**Experiment-7**

**AIM:- Improvise the docker image quality using devops.**

(Note:-Before do first follow below steps. Here we will create image with tag and also we minimize the image size changing version or using multi staging)

**Step 1:-Connecting AWS Instance Ubuntu using Mobaxterm**

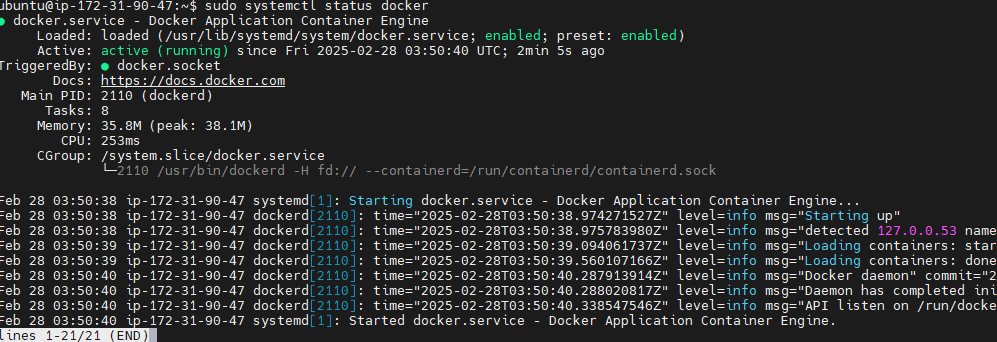
1. Login AWS(Amazon Web Services) Account
2. Lunch Instance name Docker
3. Connect to Ubuntu or mobaxterm

(**Note:-**Follow this url for docker file and application—https://github.com/devisar/devopslab)

**Step 2:-Create Docker Hub Account and create repository in Docker Hub**

**Step 3:-Install Docker and Check Status and Start Docker in mobaxterm**

1. sudo apt update -y
2. sudo apt install docker.io –y
3. sudo systemctl status docker(come outside use command ctl+z)



Above status command is docker running means no problem if not run use command below to run

1. sudo systemctl start docker

### Step 4:- Grant Access

### Why we give grant access means

A easy way to verify your Docker installation is by running the below command

docker run hello-world

If the output says:

### 

This can mean two things,

1. Docker deamon is not running.(start docker using “sudo systemctl start docker” )
2. Your user does not have access to run docker commands.

### Grant Access to your user to run docker commands

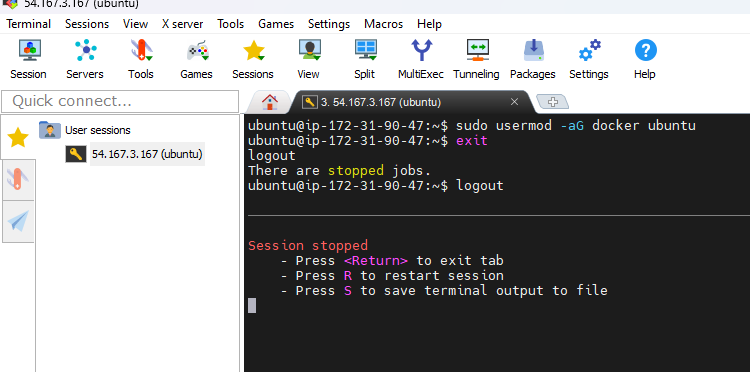
1. sudo usermod -aG docker ubuntu

In the above command ubuntu is the name of the user, you can change the username appropriately.

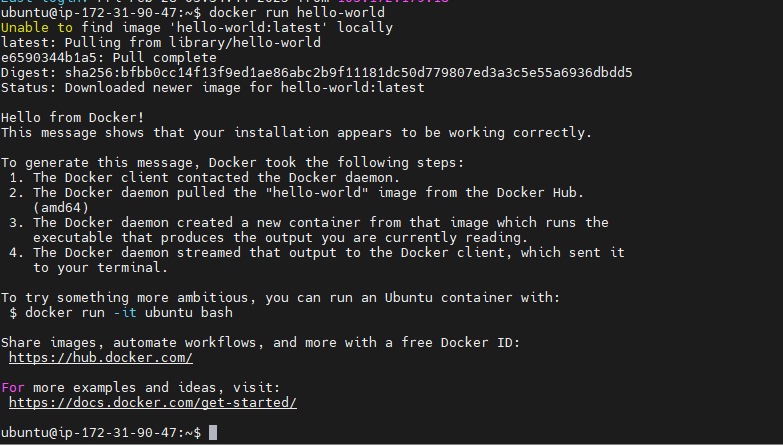
**NOTE:** : You need to logout and login back for the changes to be reflected.

