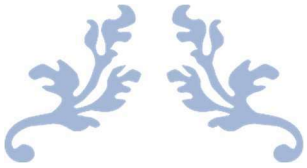


SETUP

JANUARY 2024

Abdulfatah Abdillahi

Contents

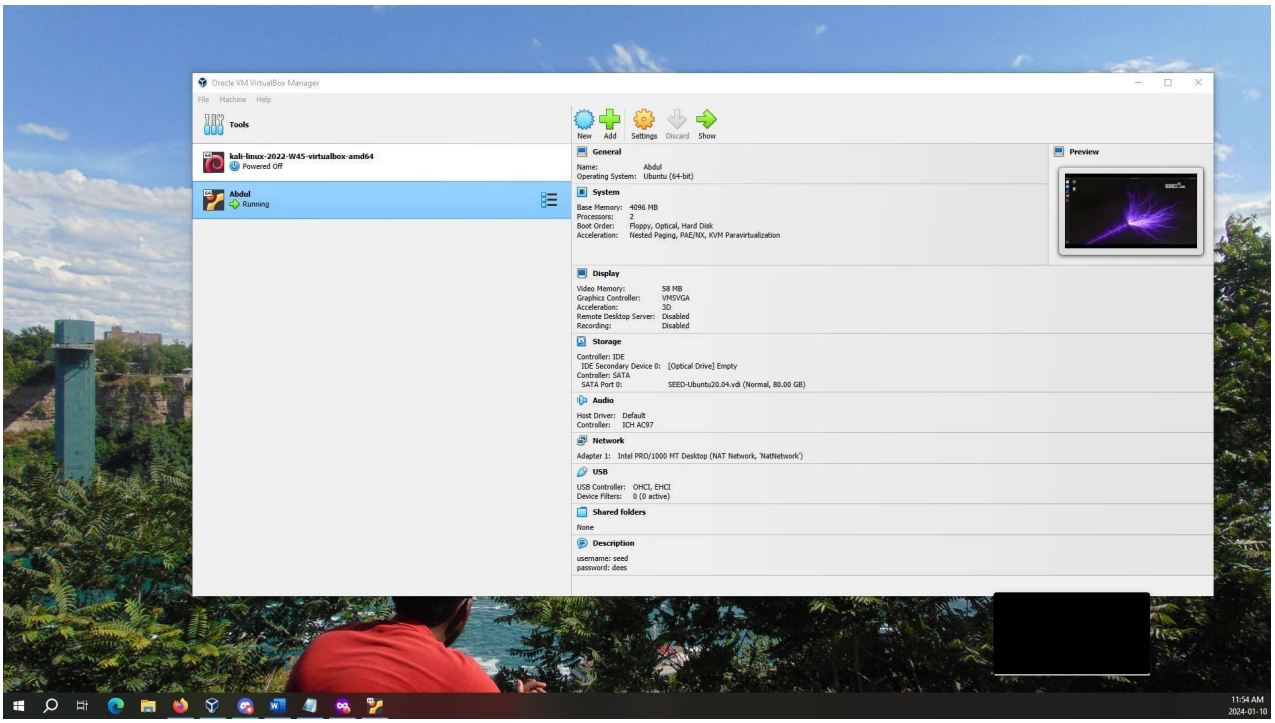
Introduction

In this lab, have set up and understood the environment required to use the SEED Labs, and create some containers.

