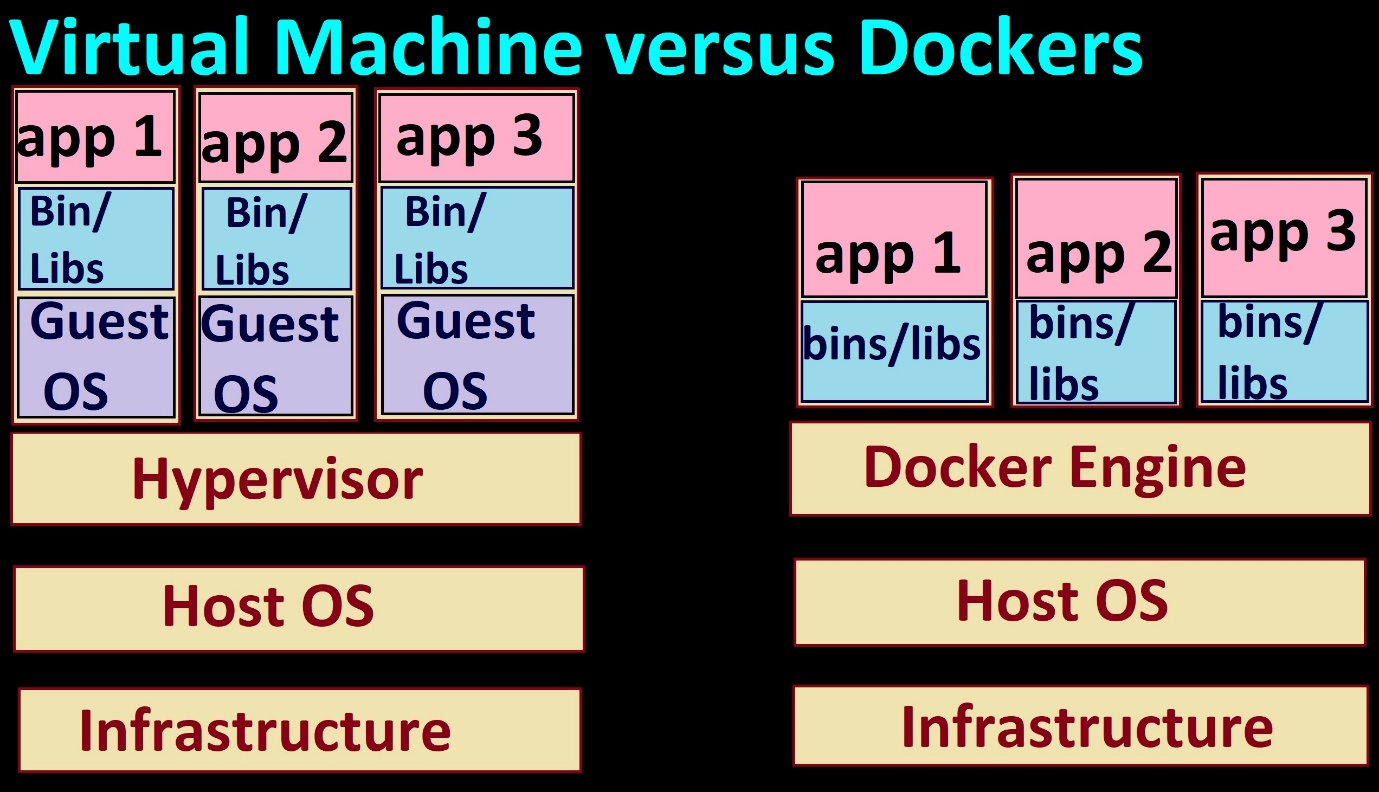
**Docker Introduction**

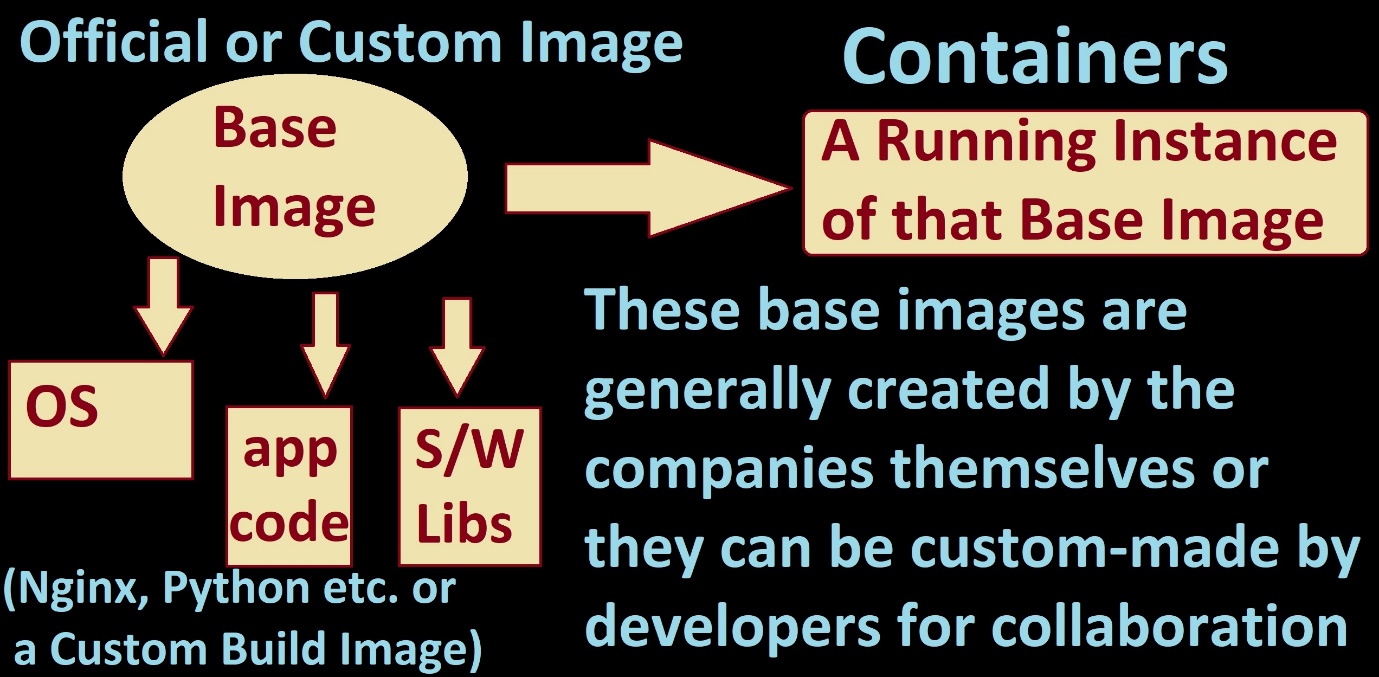
**Docker** is a tool to run your applications in an isolated environment thus removing inconsistencies and making sure your app runs exactly the same irrespective of the underlying infrastructure thus fostering collaboration with other team-members.

**Containers** **&** **Docker V/S VM’s -** Just install the required Binaries and Libraries and Fire up the container in your environment without thinking about virtualization.

