**1. Write a command to pull nginx base image on your machine and run it. You should be able to access the web server on port 1234**

Pulled and run the official Nginx Docker image (nginx:latest), using the command:

bash

**docker run -d -p 1234:80 --name nginx-container nginx**

-d → Detached mode

* Runs the container in the background.
* You get the container ID in your terminal, and it continues running without tying up your session.

-p 1234:80 → Port mapping

* Maps port 1234 on your host to port 80 inside the container.
* Since Nginx listens on port 80 by default, this lets you access.

This command starts the container in detached mode, maps port 1234 on the host to port 80 in the container (Nginx’s default HTTP port), and names the container nginx-container.

**CLI logs:**

shubham.sahu@IN-IT18554 ~ % docker version

Client:

Version: 28.0.4

API version: 1.48

Go version: go1.23.7

Git commit: b8034c0

Built: Tue Mar 25 15:06:09 2025

OS/Arch: darwin/arm64

Context: default

shubham.sahu@IN-IT18554 ~ % docker ps

Cannot connect to the Docker daemon at unix:///var/run/docker.sock. Is the docker daemon running?

shubham.sahu@IN-IT18554 ~ % docker ps

CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES

shubham.sahu@IN-IT18554 ~ % **docker run -d -p 1234:80 nginx**

Unable to find image 'nginx:latest' locally

latest: Pulling from library/nginx

c96c7b918bd5: Pull complete

16c9c4a8e9ee: Pull complete

450968563955: Pull complete

de29066b274e: Pull complete

9b14c47aa231: Pull complete

fd8a9ced9846: Pull complete

2cf157fc31fe: Pull complete

Digest: sha256:5ed8fcc66f4ed123c1b2560ed708dc148755b6e4cbd8b943fab094f2c6bfa91e

Status: Downloaded newer image for nginx:latest

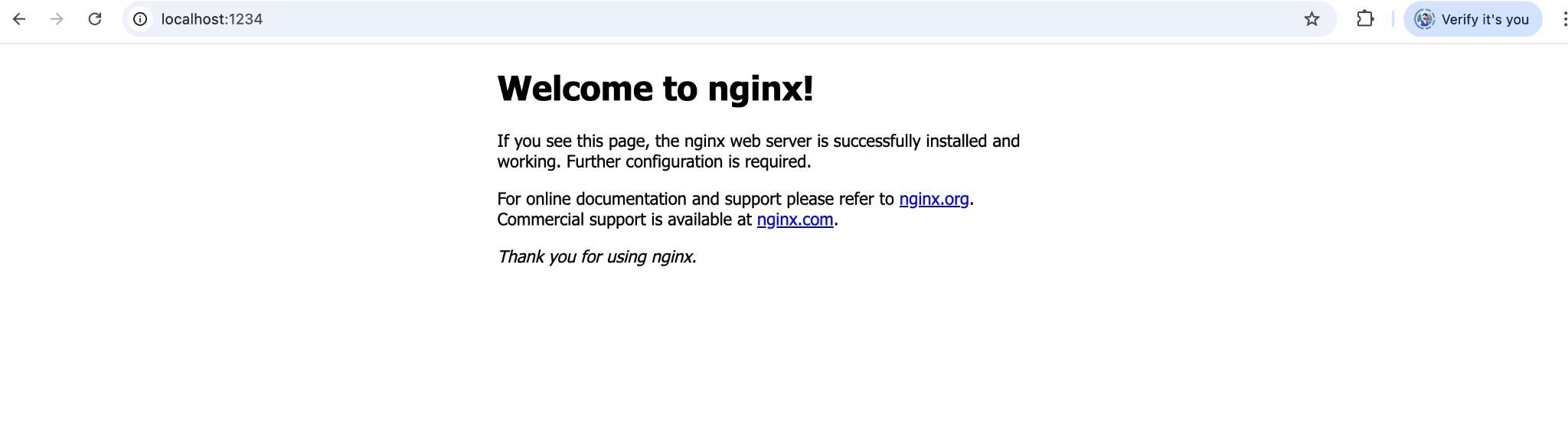
85e93484342efd43cae959bec1b8104b908285c03b47ee5ed96bb9bb77f1f4ad

shubham.sahu@IN-IT18554 ~ % docker ps

CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES

85e93484342e nginx "/docker-entrypoint.…" 32 seconds ago Up 31 seconds 0.0.0.0:1234->80/tcp kind\_jennings

shubham.sahu@IN-IT18554 ~ %



2. Create a docker volume with your name (for e.g. shubham-vol) and ensure it is created successfully. Then run a container and map the this volume

