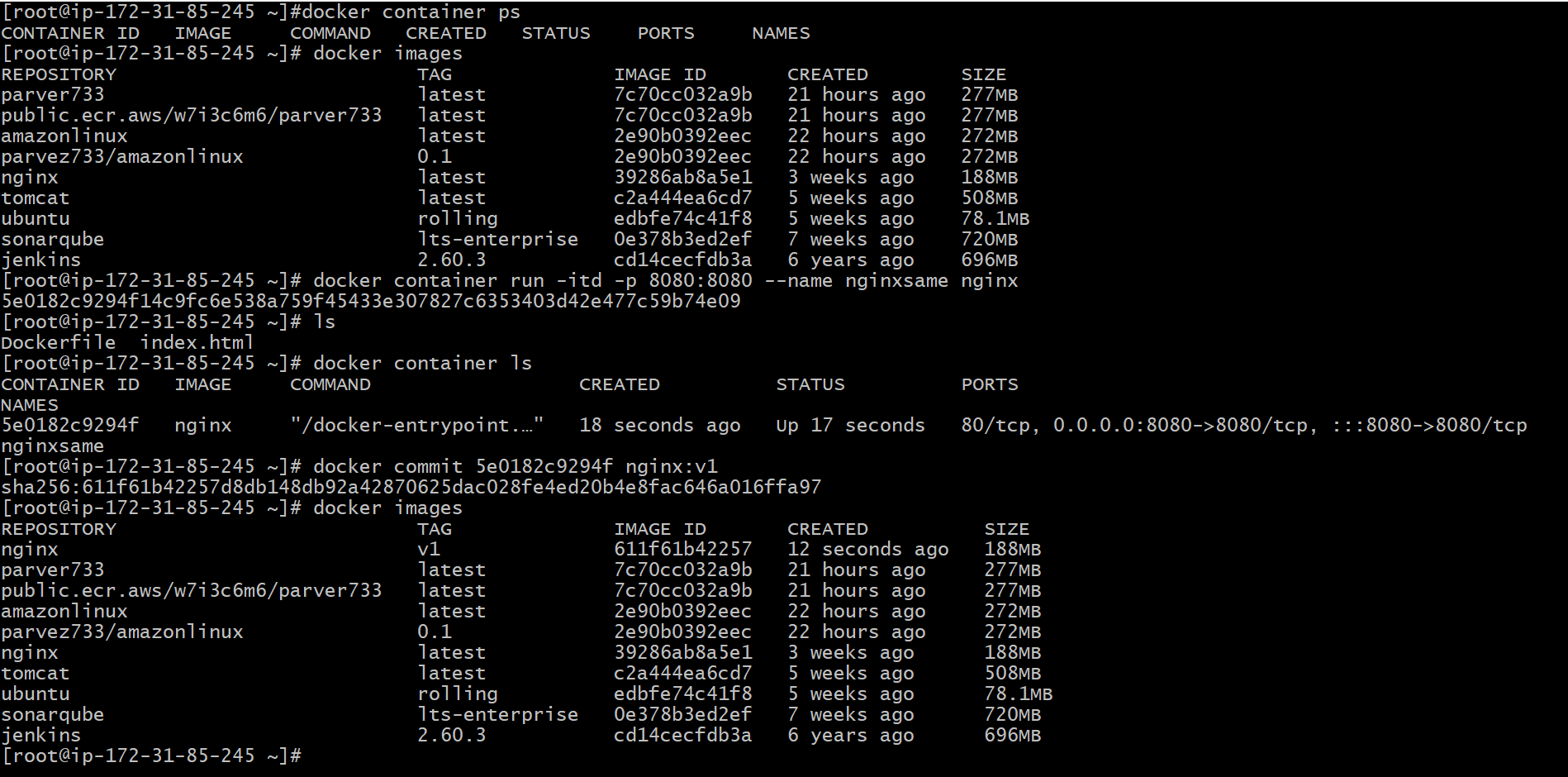
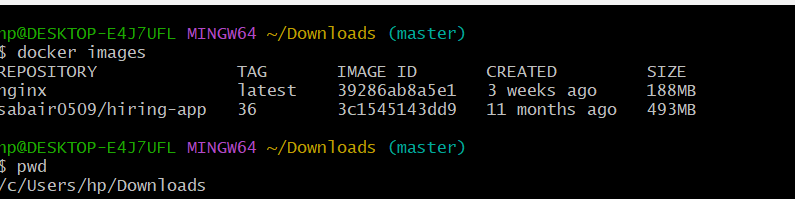
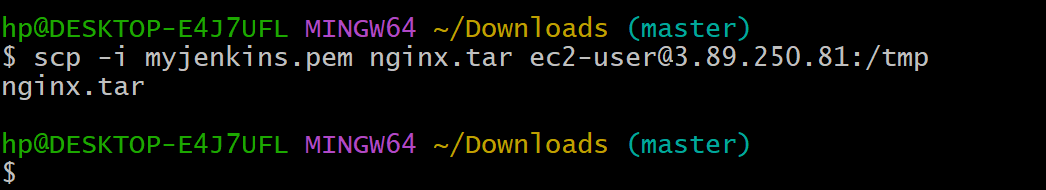
**DOCKER TASK 3**

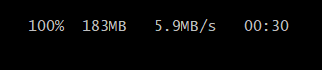
1.Create a image from running container.