VM’s are resource heavy (High amount of Storage, Memory, take minutes to fire up) whereas Docker Containers are light-weight (Less Amount of Storage, memory and fire up in seconds)

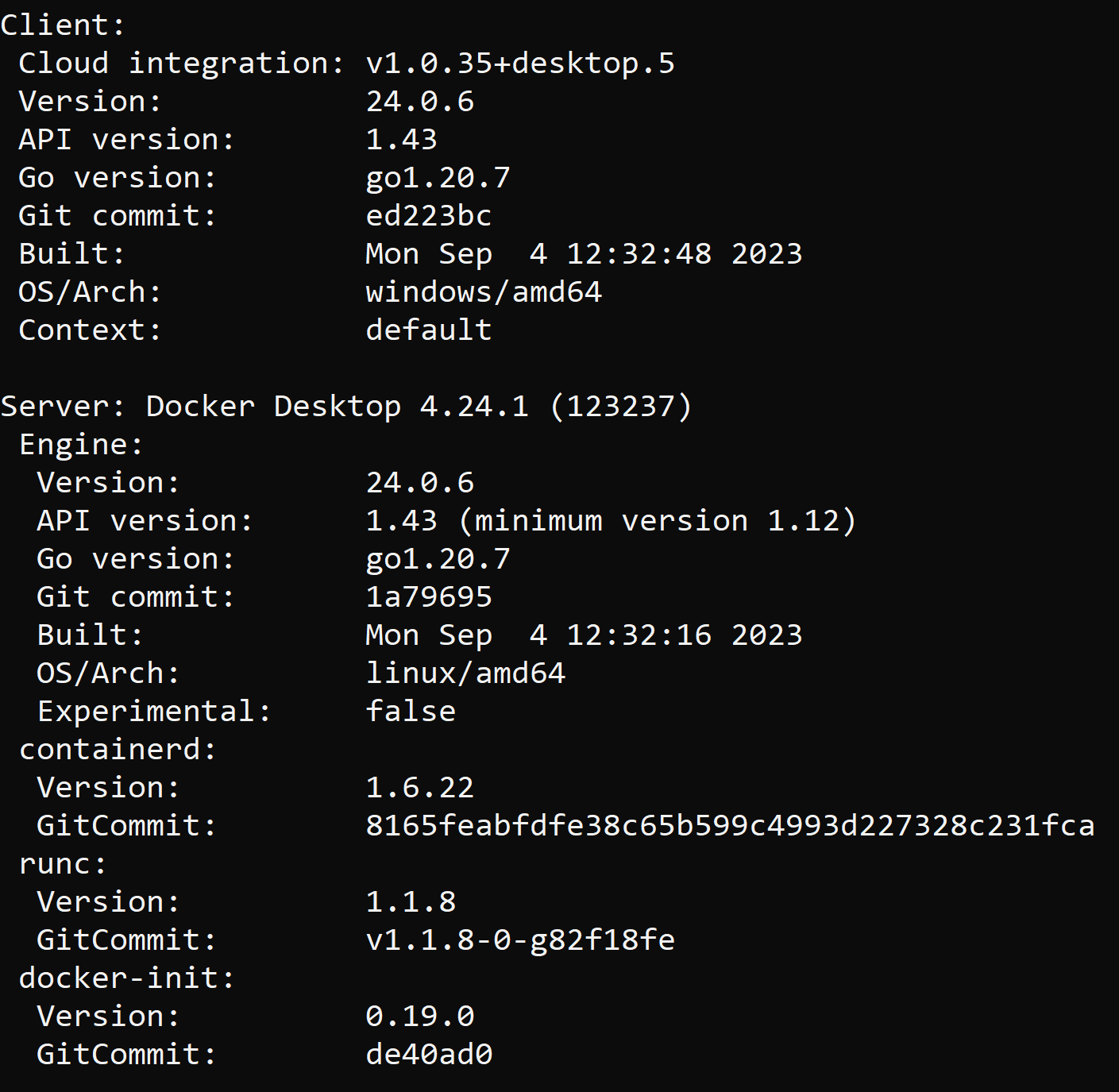


**Base Image is like a Template based on which the Containers are Created.**



# **Install Docker Desktop first**

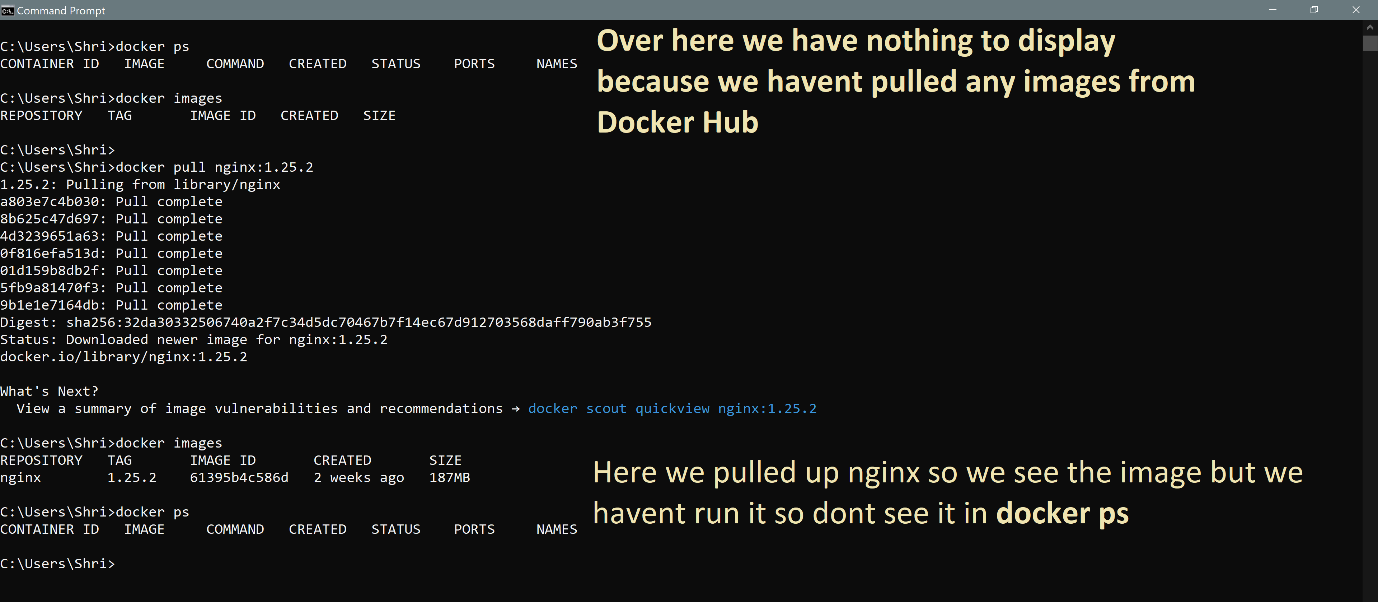