1. Logout purpose use commands exit or logout and Press R to restart session.

Again run command “docker run hello-world”



Start session



**Step 5:-Creating application and Docker file**

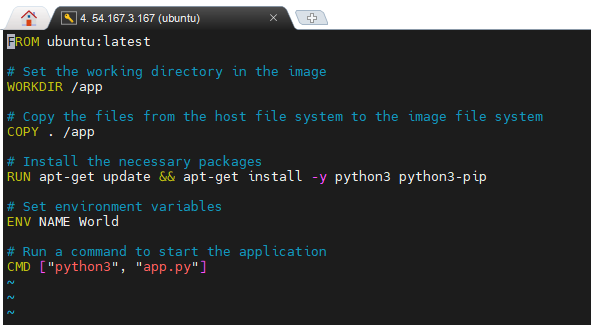
**Commands:-**

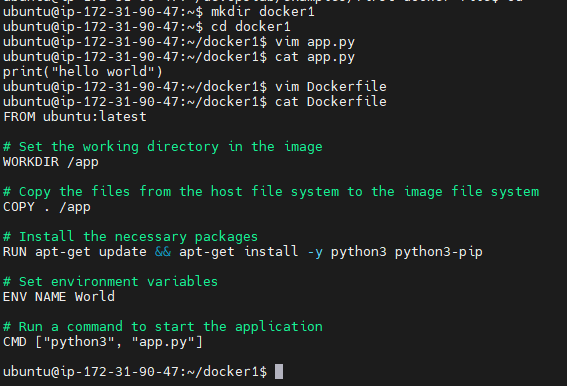
1. Mkdir docker1
2. cd docker1
3. vim app.py

print(“hello world”)

1. cat app.py
2. vim Dockerfile

(below pic commands write lab record) typing command vim before click “i” for insert data after completion Docker file commands save before click esc use :wq!



**Below image for understanding purpose**se

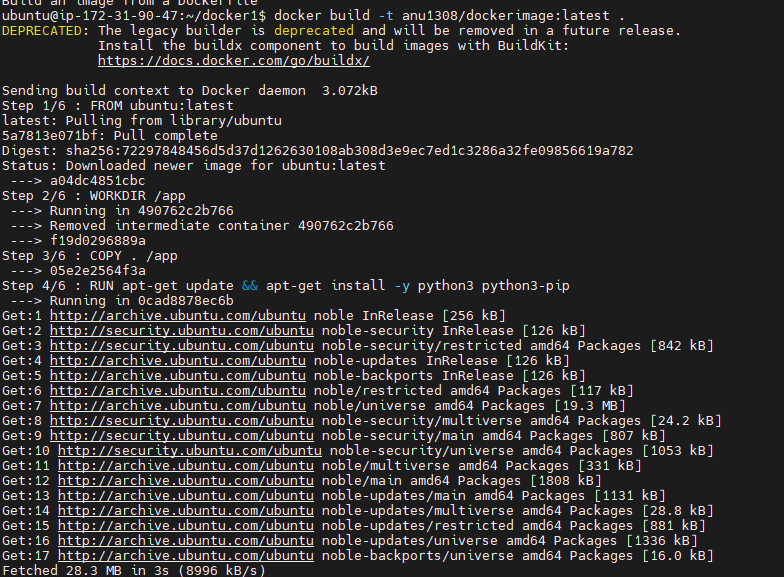
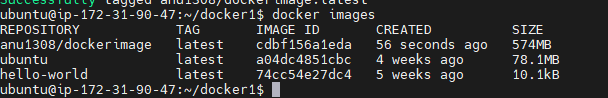
**Step 6:-** **Build and Check Docker image**

Syntax:- docker build –t dockerhub\_username/repositoryname:tag .