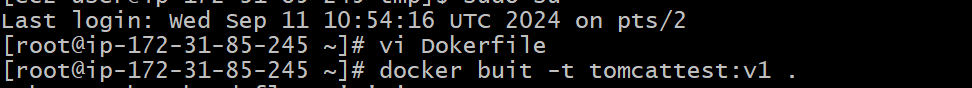
2.Copy image from local machine to docker server and load the image.

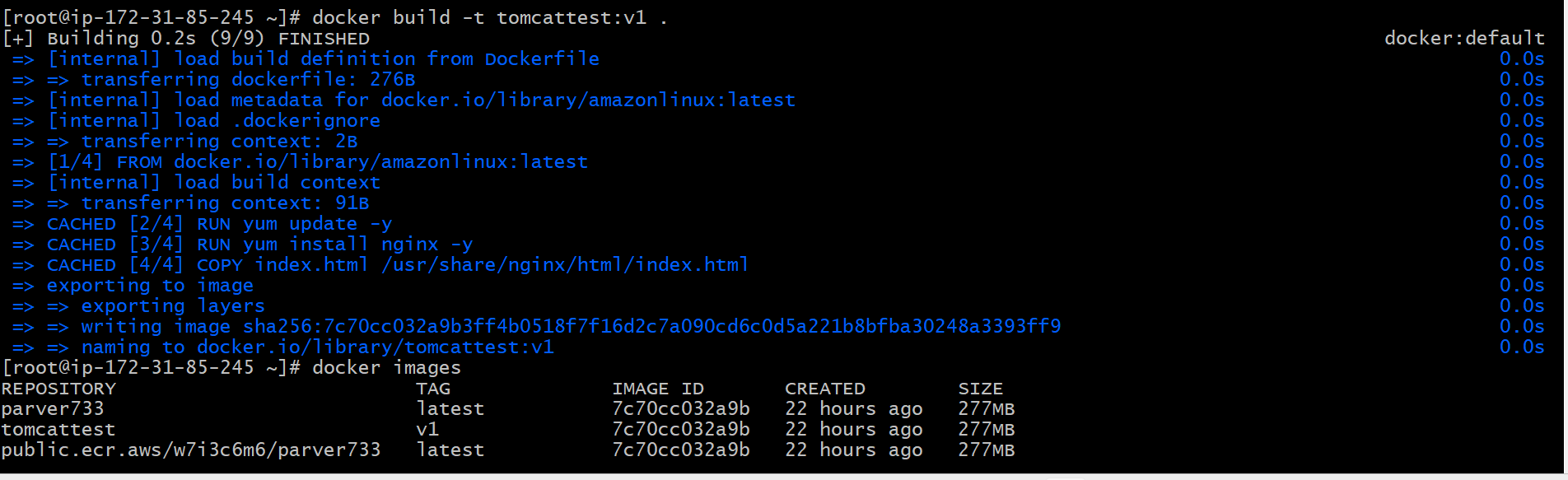


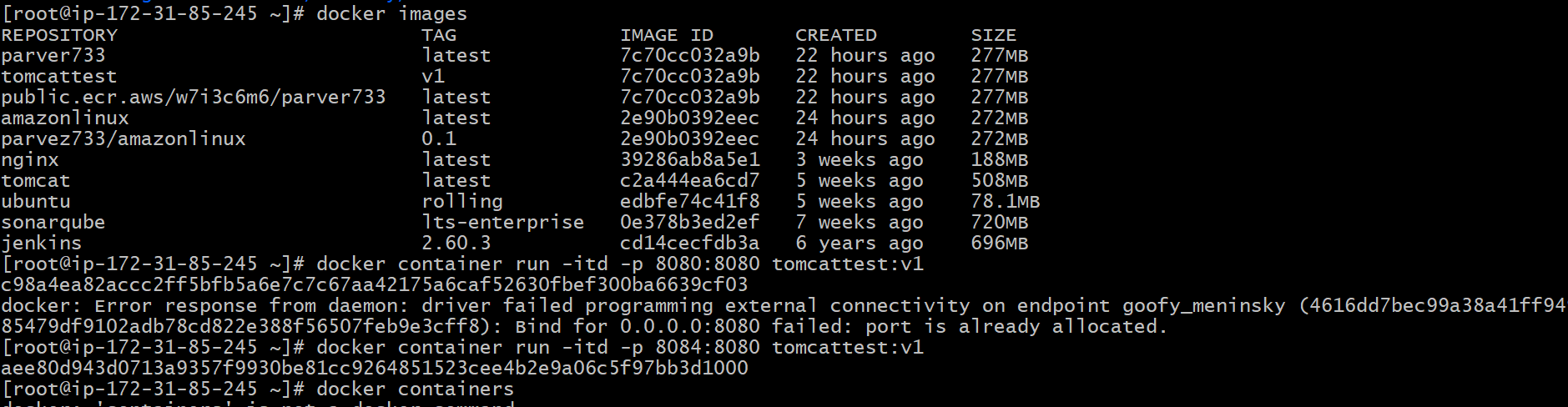




3.Create Docker image using alpine and customize with tomcat.