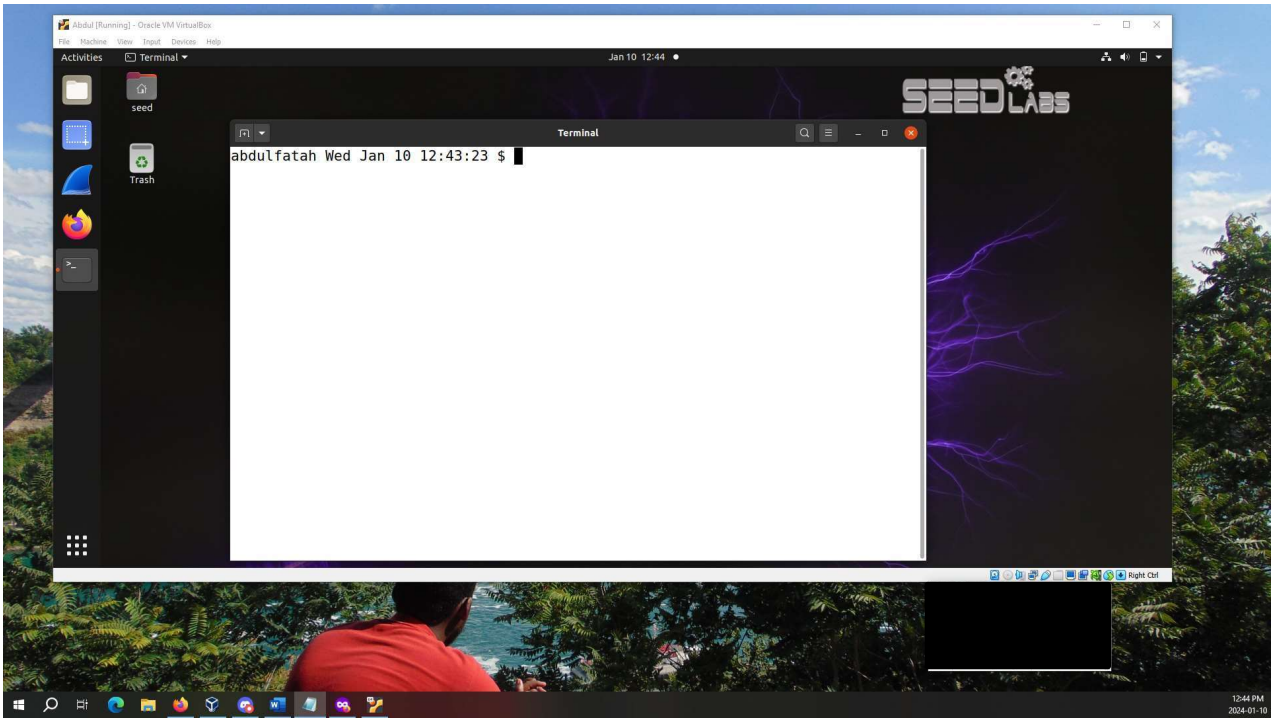
Tasks

Create SEED VM

Here we start off the lab by setting up our environment and creating a SEED VM using the pre-built SEED Ubuntu 20.04 image. This will be used for all future labs. The below screenshot shows the settings that have been configured for the VM as outlined in the seedvm-manual.



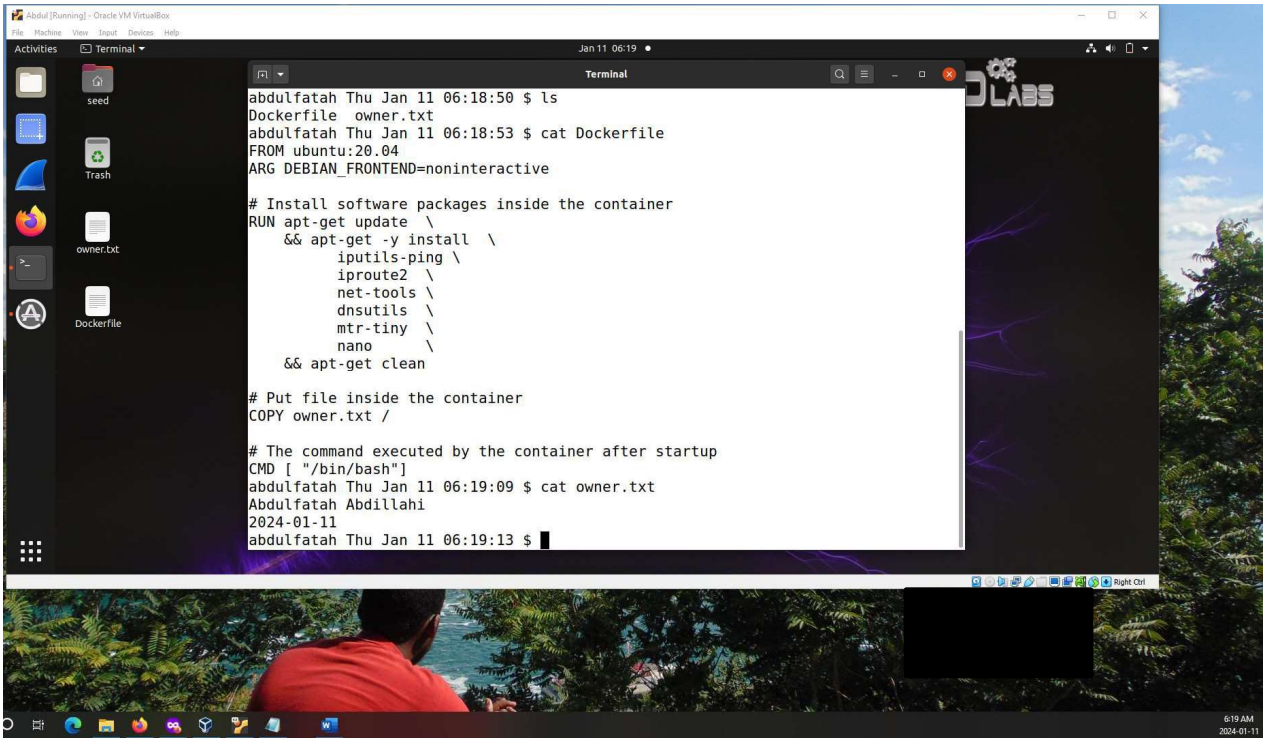
Upon running the VM for the first time, I made sure to change the command prompt as instructed to include my name (Abdulfatah), date, and time.