**a. Image: nginx:latest**

**b. Source: shubham-vol, Destination: /app**

**c. Login to the volume and create a text file - sample.txt**

### **Step 1 : Create a Docker volume**

**docker volume create shubham-vol**

**docker volume ls**

**DRIVER VOLUME NAME**

**local shubham-vol**

shubham.sahu@IN-IT18554 ~ % docker volume create shubham-vol

shubham-vol

shubham.sahu@IN-IT18554 ~ % docker volume ls

DRIVER VOLUME NAME

local shubham-vol

shubham.sahu@IN-IT18554 ~ %

**Step 2: Run an Nginx container and map the volume**

docker run -d \

--name nginx-with-volume \

-p 1234:80 \

-v shubham-vol:/app \

nginx:latest

* Maps the volume shubham-vol to the container's /app directory.
* Runs Nginx on port 1234.

**Step 3**: Create a file (sample.txt) inside the volume  
  
docker exec -it nginx-with-volume /bin/bash

shubham.sahu@IN-IT18554 ~ % docker run -d \

--name nginx-with-volume \

-p 1234:80 \

-v shubham-vol:/app \

nginx:latest

d49f2549116f77c97c362761f0bbd639438afb33de2b9355e7fae2f7a35beac1

shubham.sahu@IN-IT18554 ~ % docker exec -it nginx-with-volume /bin/bash

root@d49f2549116f:/# cd /app

echo "This is a sample file" > sample.txt

ls -l

total 4

-rw-r--r-- 1 root 07:26 sample.txt

root@d49f2549116f:/app# ls

sample.txt

root@d49f2549116f:/app#

**3. Delete the container and check whether your volume still exists. Repeat step number 2 and check whether you can see sample.txt inside container.**

### **Step 1: Delete the container**

Stop and remove the container:

docker stop nginx-with-volume

docker rm nginx-with-volume

**Step 2: Check if the volume still exists**

Run:

docker volume ls

DRIVER VOLUME NAME

local shubham-vol

### **Step 3: Re-run the container with the same volume**

docker run -d \

--name nginx-with-volume \

-p 1234:80 \

-v shubham-vol:/app \

Nginx:latest

**Step 4: Check if sample.txt still exists inside the container**

docker exec -it nginx-with-volume /bin/bash

cd /app

ls -l

You should see:

-rw-r--r-- 1 root root 24 Apr 23 12:00 sample.txt

Confirms the Docker **volume persists independently** of the container.

**Logs:**

root@3d93e5a53333:/app# shubham.sahu@IN-IT18554 ~ % docker run --rm -v shubham-vol:/app alpine ls /app

sample.txt

shubham.sahu@IN-IT18554 ~ % docker volume ls

DRIVER VOLUME NAME

local shubham-vol

shubham.sahu@IN-IT18554 ~ % docker run -d \

--name nginx-with-volume \

-p 1234:80 \

-v shubham-vol:/app \

nginx:latest

3d93e5a533339fd8400abb314fd12438a19686d697d3f24e555b6e860d119b96

shubham.sahu@IN-IT18554 ~ % docker exec -it nginx-with-volume /bin/bash

cd /app

root@3d93e5a53333:/app# ls -l

total 4

-rw-r--r-- 1 root 07:26 sample.txt

root@3d93e5a53333:/app# ls

sample.txt

root@3d93e5a53333:/app#

**4. Achieve the same thing using docker-compose**

**a. Map shubham-vol inside your container with image - nginx:latest**

**b. Create a volume inside docker-compose file and map it to the container with same image.**

version: '3.9'

services:

nginx:

image: nginx:latest

container\_name: nginx-compose

ports:

- "1234:80"

volumes:

- shubham-vol:/app # Map the named volume to /app in the container

volumes:

shubham-vol:

external: true

**Logs**

shubham.sahu@IN-IT18554 Desktop % **mkdir docker-nginx-volume**

shubham.sahu@IN-IT18554 Desktop % **cd docker-nginx-volume**

shubham.sahu@IN-IT18554 docker-nginx-volume % ls

docker-compose.yml

shubham.sahu@IN-IT18554 docker-nginx-volume % **cat docker-compose.yml**

version: '3.9'

services:

nginx:

image: nginx:latest

container\_name: nginx-compose

ports:

- "1234:80"

volumes:

- shubham-vol:/app # Map the named volume to /app in the container

volumes:

shubham-vol:

external: true

shubham.sahu@IN-IT18554 docker-nginx-volume % **docker-compose up -d**

[+] Running 1/1

✔ Container nginx-compose Started 0.1s

shubham.sahu@IN-IT18554 docker-nginx-volume % docker ps

CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES

181ea5f27c1a nginx:latest "/docker-entrypoint.…" 9 seconds ago Up 9 seconds 0.0.0.0:1234->80/tcp nginx-compose

shubham.sahu@IN-IT18554 docker-nginx-volume % docker container ls

CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES

181ea5f27c1a nginx:latest "/docker-entrypoint.…" 19 seconds ago Up 18 seconds 0.0.0.0:1234->80/tcp nginx-compose

shubham.sahu@IN-IT18554 docker-nginx-volume % docker volume ls

DRIVER VOLUME NAME

local shubham-vol

**5. Write Dockerfile for sample node.js application. - Make use of ENTRYPOINT in Dockerfile to start multiple services at the start of container. (Services like - cron, sshd) - Install openssh in container and take ssh of container from other host.**

This task involves creating a custom Docker image for a sample Node.js application that also runs background services like cron and OpenSSH.

We use a Dockerfile to install dependencies and configure services, and an ENTRYPOINT script to launch everything when the container starts. After building the image, we push it to Docker Hub, and then reuse it to create a new container on any host. SSH access is enabled to remotely connect into the running container for debugging or management.

shubham.sahu@IN-IT18554 sample-node-app % **docker build -t node-ssh-cron-app .**

[+] Building 34.9s (12/12) FINISHED docker:desktop-linux

=> [internal] load build definition from Dockerfile 0.0s

=> => transferring dockerfile: 586B 0.0s

=> [internal] load metadata for docker.io/library/node:18 3.0s

=> [auth] library/node:pull token for registry-1.docker.io 0.0s

=> [internal] load .dockerignore 0.0s

=> => transferring context: 2B 0.0s

=> [1/6] FROM docker.io/library/node:18@sha256:df9fa4e0e39c9b97e30240b5bb1d99bdb861573a82002b2c52ac7d6b8d6d773e 24.7s

=> => resolve docker.io/library/node:18@sha256:df9fa4e0e39c9b97e30240b5bb1d99bdb861573a82002b2c52ac7d6b8d6d773e 0.0s

=> => sha256:43b3ca1db9e3b5c7786b79ce7f1c2ad33a01f5c3934459c86fe5a914cf54cefb 448B / 448B 0.3s

=> => sha256:71daa2c787b0984bbf3b93b60686fc9fe305d28e833914019b2745ab9f36730e 48.33MB / 48.33MB 12.8s

=> => sha256:e171895483c62ea3506398c39c42c8ad41df28591b94facd18ba90fad2573b36 1.25MB / 1.25MB 1.6s

=> => sha256:62cad2f6aff7af926b513c25c5e2932eb6adcf52272801161e3dc7b88d3ec7ef 45.73MB / 45.73MB 6.7s

=> => sha256:0e3cee1fc214f8bfbdeb8e88f9154354b2419758c1c408d829aca79b3c85b043 3.33kB / 3.33kB 1.0s