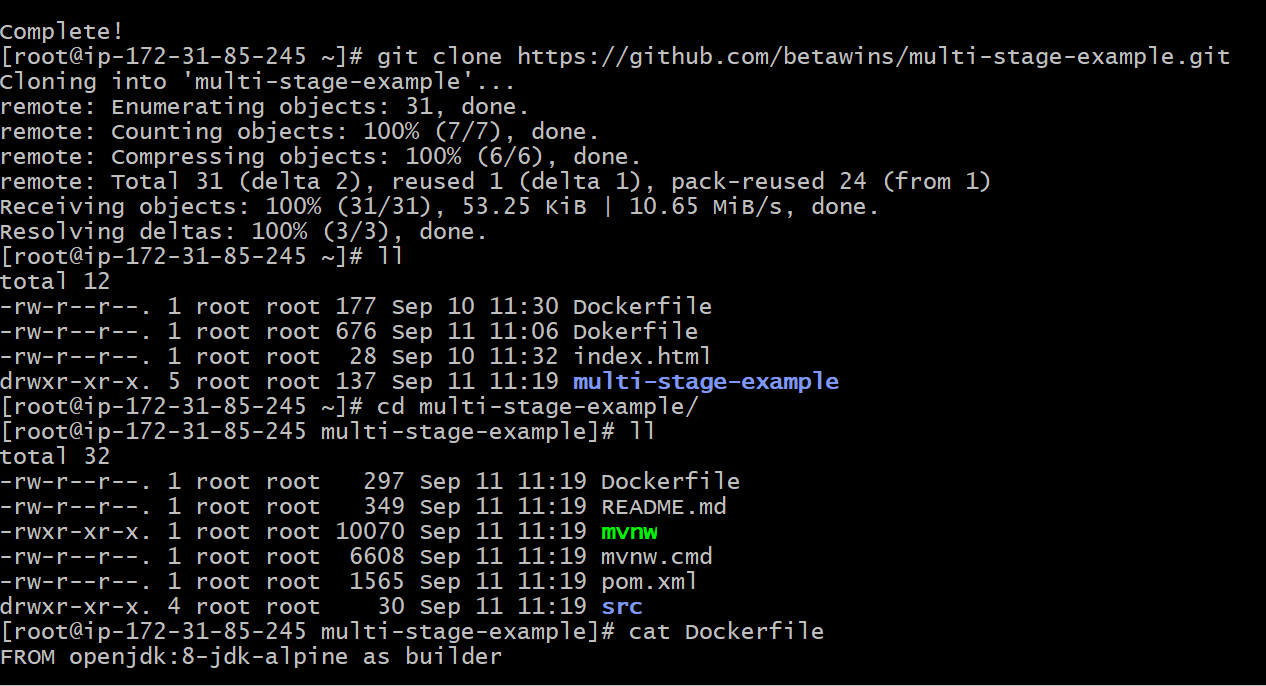


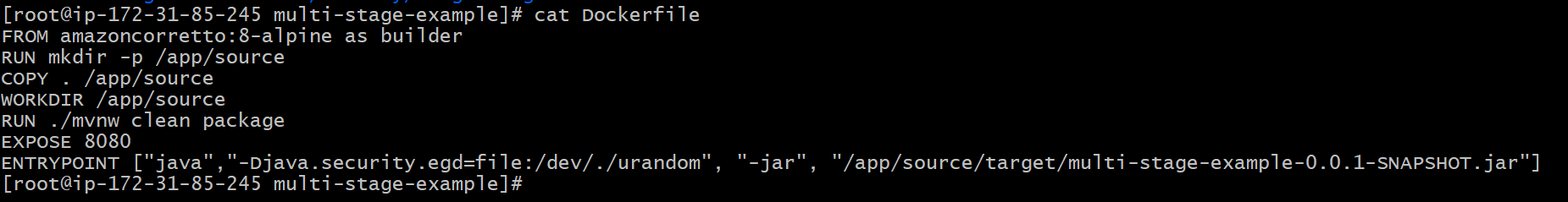


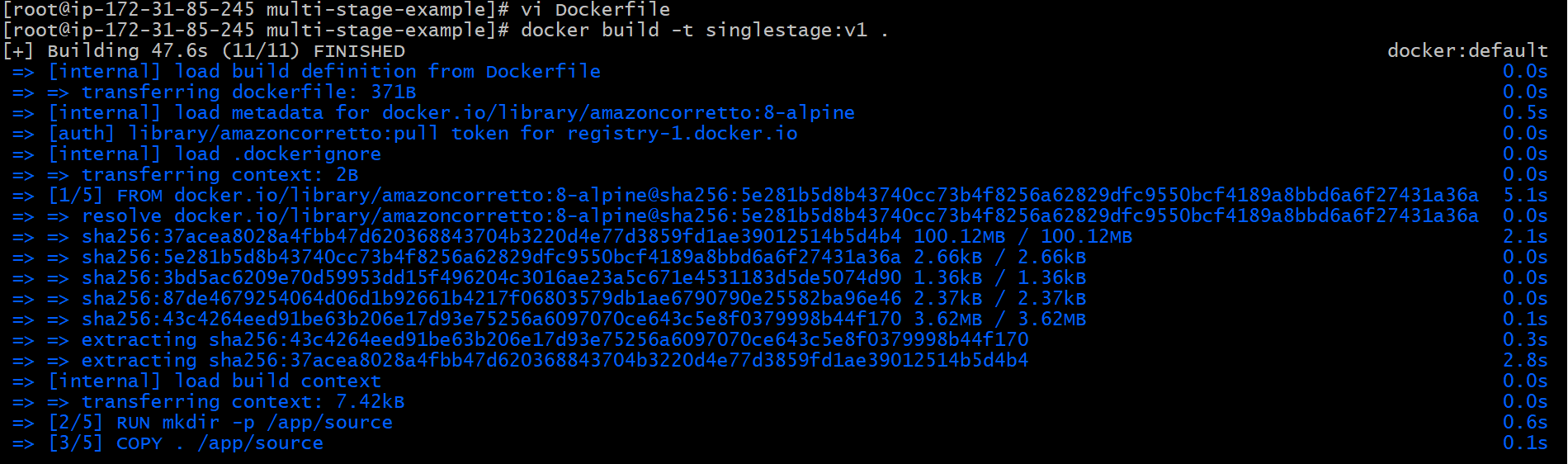


Create single stage and multi stage docker file using the below source code. <https://github.com/betawins/multi-stage-example.git>