docker version



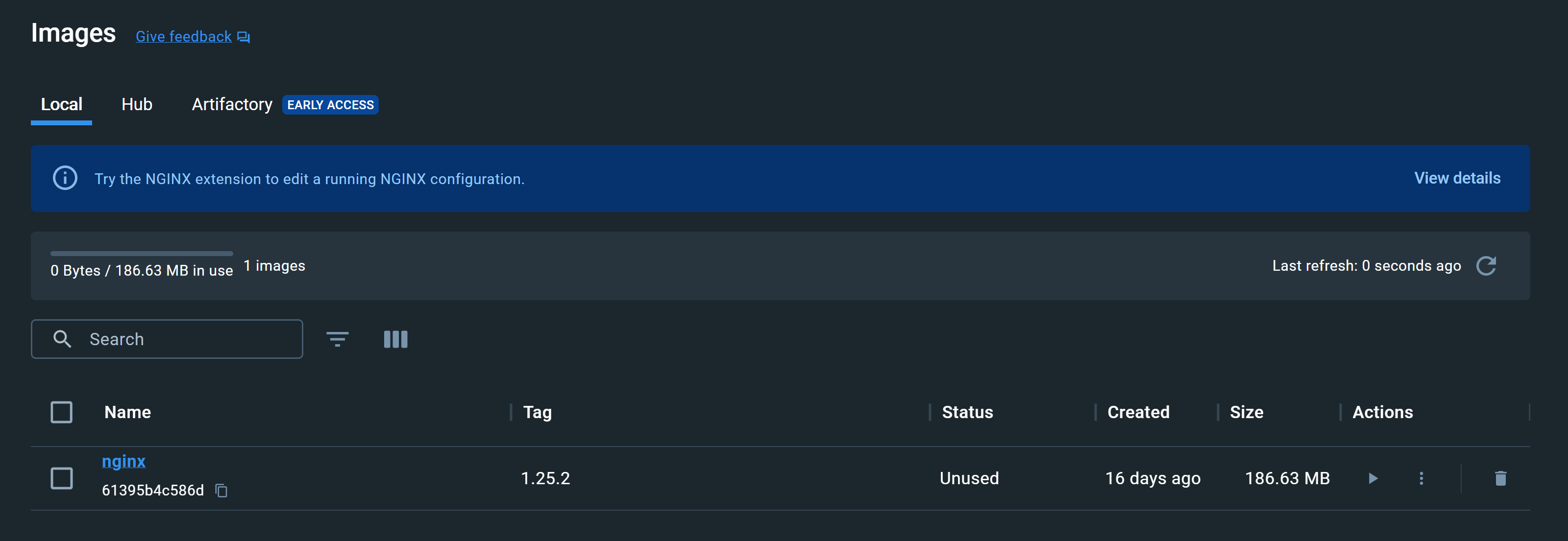
docker images

docker ps

docker pull nginx:1.25.2

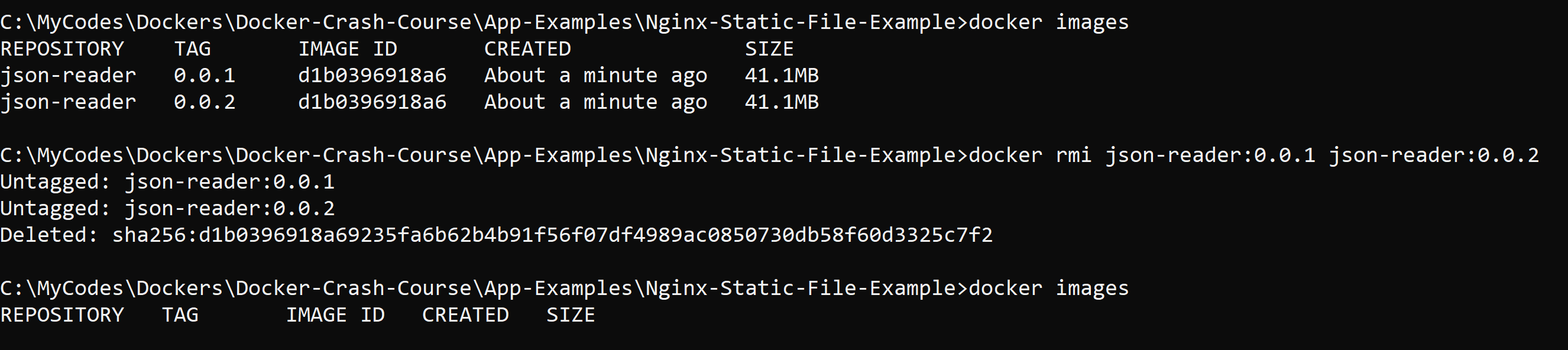


Also visible on your Docker Desktop

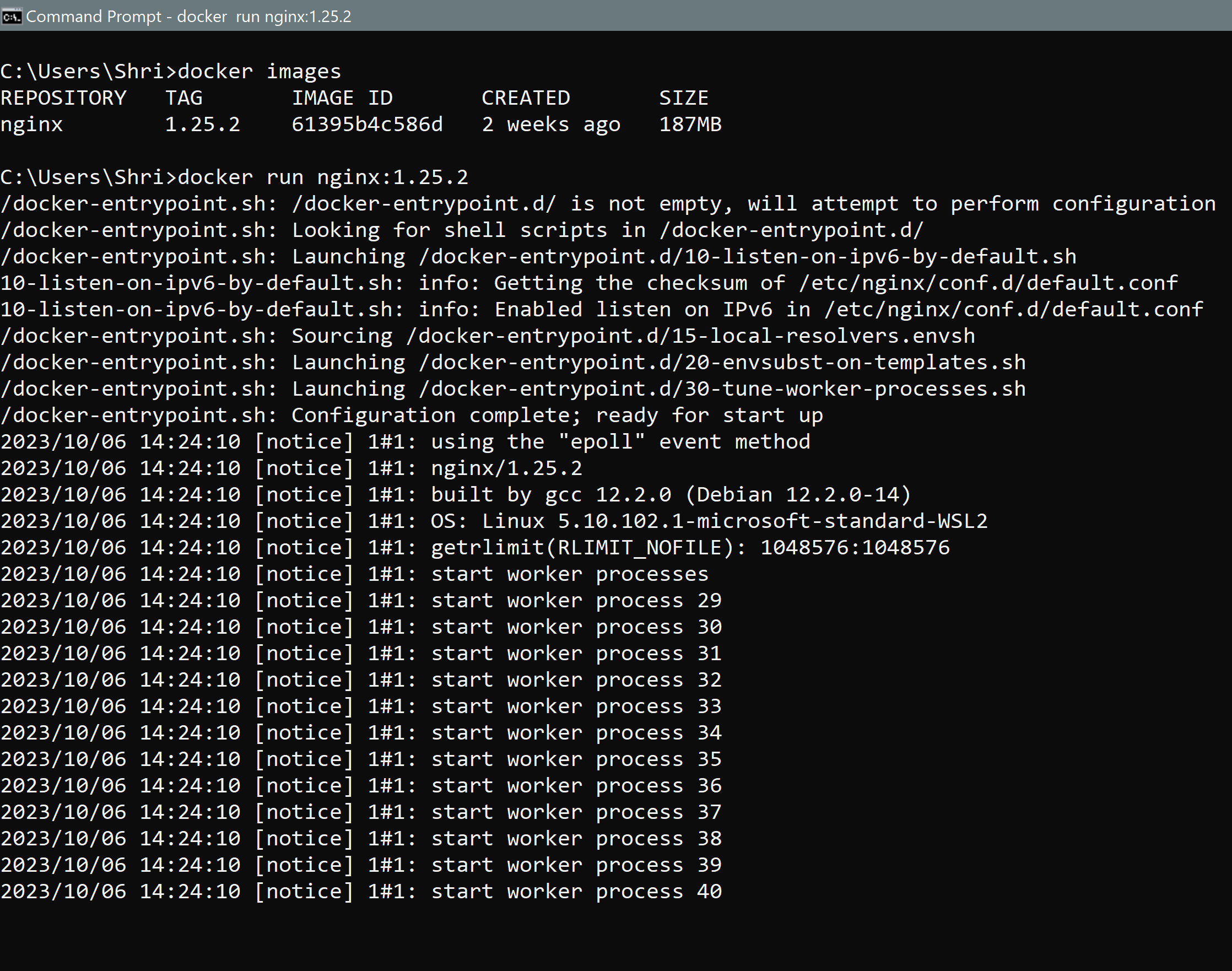


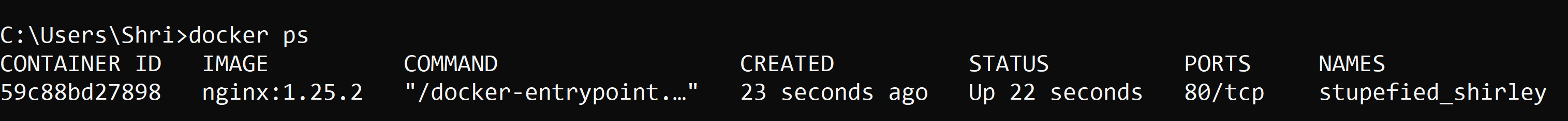
To remove a image use the command:

**docker rmi imagename:tagName**

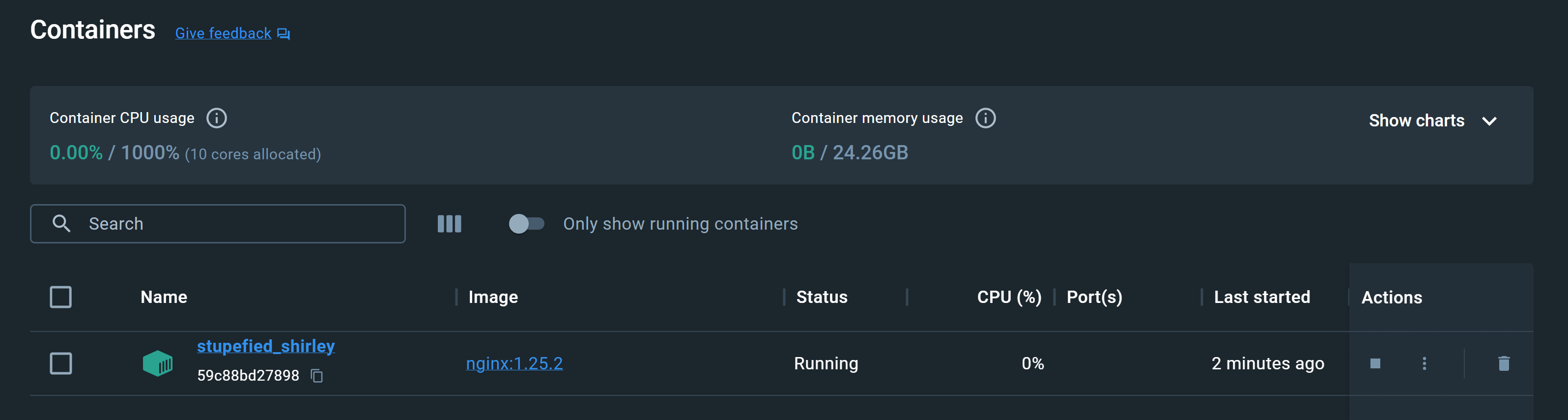


When we run the container, it starts and is visible on docker ps



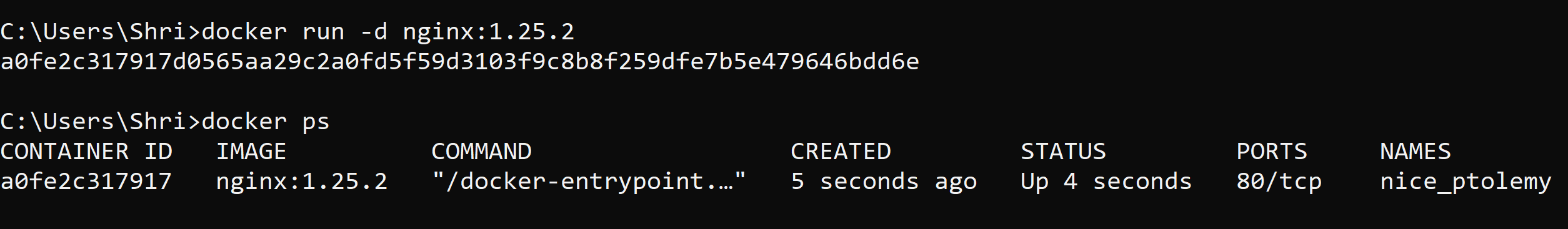


Also visible on Docker Desktop

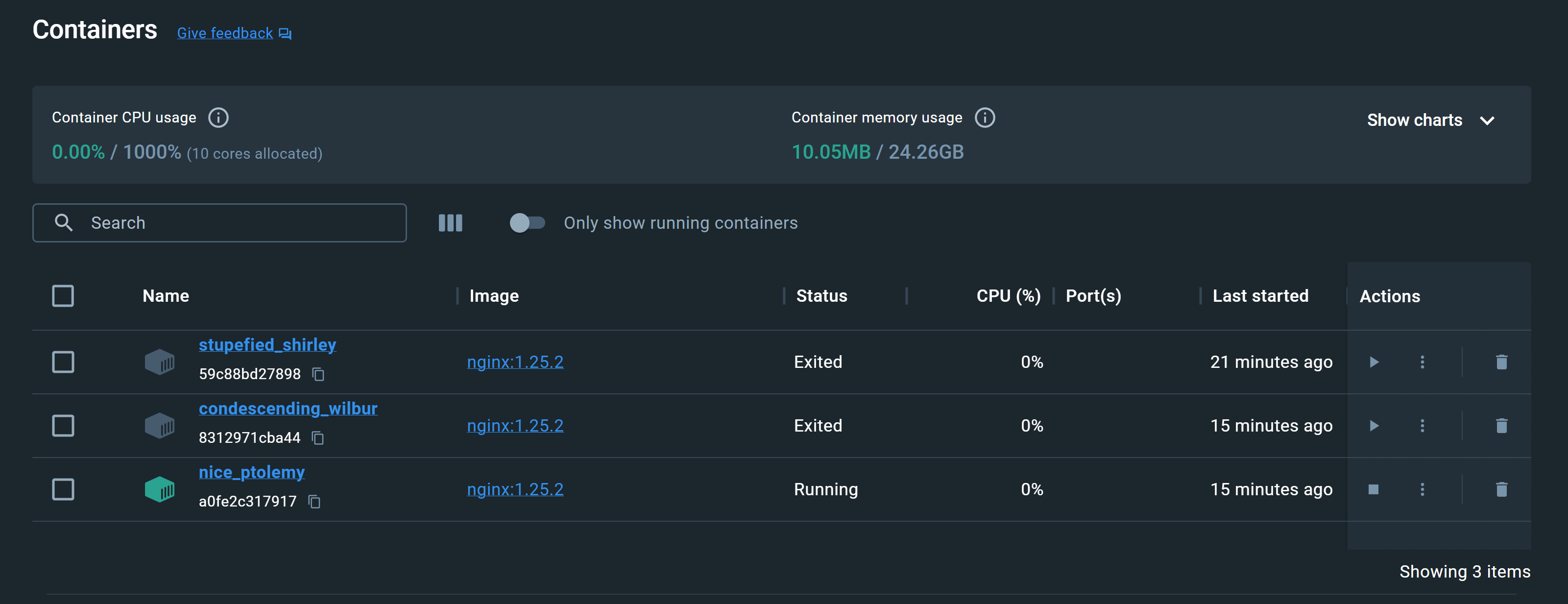


But our terminal is locked so it would make sense for us to start the container in the background so we can use the cmd prompt for anything else we want and not block our command prompt altogether. So we stop the container with Control + C and run the container with the -d or in detached mode

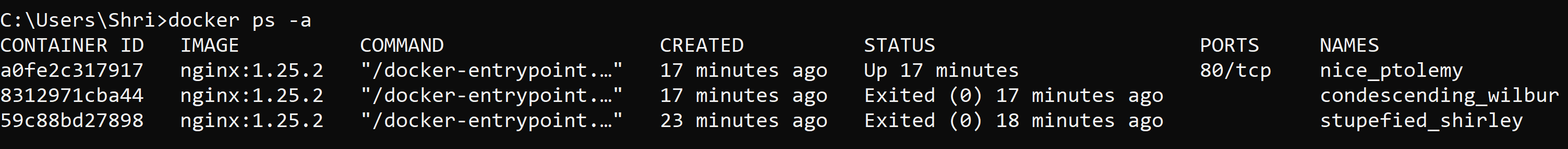
docker run -d nginx:1.25.2



But docker ps shows only the active container, the other closed ones are visible on Docker Desktop:



To get these on the cmd prompt use: docker ps -a

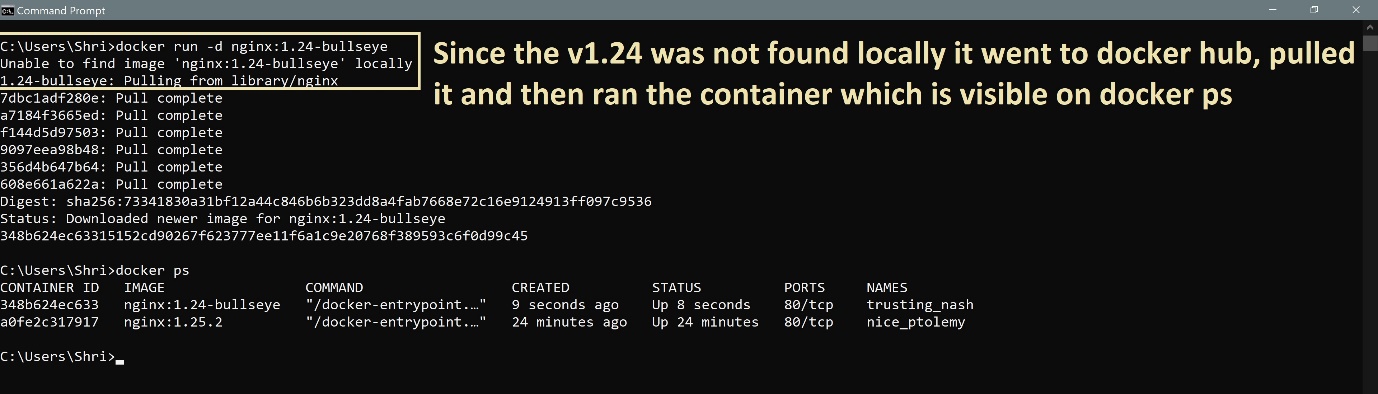


Since it is running in background, still we might need to see the logs for that we use :

docker logs containerID EG. docker logs a0fe2c317917

Now if the image is not available locally then if we run the docker run command directly, it will go to the Docker Hub, pull up the image and you get it running accordingly.

docker run -d nginx:1.24-bullseye



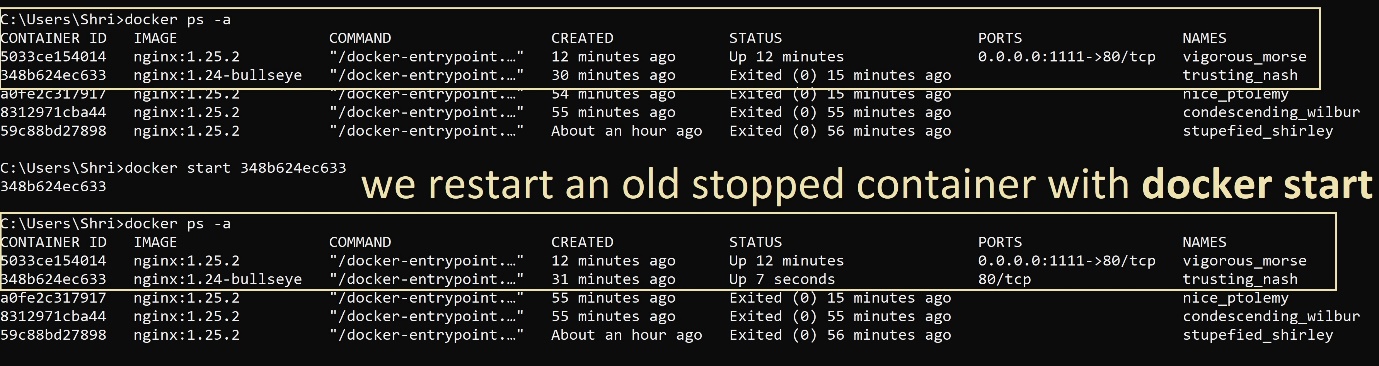
**Thus, it does both pull and run with this one command.**

Docker run starts a new container every time.

We can restart an old stopped container with the docker start command.

docker start containerID

Eg. docker start 348b624ec633

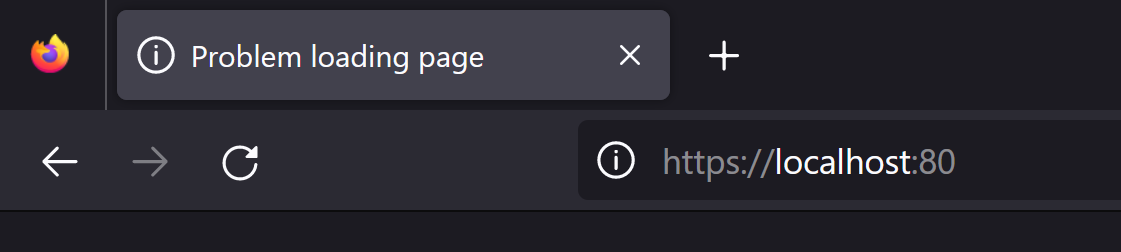


We can start/stop/run the containers with the containerID or containerName:

docker start/stop containerID/containerName

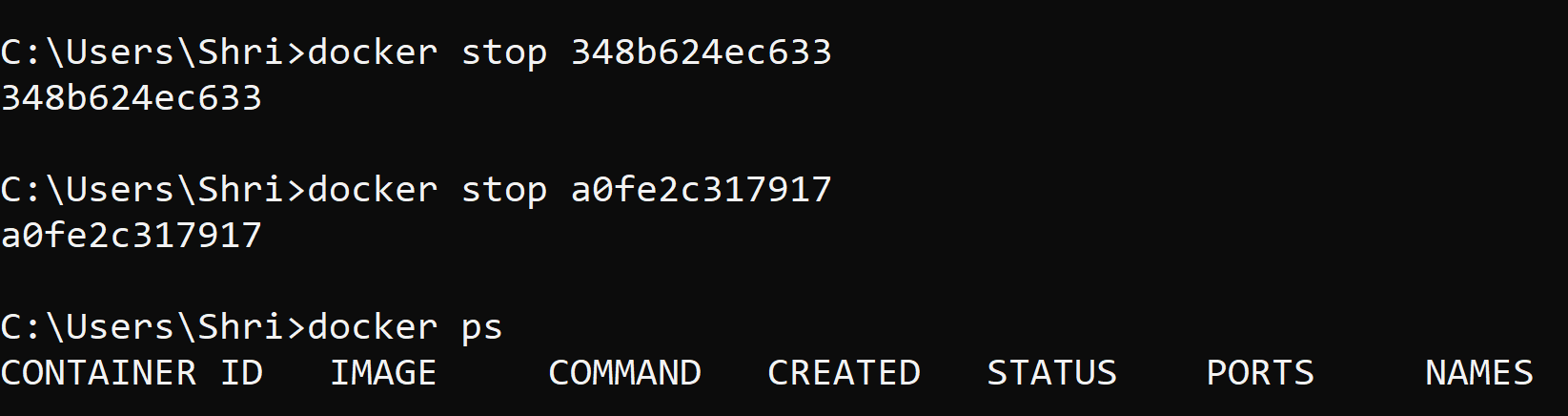


Currently our containers are running in their own docker environment but not exposed to the Client Machine(localhost) to do that we have to do a port binding