# docker build –t anu1308/dockerimage:latest .

1. docker images

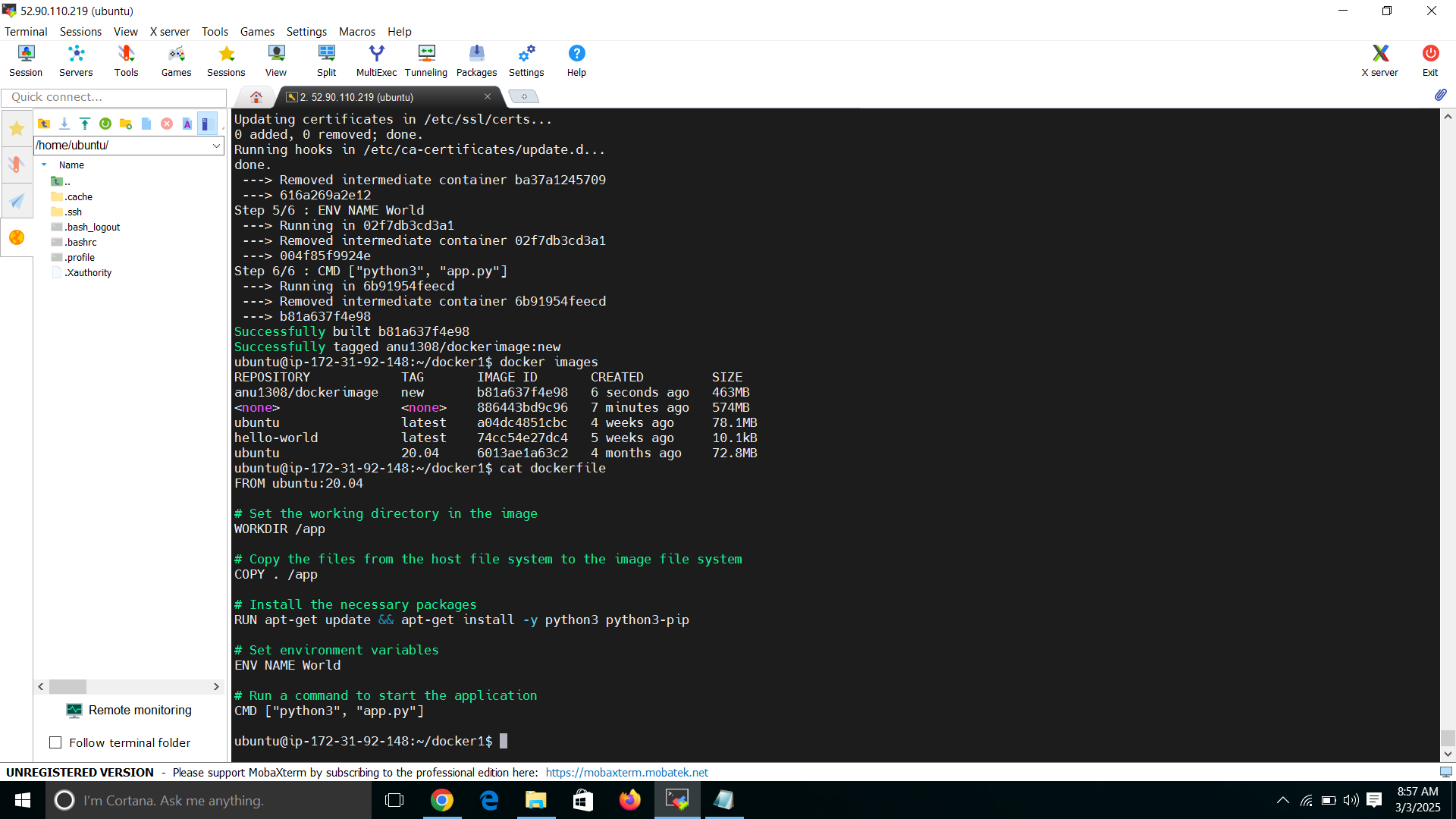
**Below images for understanding purpose**

**Step 7:-Changing Version to minimize image quality**

Instead of using ubuntu:latest, consider lighter base images like python:3.9-slim or python:3.9-alpine or specific versions of ubuntu like ubuntu:20.04 for smaller,stable and secure images.

(Note:-Write from to cmd all in you are lab record)