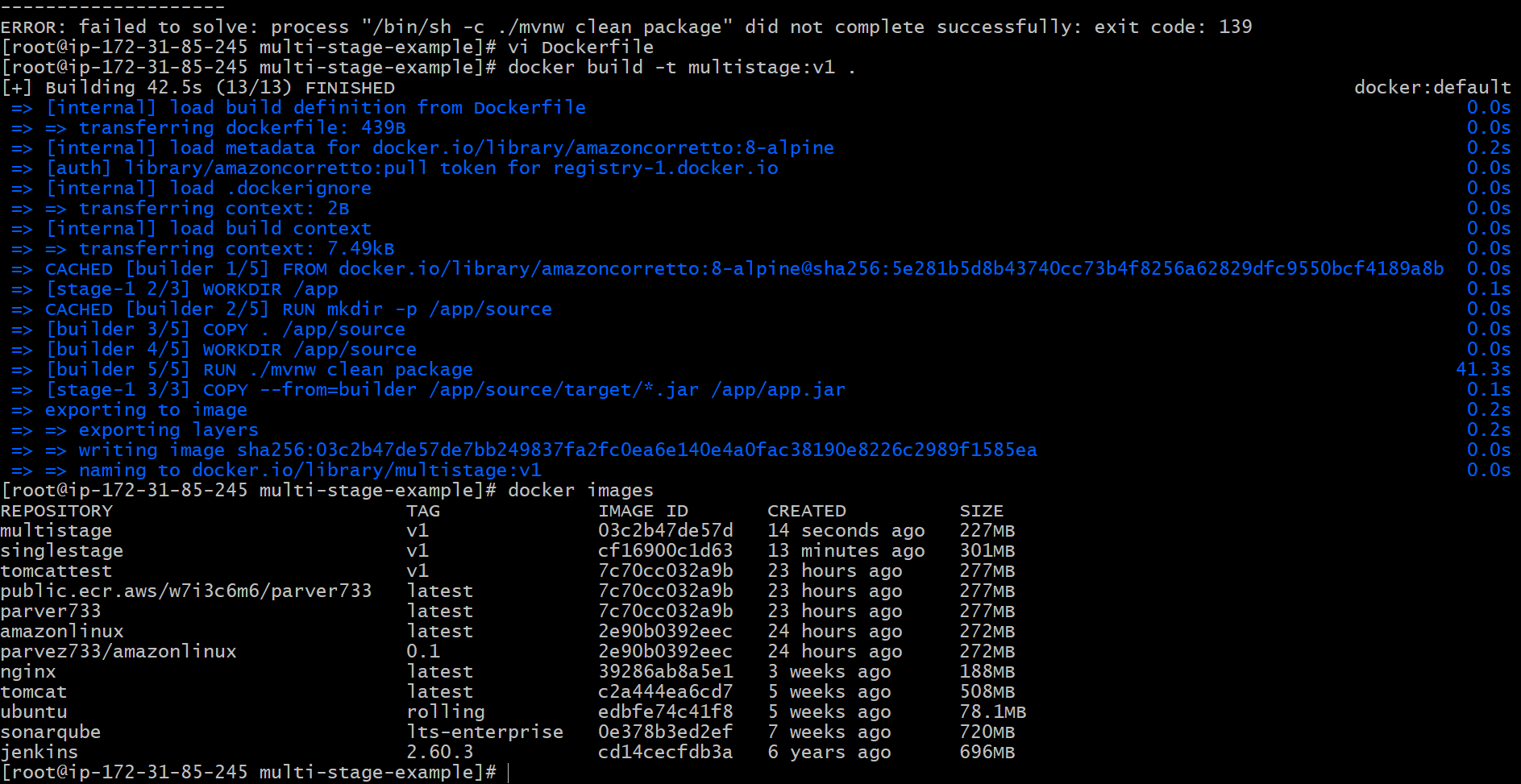
Single Stage Dockerfile



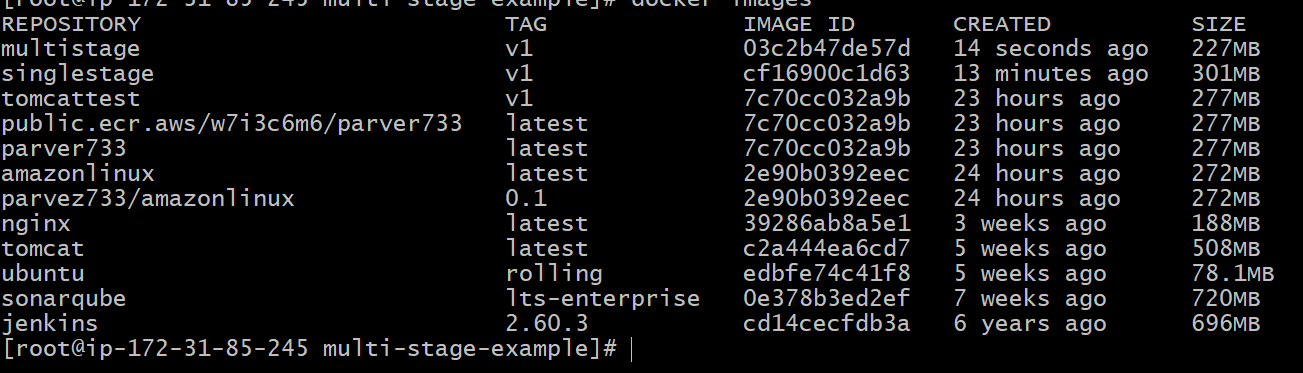


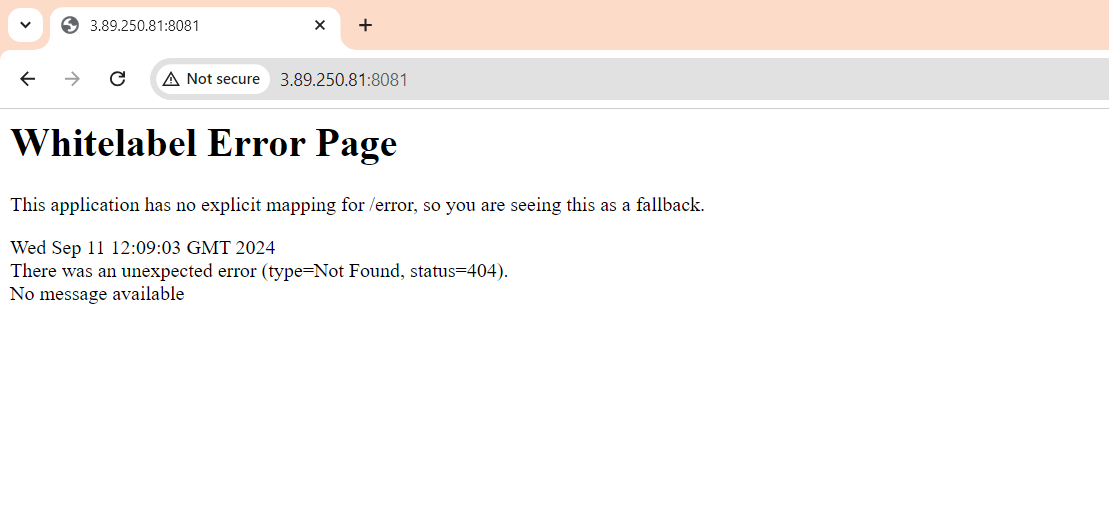


Multi-stage Docker file



Here, we can see in singlestage the size and compare to the multistage the docker file is decreases that is the use multistageDockerfile



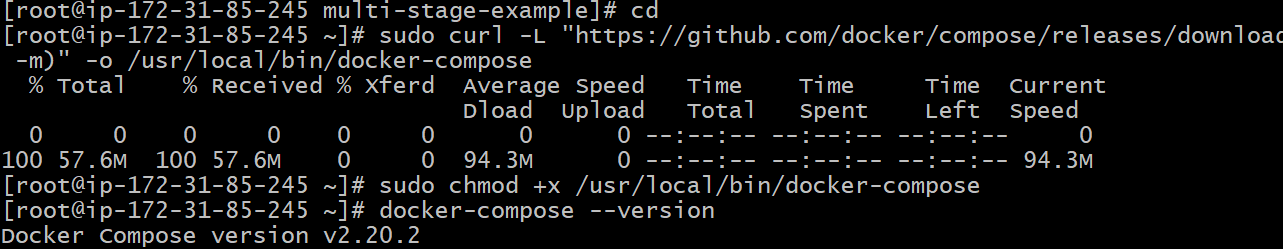


5.Install docker compose and execute sample application.

a) sudo curl -L "https://github.com/docker/compose/releases/download/v2.20.2/docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose

b) give permissions sudo chmod +x /usr/local/bin/docker-compose

c)check the version docker-compose --version



vi docker-compose.yml

sample-two-tier:docker-compose file

========================

version: '3'

services:

db:

image: mysql:5.7

volumes:

- db\_data:/var/lib/mysql

restart: always

environment:

- MYSQL\_ROOT\_PASSWORD=somewordpress

- MYSQL\_DATABASE=wordpress

- MYSQL\_USER=wordpress

- MYSQL\_PASSWORD=wordpress

wordpress:

depends\_on:

- db

image: wordpress:latest

ports:

- "8000:80"

restart: always

environment:

- WORDPRESS\_DB\_HOST=db:3306

- WORDPRESS\_DB\_USER=wordpress

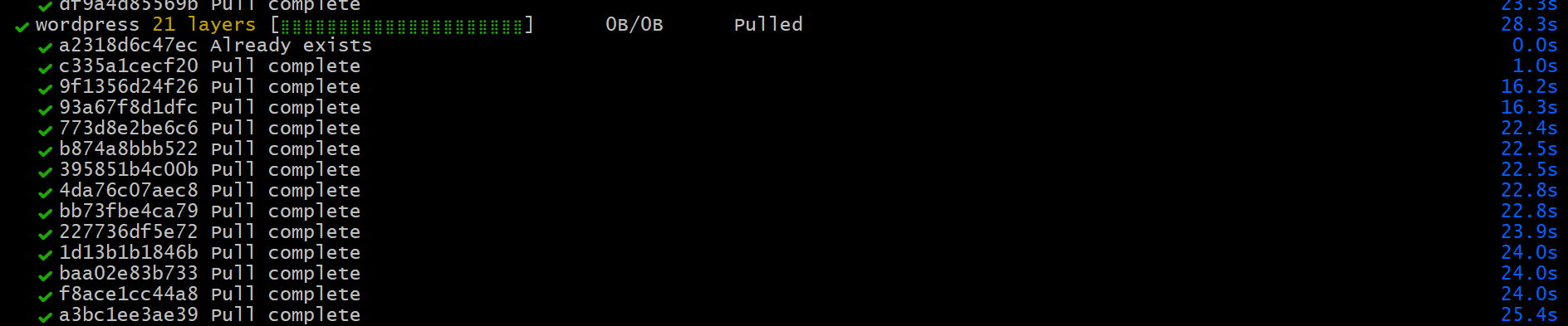
- WORDPRESS\_DB\_PASSWORD=wordpress

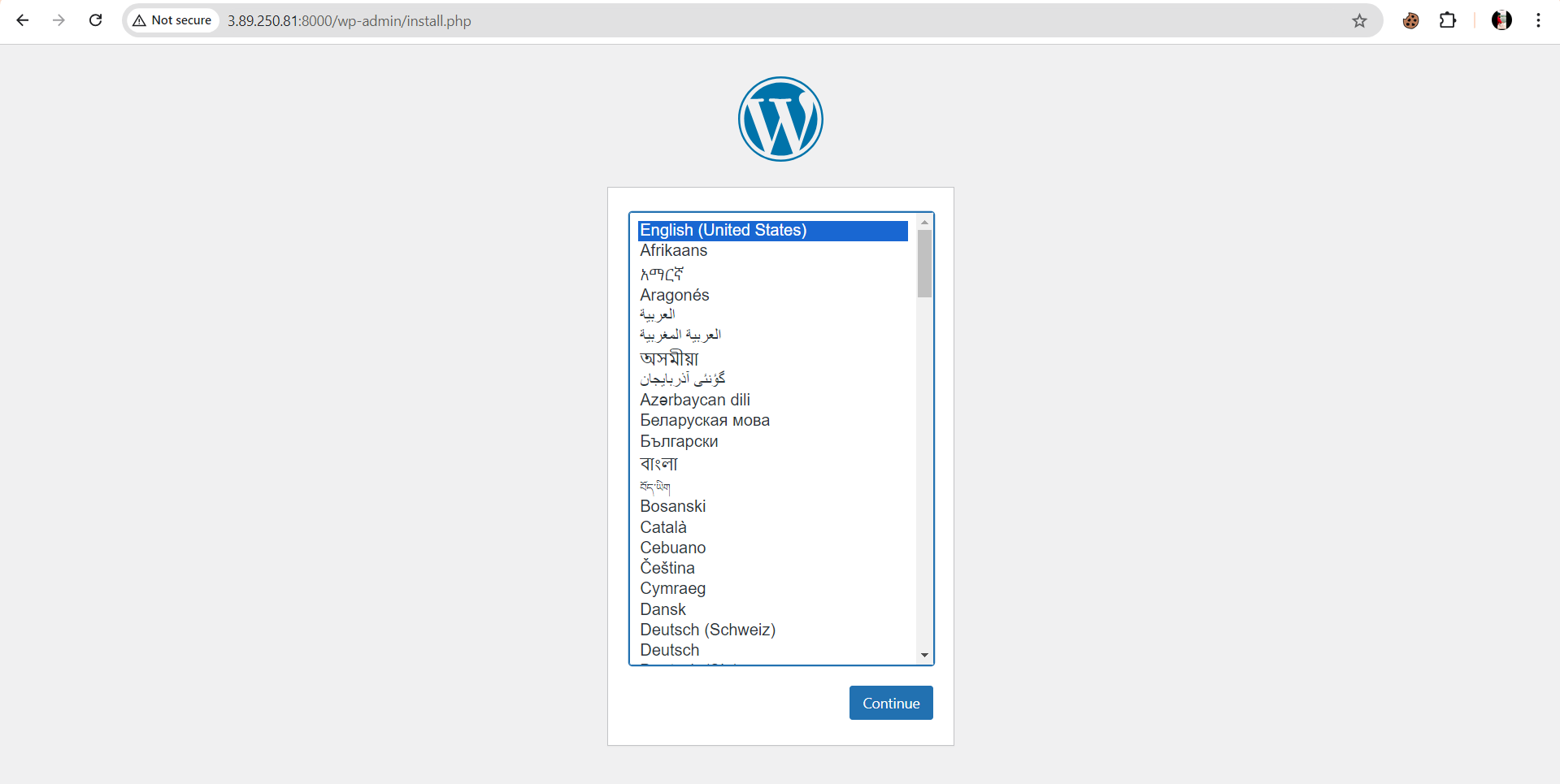
- WORDPRESS\_DB\_NAME=wordpress

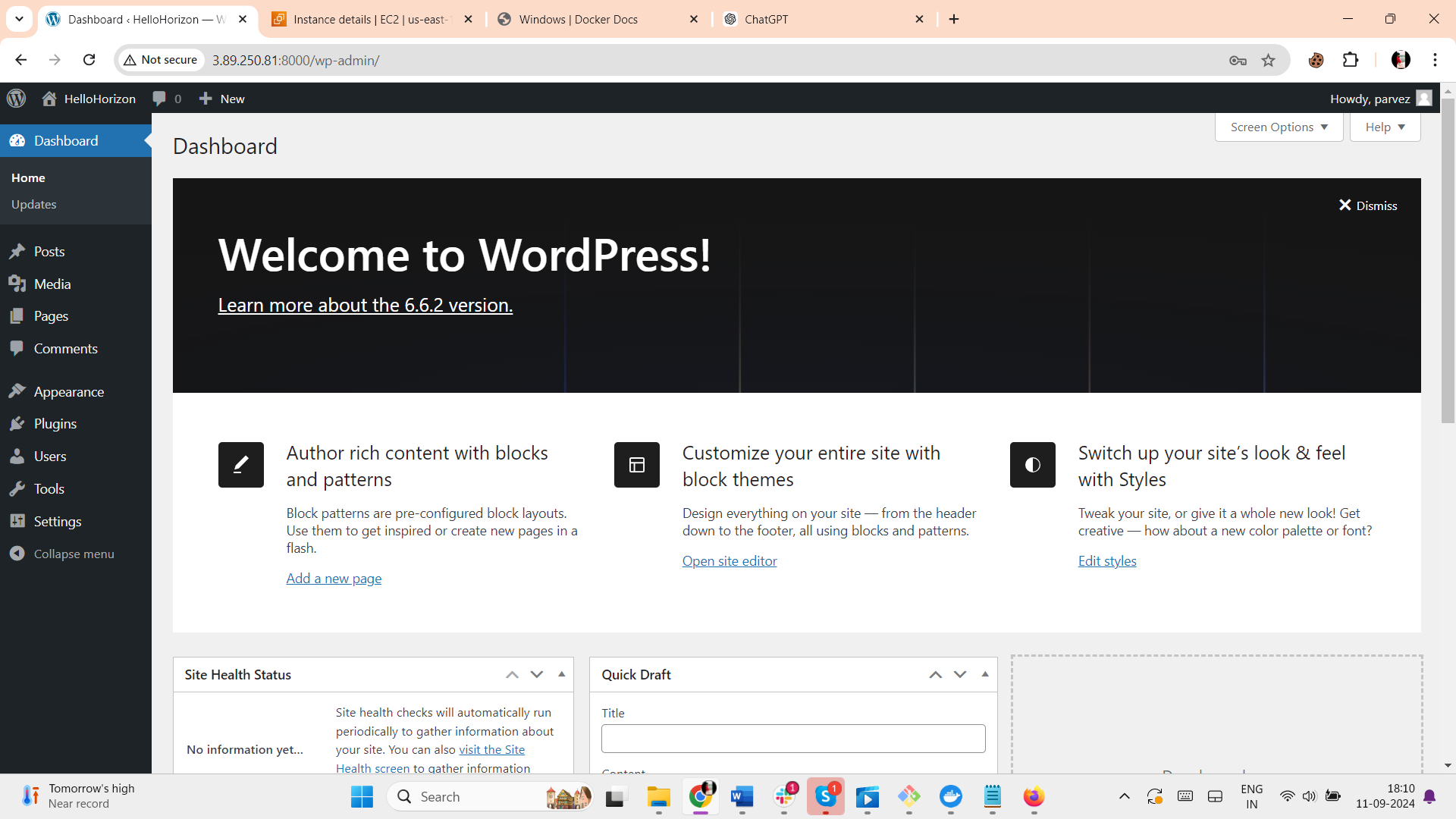
volumes:

db\_data: { }

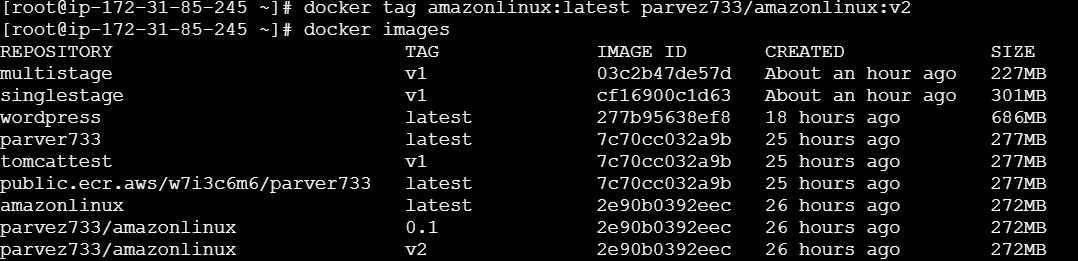








Implement solution to scan images when pushed to docker registry.



After the above step we need to change the scan option in the repository setting scout scan.

After that push the image