**So, we bind the docker’s nginx port 80 to our local port let’s say 1111**

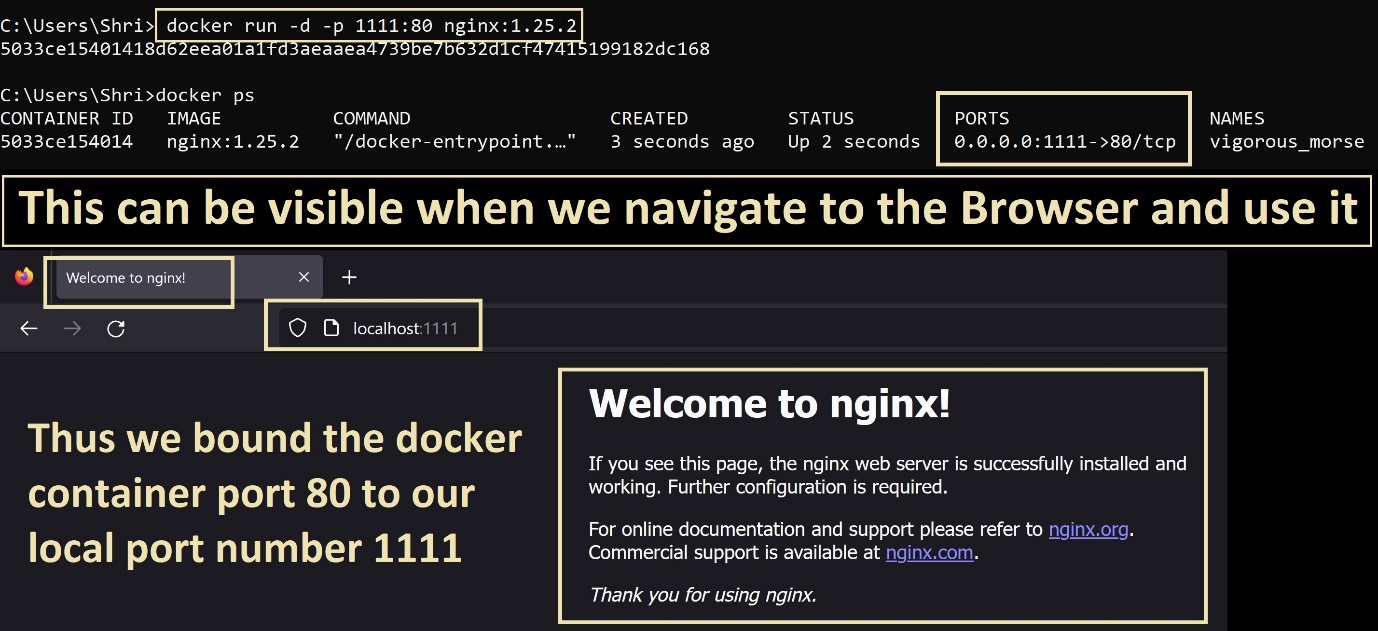
So first we stop the containers and then we restart it with the port binding flag



**docker run -d(detach) -p(publish) localPortNumber:containerPortNumber imageName**

Eg. docker run -d -p 1111:80 nginx:1.25.2

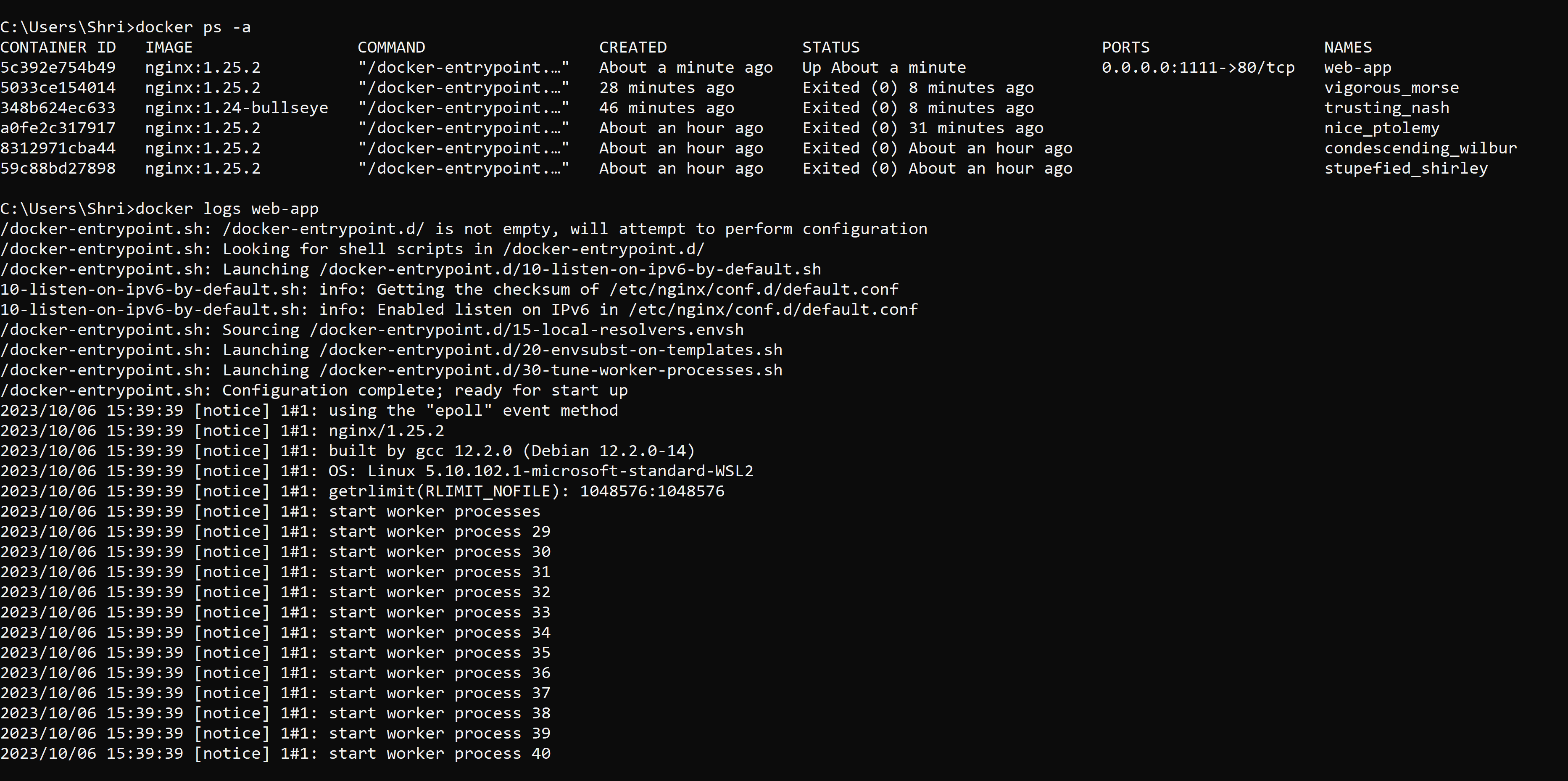
Let’s check on the Browser and it is also visible on Port option in docker ps:



The docker run commands creates auto-generated names for the containers but if we want to have a proper name for our containers, we can use the - - name flag:

Eg. we call our container as web-app like

docker run -d --name web-app -p 1111:80 nginx:1.25.2

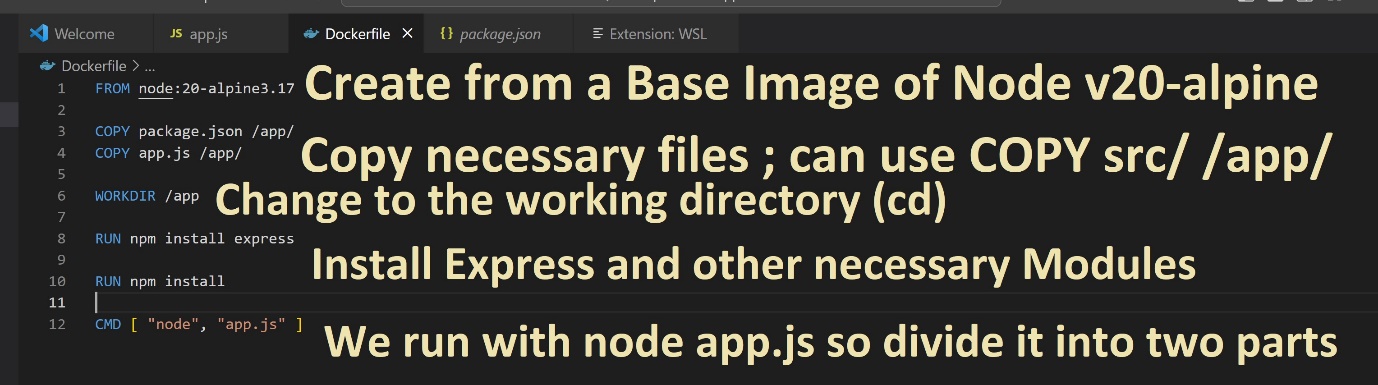


Create Docker File based on a Base Image

So here we choose Node JS cause our App is a simple Node App

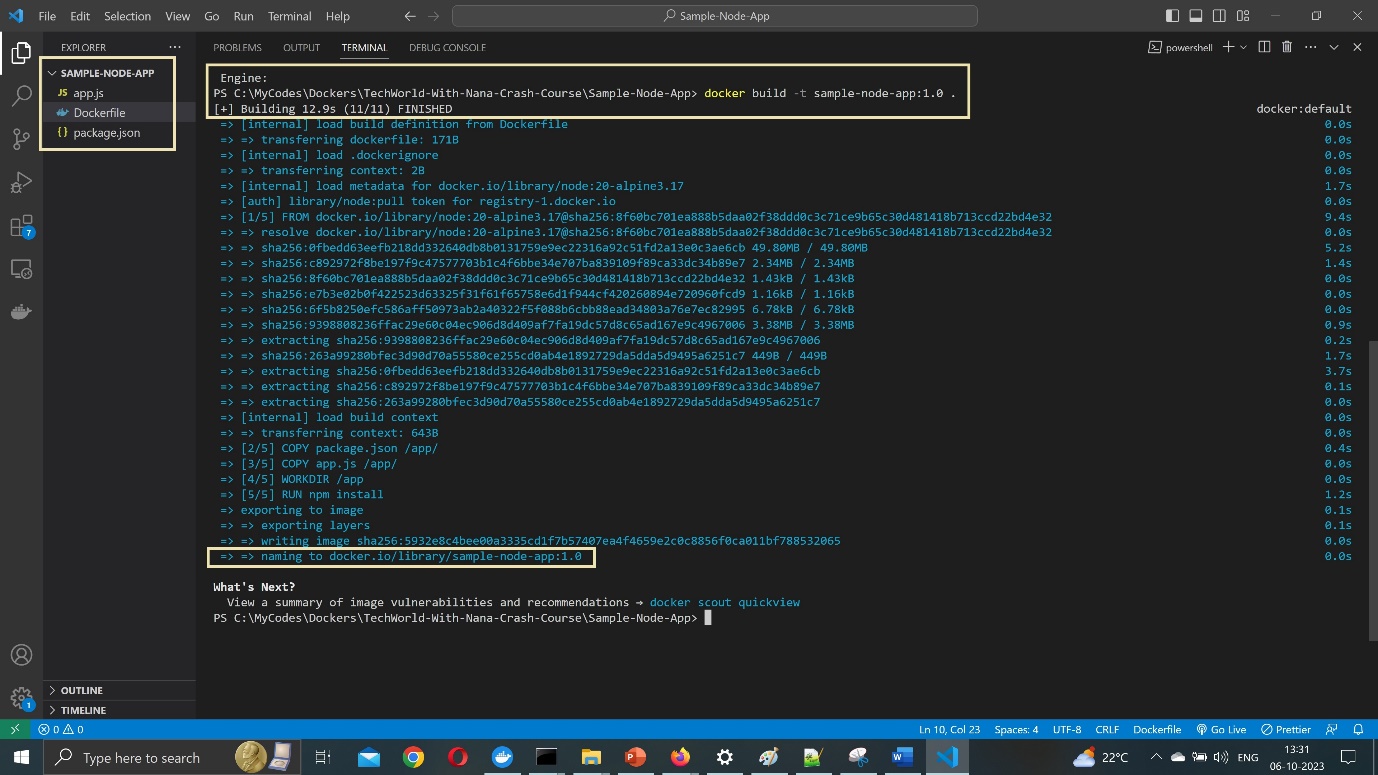
Using the node base image makes sure you have all the necessary software’s installed and ready to use like Node, NPM etc.

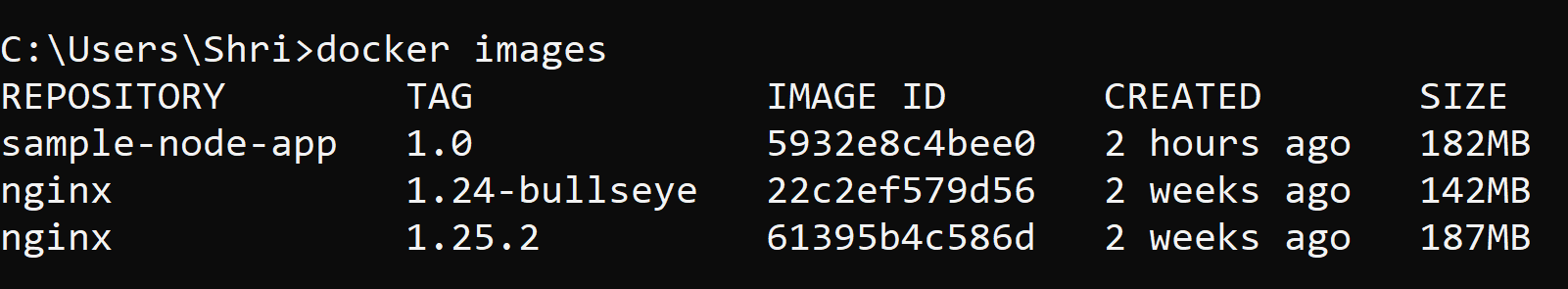
**Name file as Dockerfile and not DockerFile [ f should be small ]**



docker build -t sample-node-app:1.0 .

**(. Because we are in the same Directory)**

****

Also visible in docker images:

**We can create multiple versions and run it by using different ports:**



**You can delete unused Docker Containers with the command:**

**docker rm containerID/containerName**



**Free Space taken up by Docker**

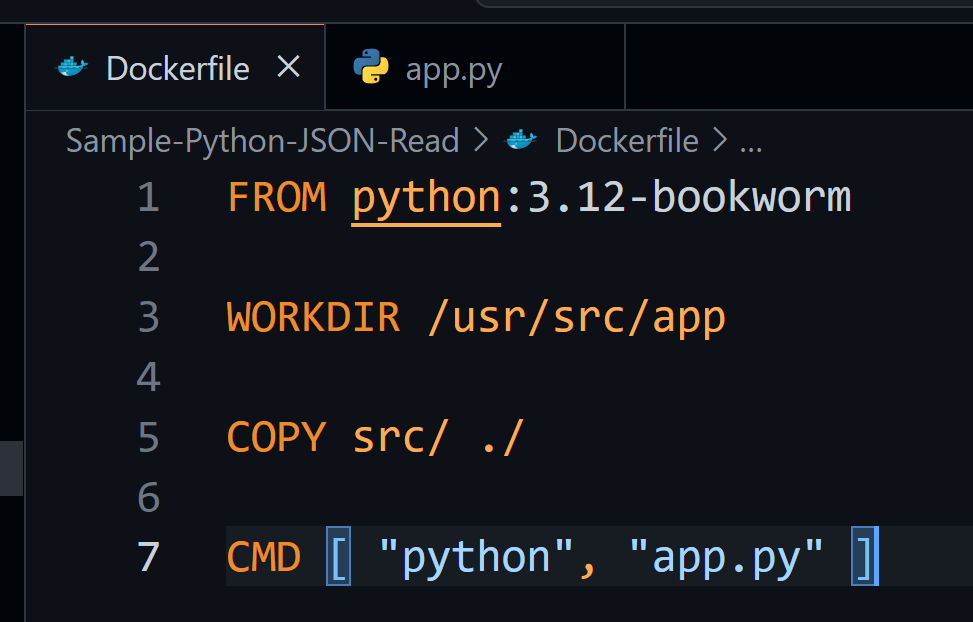
**https://stackoverflow.com/questions/37518096/how-can-i-reduce-the-disk-space-used-by-docker**