Docker Review for SEED Labs

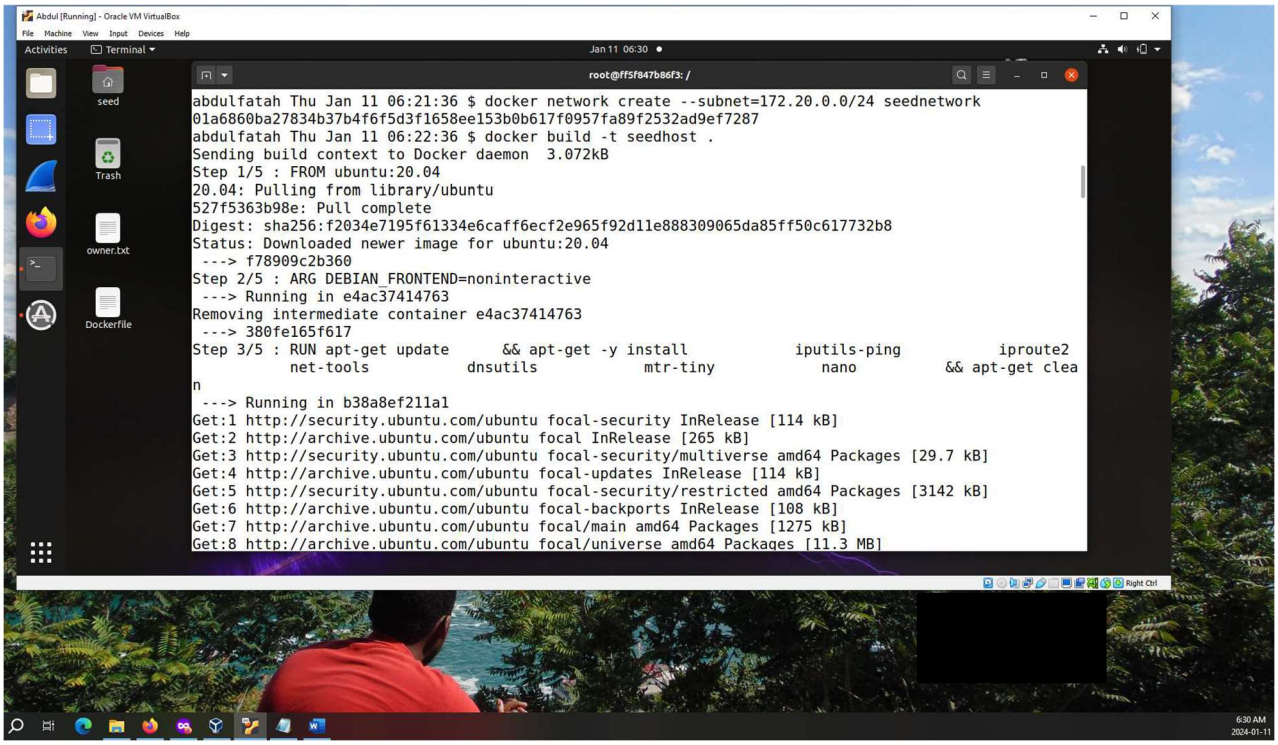
Writing the Dockerfile

For this part of the lab, we have built a custom Dockerfile from which a docker image can be built. As you can see in the screenshot, in the Dockerfile we specified that the container will be built based on the official Ubuntu 20.04 Docker image. We also installed numerous software packages and ran the “apt-get clean” command to empty the local repository and thereby decrease the overall size of the image. Moreover, as per the instruction in Blackboard I have the file copy line to copy a file named owner.txt, which included my name and the date.