=> => sha256:002e18bd5659ca9d155e99922678788bec836a3ac4964d8a9567ce59e2154de9 201.33MB / 202.74MB 30.5s

=> => sha256:ebf144460616b42eb1462fd80a5e1909e578b1e1f7285b185e468ba2b01308b9 64.36MB / 64.36MB 11.6s

=> => sha256:9d81c64672754c46e5d99e385c8f3283bec2060a79ad7dacdb2f5ce904caa401 22.02MB / 23.54MB 25.0s

=> => extracting sha256:71daa2c787b0984bbf3b93b60686fc9fe305d28e833914019b2745ab9f36730e 0.6s

=> => extracting sha256:9d81c64672754c46e5d99e385c8f3283bec2060a79ad7dacdb2f5ce904caa401 0.2s

=> => extracting sha256:ebf144460616b42eb1462fd80a5e1909e578b1e1f7285b185e468ba2b01308b9 0.7s

=> => extracting sha256:002e18bd5659ca9d155e99922678788bec836a3ac4964d8a9567ce59e2154de9 2.1s

=> => extracting sha256:0e3cee1fc214f8bfbdeb8e88f9154354b2419758c1c408d829aca79b3c85b043 0.0s

=> => extracting sha256:62cad2f6aff7af926b513c25c5e2932eb6adcf52272801161e3dc7b88d3ec7ef 0.6s

=> => extracting sha256:e171895483c62ea3506398c39c42c8ad41df28591b94facd18ba90fad2573b36 0.0s

=> => extracting sha256:43b3ca1db9e3b5c7786b79ce7f1c2ad33a01f5c3934459c86fe5a914cf54cefb 0.0s

=> [internal] load build context 0.0s

=> => transferring context: 1.34kB 0.0s

=> [2/6] RUN apt-get update && apt-get install -y openssh-server cron 5.2s

=> [3/6] RUN mkdir /var/run/sshd && echo 'root:root' | chpasswd && sed -i 's/#PermitRootLogin prohibit-password/PermitRootLogin yes/' /etc/ssh/sshd\_config 0.1s

=> [4/6] WORKDIR /usr/src/app 0.0s

=> [5/6] COPY . . 0.0s

=> [6/6] RUN npm install 0.3s

=> exporting to image 1.4s

=> => exporting layers 1.1s

=> => exporting manifest sha256:228a26cb44f000992e958beb3887e083b721ac2aeae2f9084c3bc069199e86f7 0.0s

=> => exporting config sha256:f050153ce9603eb979a3bd4296c5816d8d155cdda36d695d3d3ce266a545c94f 0.0s

=> => exporting attestation manifest sha256:e01f85634fe71339d98e8912145a8e4b438d8cefc4ab76fddffb8b32f6b538b6 0.0s

=> => exporting manifest list sha256:c5a418a594e7e7ad7786040afda3a4bc96d7adc6bb9880e9062bc46ae2682db0 0.0s

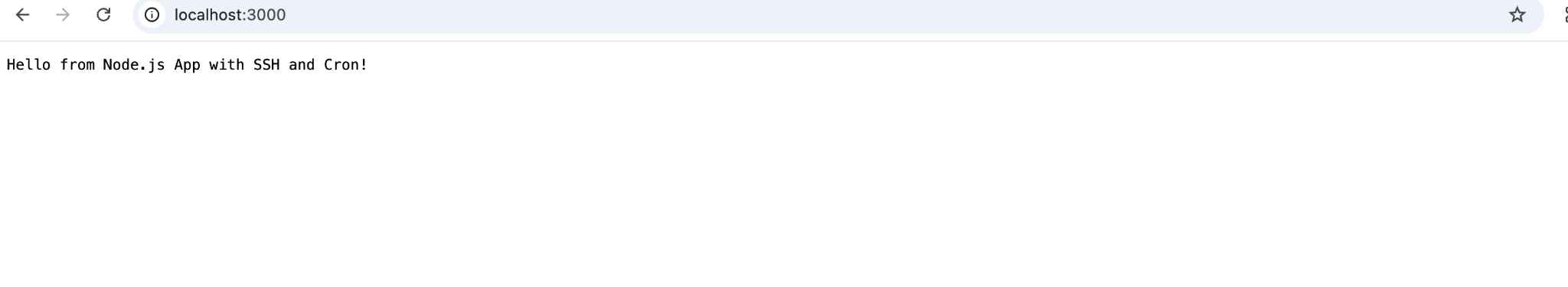
=> => naming to docker.io/library/node-ssh-cron-app:latest 0.0s