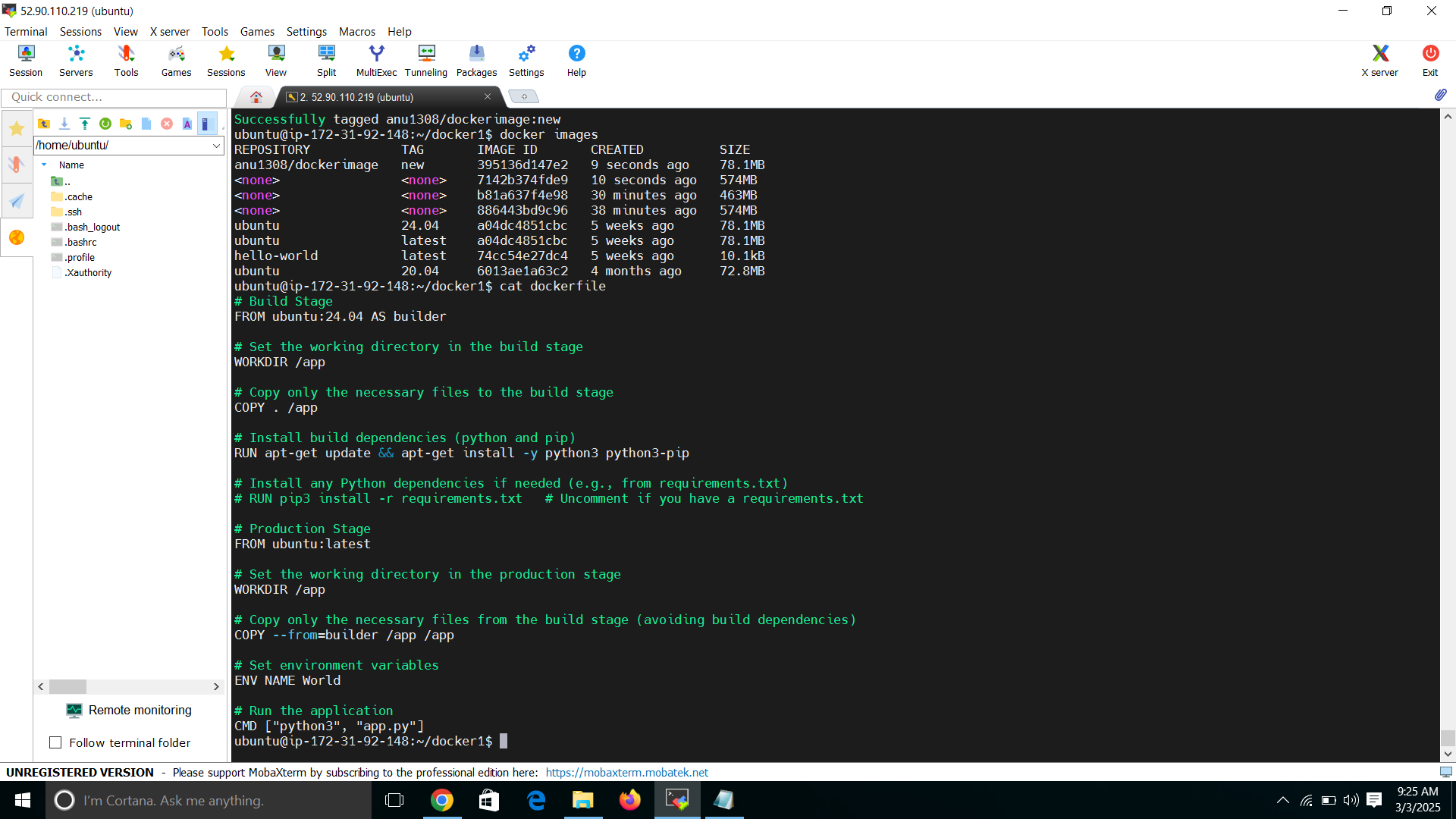


**Step 8:-Using Multi stages to minimizing image quality**

Multi-stage builds allow you to separate the build environment from the production environment.This reduces the final image size.

* **Steps:**
  + Create a **build stage** where you compile or install dependencies.
  + Copy only the necessary files to the **final stage**, leaving behind build tools and unnecessary files.

(Note:-Write from to cmd all in you are lab record)