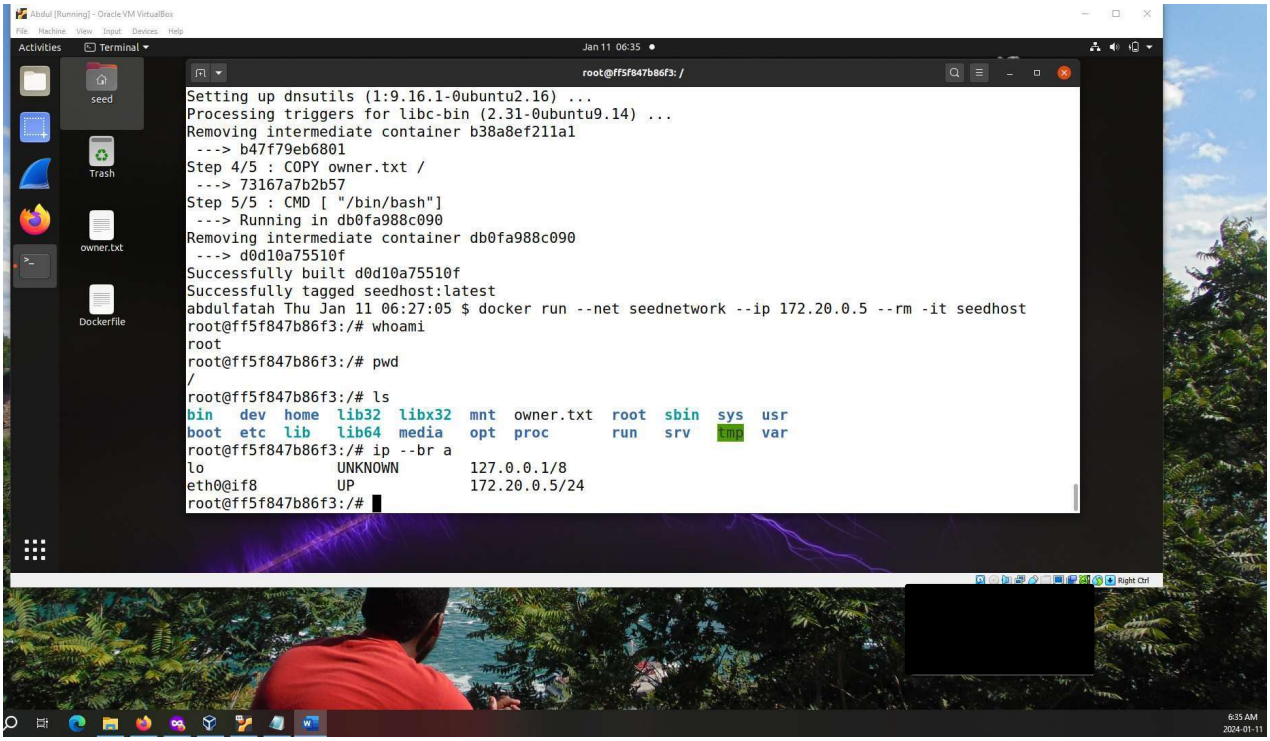


Building and starting the container

Here, we are assigning a static IP address to our container but before doing that we must first attach our container to a network. Here we use the command “docker network create --subnet=172.20.0.0/24 seednetwork” to create a network with the prefix *172.20.0.0/24* called *seednetwork*. Then I run the command “docker build -t seedhost .” to create a container called *seedhost* using the *Dockerfile* we just created beforehand.