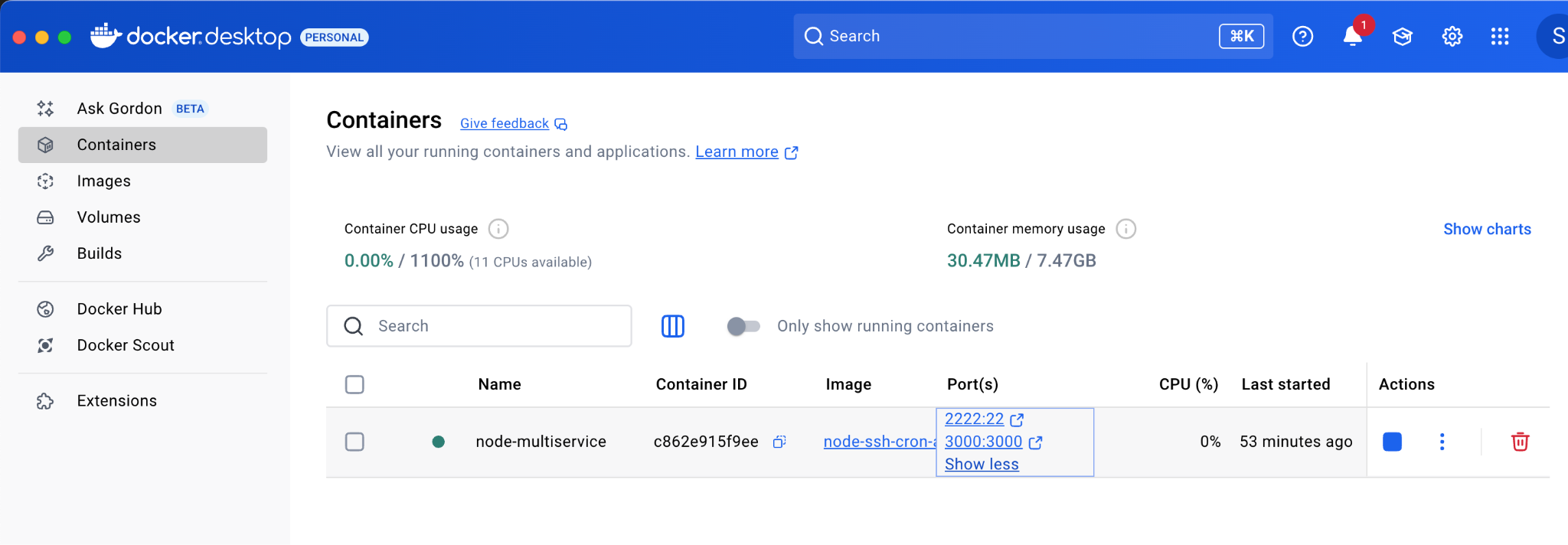
=> => unpacking to docker.io/library/node-ssh-cron-app:latest 0.3s

View build details: docker-desktop://dashboard/build/desktop-linux/desktop-linux/x2xay9ljxb8g7cb5113r36qd9

shubham.sahu@IN-IT18554 sample-node-app % **docker run -d -p 3000:3000 -p 2222:22 --name node-multiservice node-ssh-cron-app**

c862e915f9eef597d7928ee6a45f38528d2398add52d407bdd3ac8051b95ab86





**6. Build custom docker image and push it to docker hub and use it to create new container**

shubham.sahu@IN-IT18554 sample-node-app % **docker login**

Authenticating with existing credentials... [Username: shubham81994]