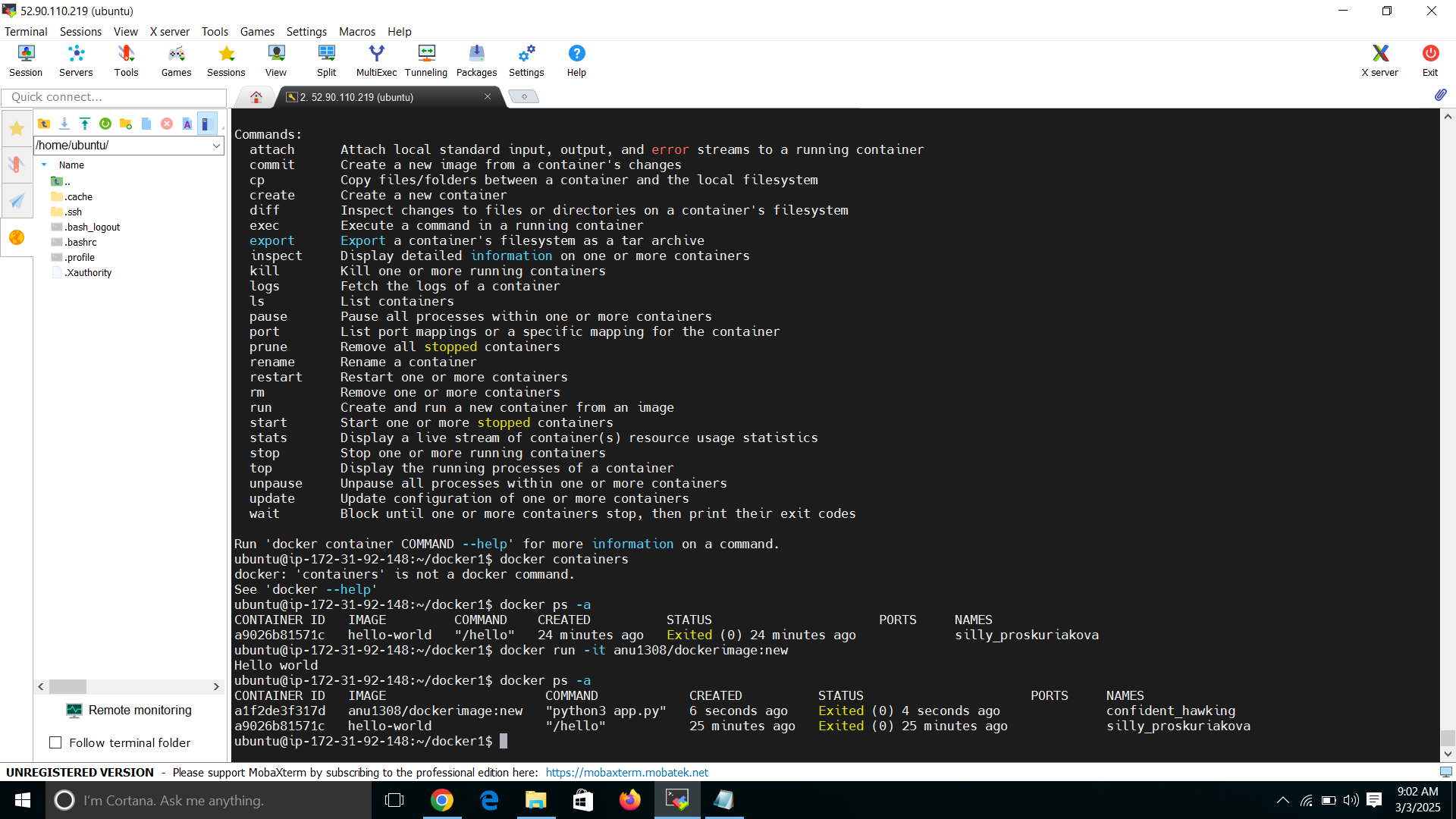
**Step 9:-Creating Container and check container**