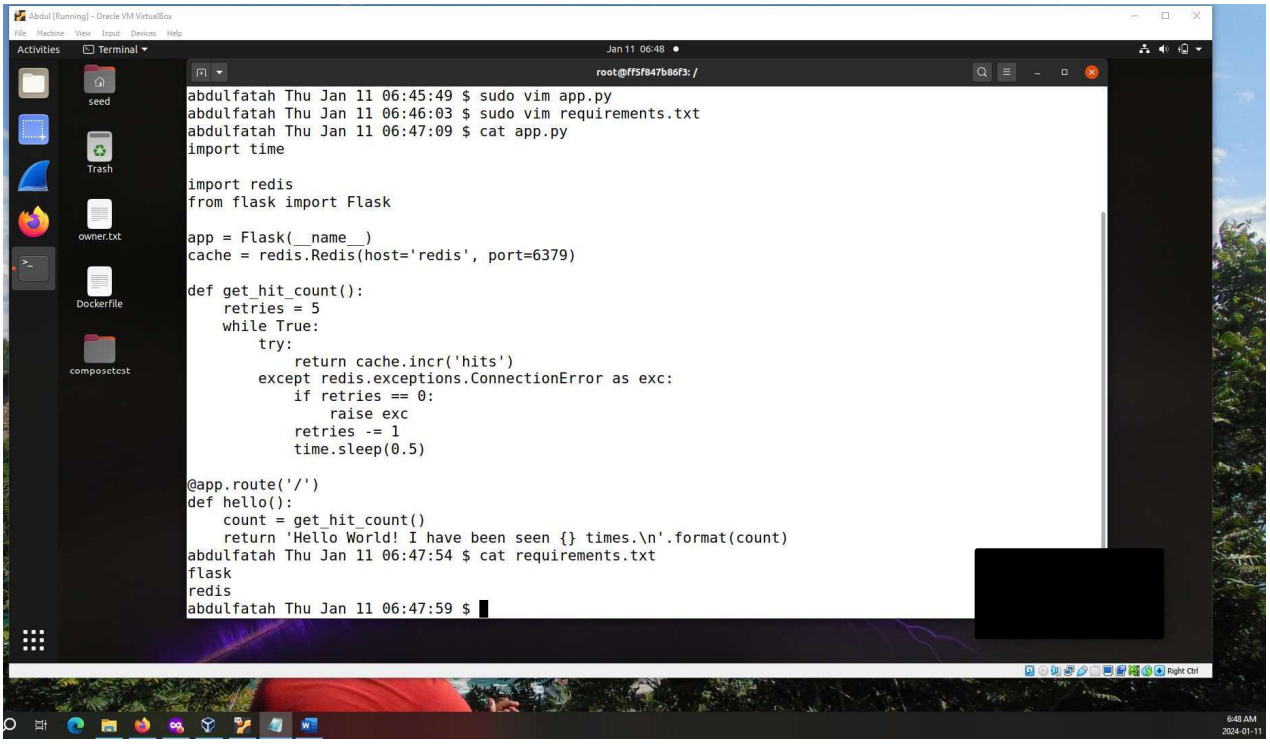


Lastly, I ran the command “docker run --net seednetwork --ip 172.20.0.5 --rm -it seedhost ” to start an interactive session with the container and specify an Ip address from the *seedhost* network (we just created above). I then ran some commands (whoami, ls, ip --br a) to prove that my container was working like it was supposed to, including the assigned IP address and owner.txt file being copied.