**i** Info → *To login with a different account, run 'docker logout' followed by 'docker login'*

Login Succeeded

shubham.sahu@IN-IT18554 sample-node-app % **docker build -t shubham81994/node-ssh-cron-app:latest .**

[+] Building 2.1s (12/12) FINISHED docker:desktop-linux

=> [internal] load build definition from Dockerfile 0.0s

=> => transferring dockerfile: 633B 0.0s

=> [internal] load metadata for docker.io/library/node:18 2.0s

=> [auth] library/node:pull token for registry-1.docker.io 0.0s

=> [internal] load .dockerignore 0.0s

=> => transferring context: 2B 0.0s

=> [1/6] FROM docker.io/library/node:18@sha256:df9fa4e0e39c9b97e30240b5bb1d99bdb861573a82002b2c52ac7d6b8d6d773e 0.0s

=> => resolve docker.io/library/node:18@sha256:df9fa4e0e39c9b97e30240b5bb1d99bdb861573a82002b2c52ac7d6b8d6d773e 0.0s

=> [internal] load build context 0.0s

=> => transferring context: 315B 0.0s

=> CACHED [2/6] RUN apt-get update && apt-get install -y openssh-server cron 0.0s

=> CACHED [3/6] RUN mkdir /var/run/sshd && echo 'root:root' | chpasswd && sed -i 's/#PermitRootLogin prohibit-password/PermitRootLogin yes/' /etc/ssh/sshd\_config 0.0s

=> CACHED [4/6] WORKDIR /usr/src/app 0.0s

=> CACHED [5/6] COPY . . 0.0s

=> CACHED [6/6] RUN npm install 0.0s

=> exporting to image 0.0s

=> => exporting layers 0.0s

=> => exporting manifest sha256:228a26cb44f000992e958beb3887e083b721ac2aeae2f9084c3bc069199e86f7 0.0s

=> => exporting config sha256:f050153ce9603eb979a3bd4296c5816d8d155cdda36d695d3d3ce266a545c94f 0.0s

=> => exporting attestation manifest sha256:9f463d3cbc8690a1889af276adaabea20a25daaf6c4e144446f8b943a57d4203 0.0s

=> => exporting manifest list sha256:46f771403638de51ee6d084dc2a7e13f1b9bae893aa90db73f38ca4463193258 0.0s

=> => naming to docker.io/shubham81994/node-ssh-cron-app:latest 0.0s

=> => unpacking to docker.io/shubham81994/node-ssh-cron-app:latest 0.0s

View build details: docker-desktop://dashboard/build/desktop-linux/desktop-linux/zp3qnfwgwuanwzkkehbeawrf1

shubham.sahu@IN-IT18554 sample-node-app % **docker push shubham81994/node-ssh-cron-app:latest**

The push refers to repository [docker.io/shubham81994/node-ssh-cron-app]

43b3ca1db9e3: Pushed

62cad2f6aff7: Pushed

ebf144460616: Pushed

e171895483c6: Pushed

eec17f84ad0a: Pushed

0e3cee1fc214: Pushed

a0d4055886c0: Pushed

99a6d4efaea7: Pushed

faa678027fe0: Pushed

002e18bd5659: Pushed

71daa2c787b0: Pushed

9d81c6467275: Pushed

68075717cb06: Pushed

a82933750783: Pushed

latest: digest: sha256:46f771403638de51ee6d084dc2a7e13f1b9bae893aa90db73f38ca4463193258 size: 856

shubham.sahu@IN-IT18554 sample-node-app % **docker images**

REPOSITORY TAG IMAGE ID CREATED SIZE

node-ssh-cron-app latest c5a418a594e7 21 hours ago 1.66GB

shubham81994/node-ssh-cron-app latest 46f771403638 21 hours ago 1.66GB

nginx latest 5ed8fcc66f4e 8 days ago 281MB

alpine latest a8560b36e8b8 2 months ago 12.8MB

shubham.sahu@IN-IT18554 sample-node-app %