Create command:-

docker run –it anu1308/dockerimage:new

Check Container command:-

docker ps -a



**Output:-**

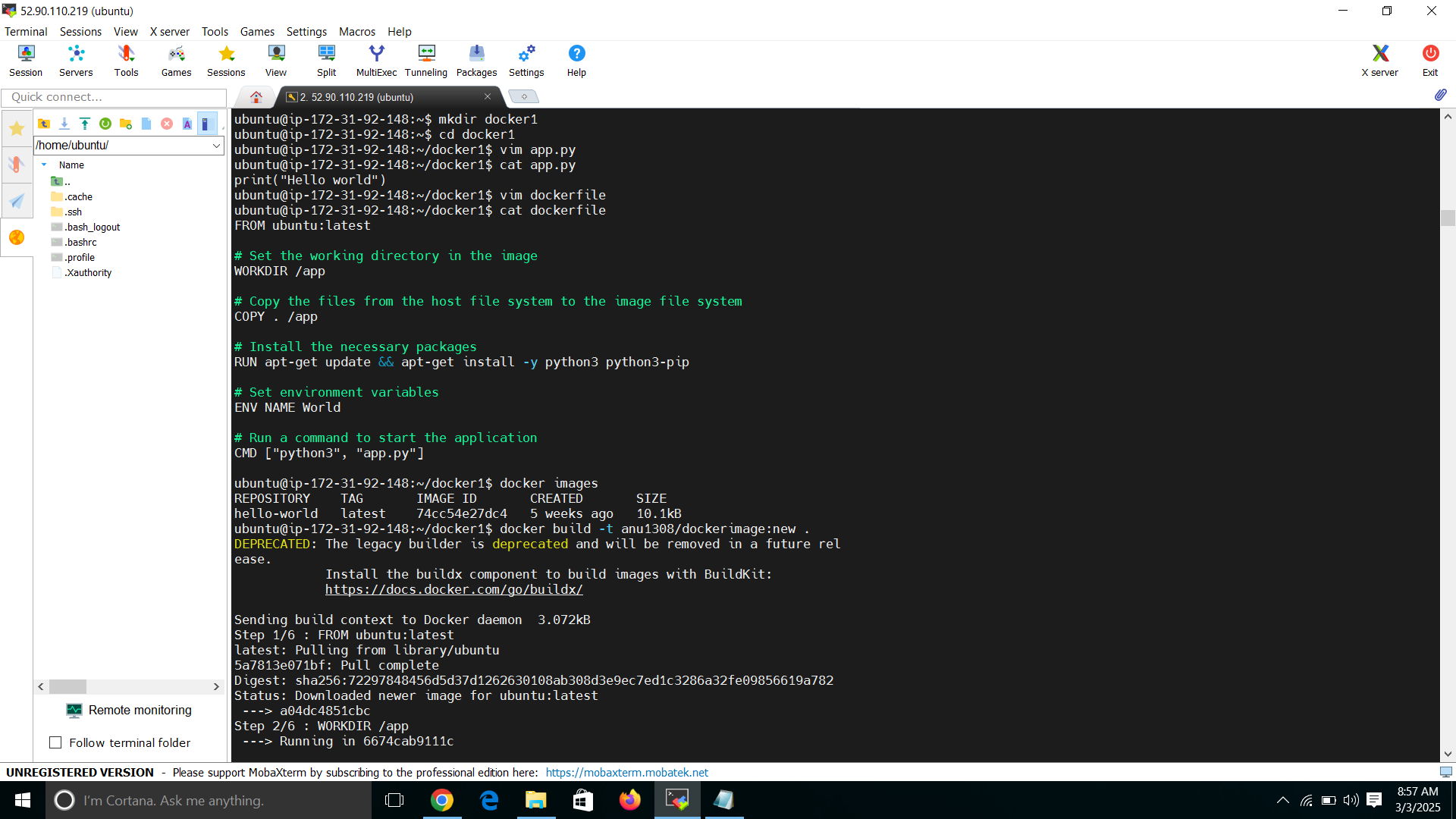
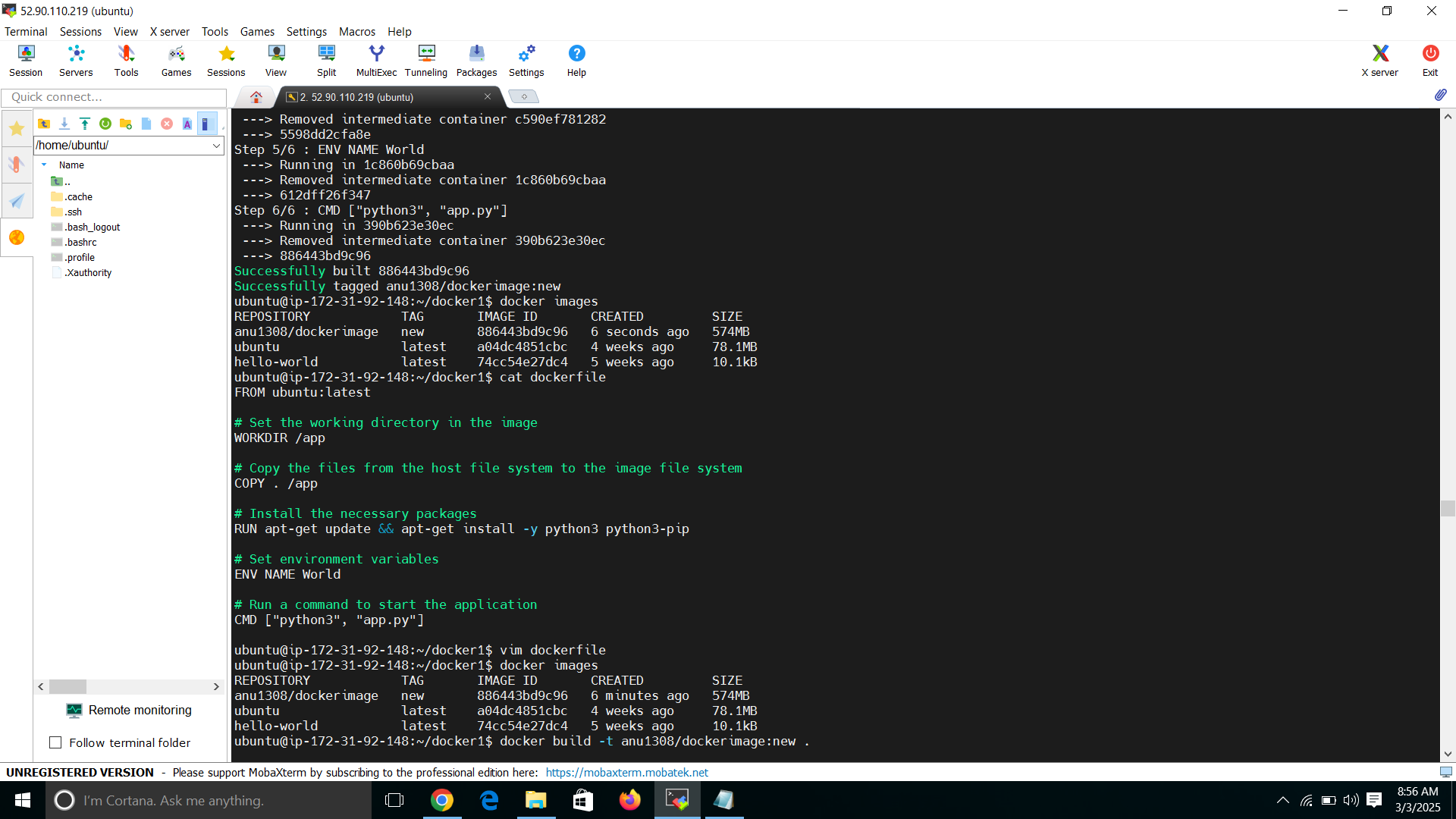
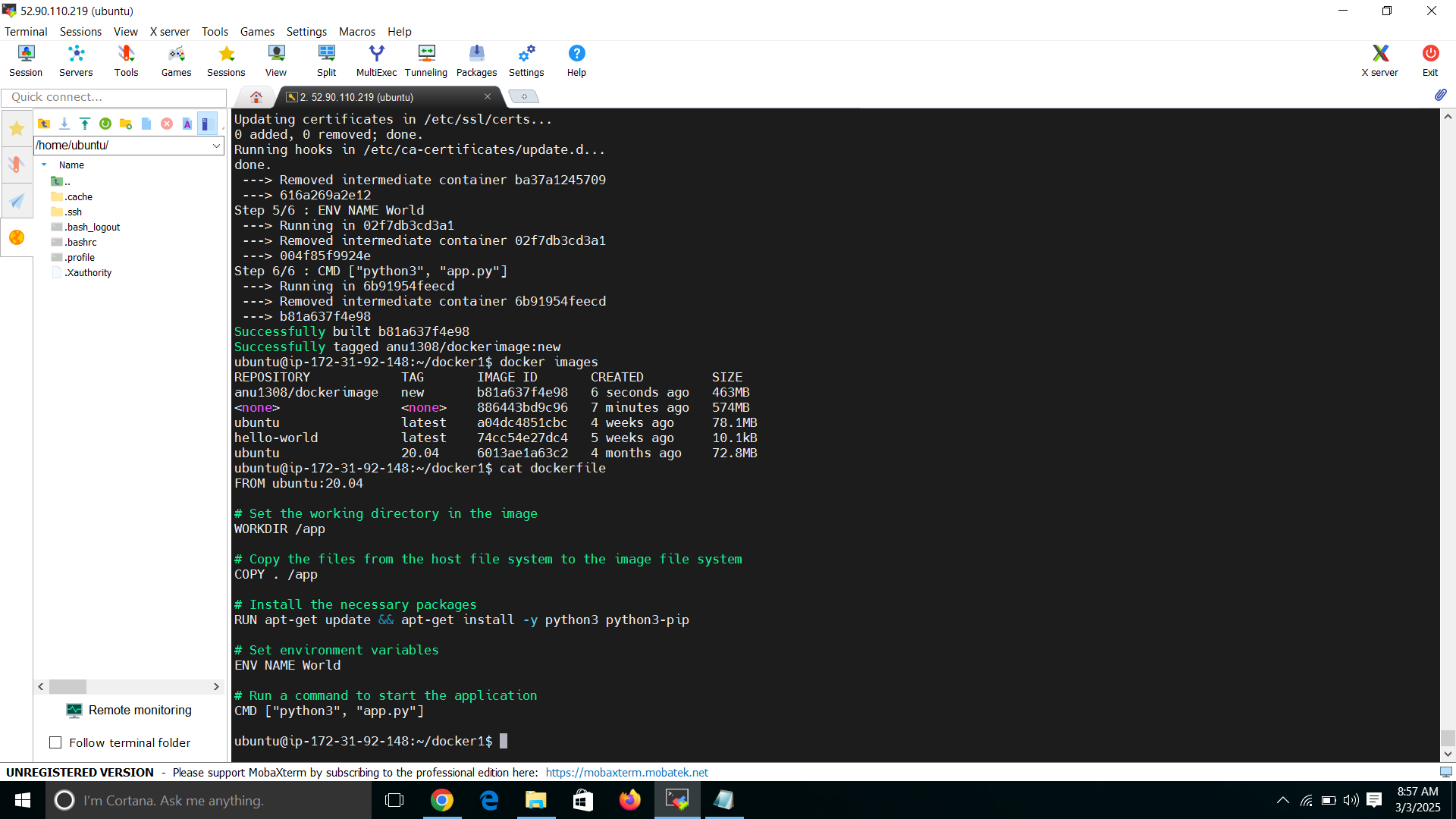
****