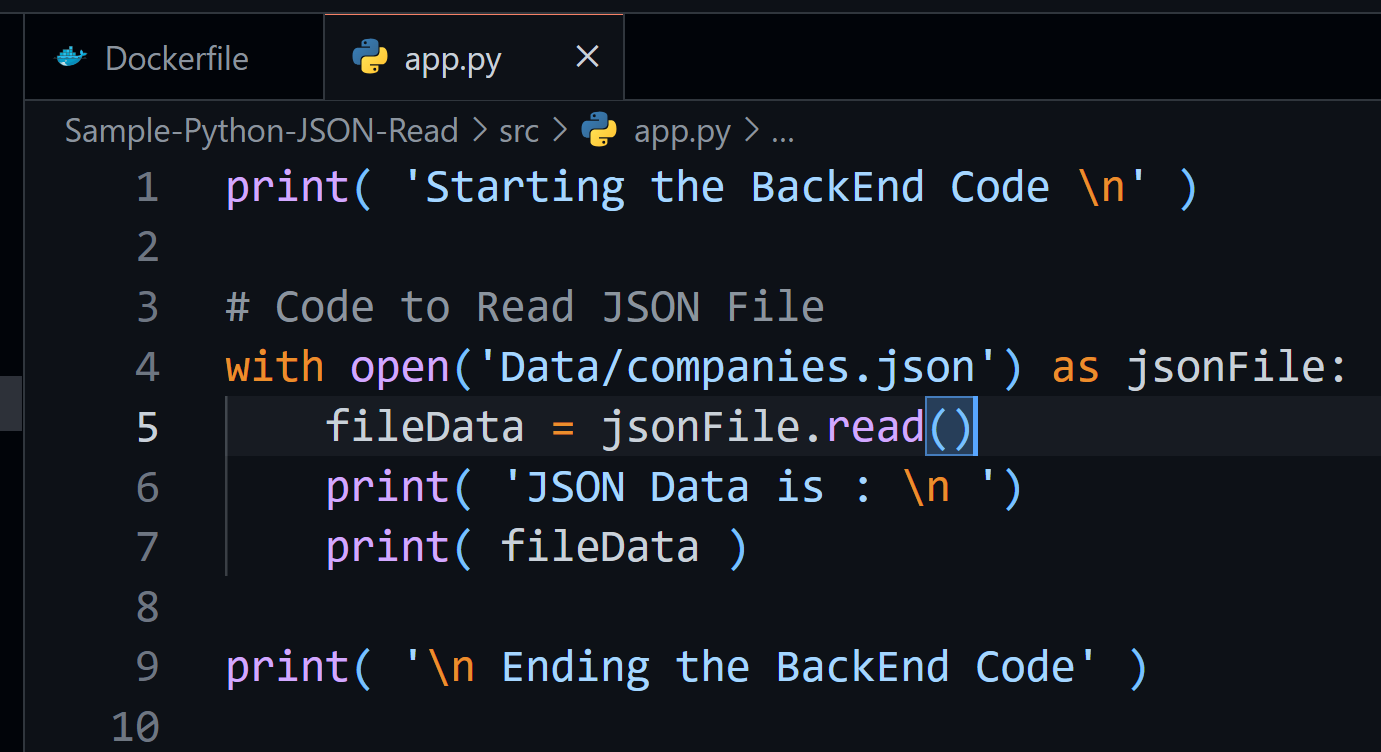
**Deploy and Push Containers to Docker Repository**

**Step 0 –** login to the Docker Account using the command

**docker login -u dockerUserName Eg.** docker login -u rohandeshpande5

Step 1 – **Create a proper Dockerfile and write the appropriate code**

****

****

Step 2 - **docker build -t local-image:tagname .**

Step 3 - **docker tag local-image:tagname dockerUserName/local-image:tagname**

Step 4 - **docker push dockerUserName/local-image:tagname**

Eg.

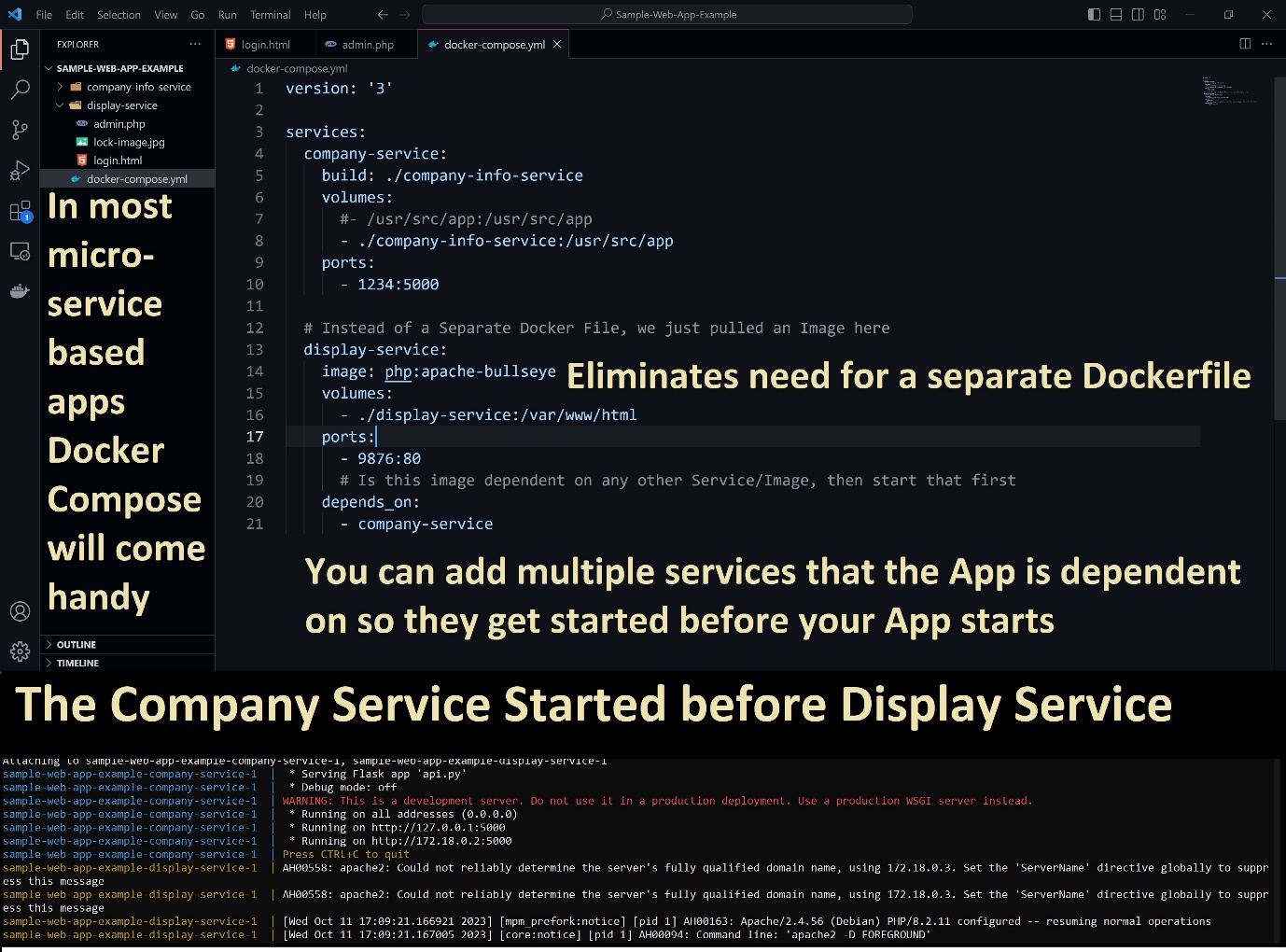
docker build -t json-reader:0.0.2 .

docker tag json-reader:0.0.2 rohandeshpande5/json-reader:0.0.2

docker push rohandeshpande5/json-reader:0.0.2

**Docker Compose**

**Base file created for all the necessary services or micro-services to run and with one command start all the necessary containers**

****