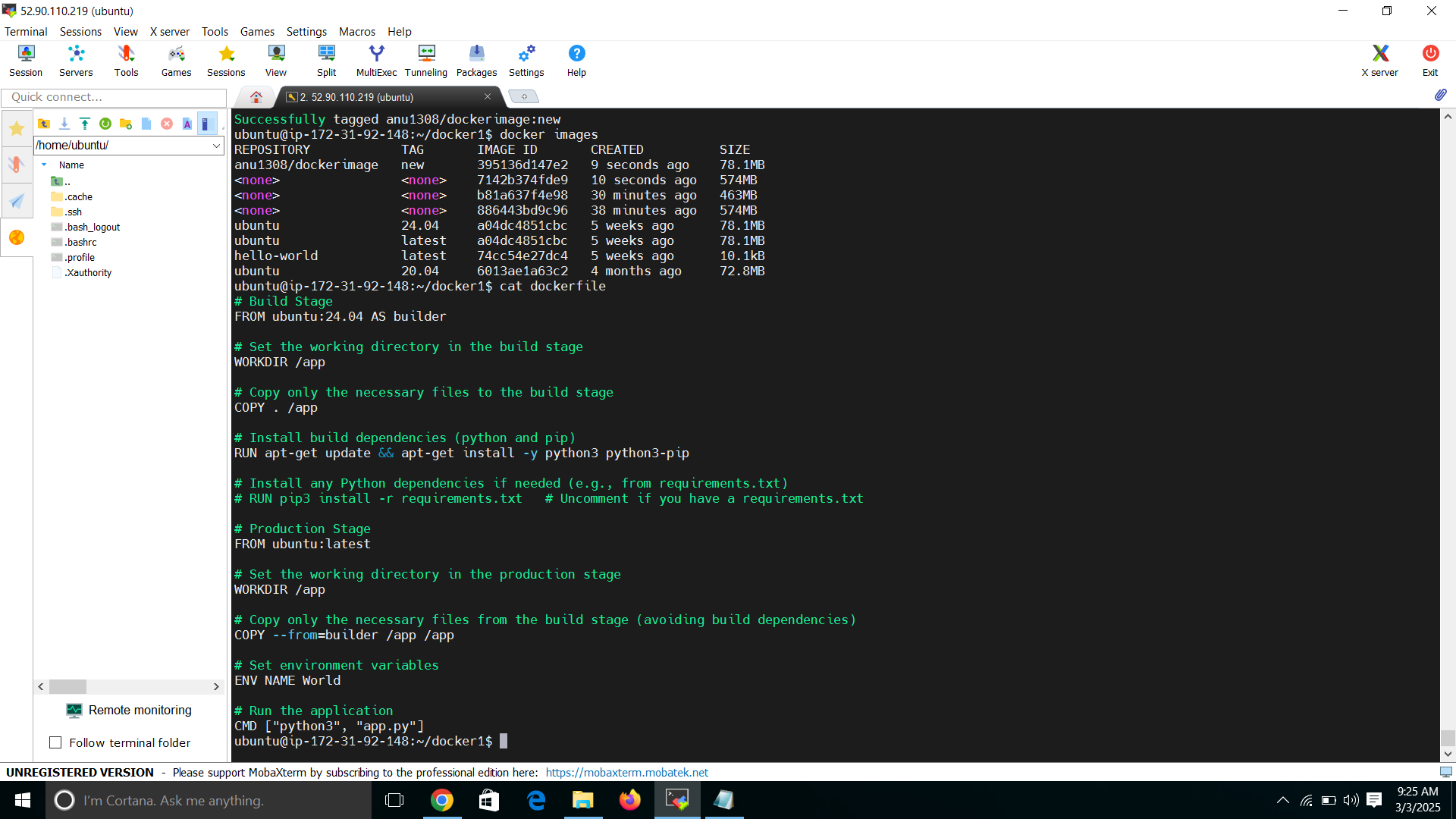
Image tag

****

**Fig.1.Before Build no image is created and After Build Image created**

****

**Fig.2.After changing version latest to 20.04 Image size minimize 574 to 463**



**Fig.3.using Multi Stage to minimize image size 463 to 78.1 MB**