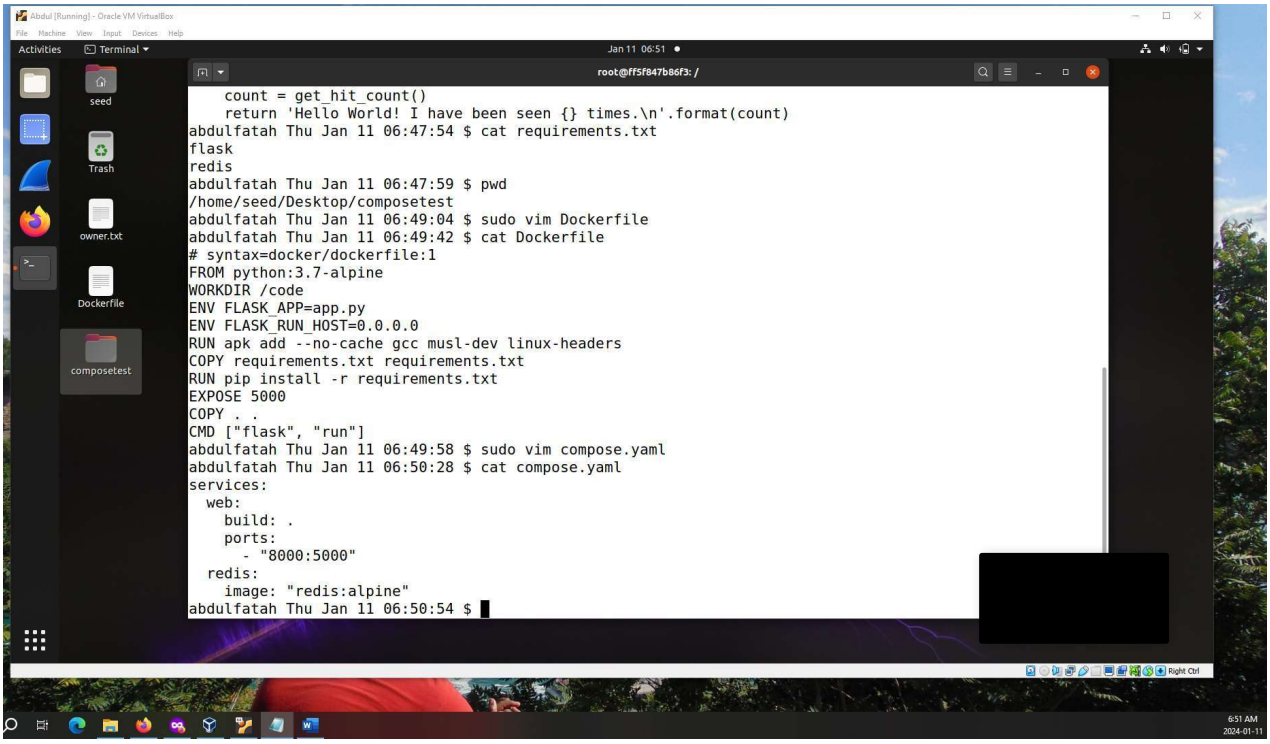


Docker Compose

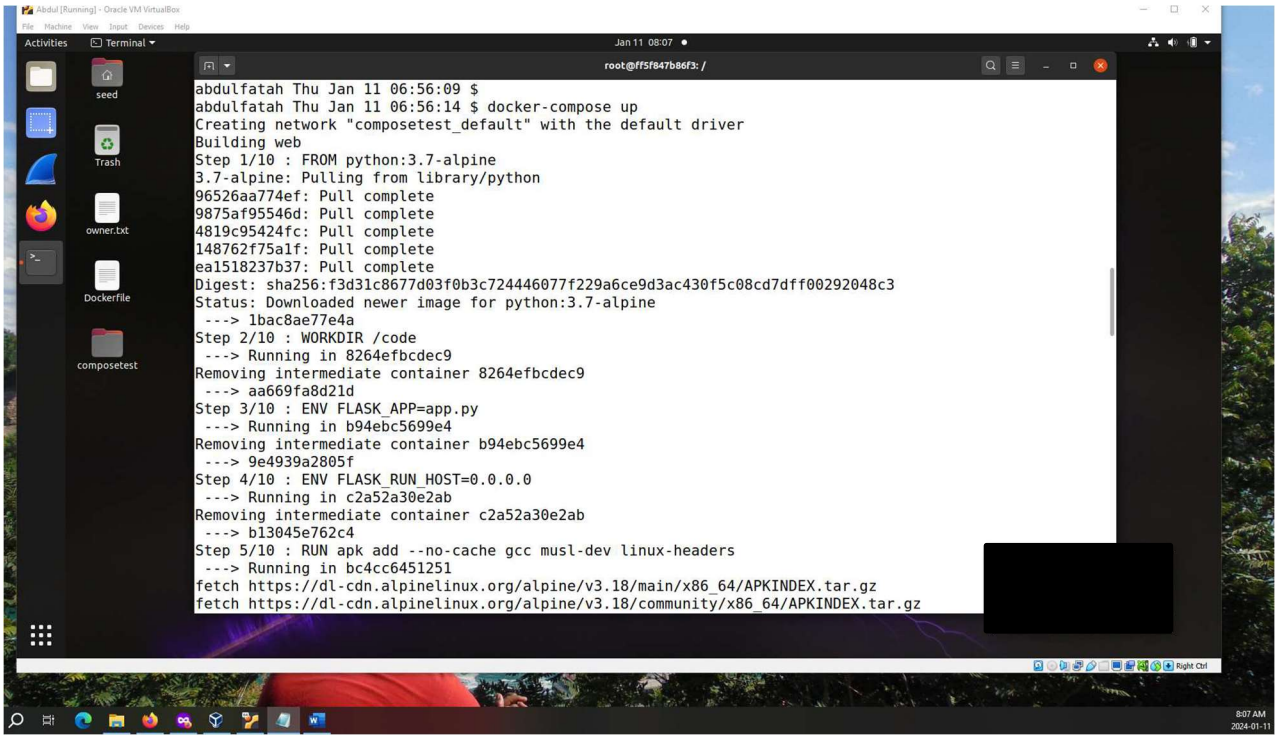
Here, we explore some concepts of docker compose by building a python web application. We first started creating a file named *app.py* which contained python code to create a simple Flask web application that uses Redis for caching. Also, I made another file called *requirements.txt* that contains the words “flask” and “redis”, which will come in handy when later creating the Dockerfile.



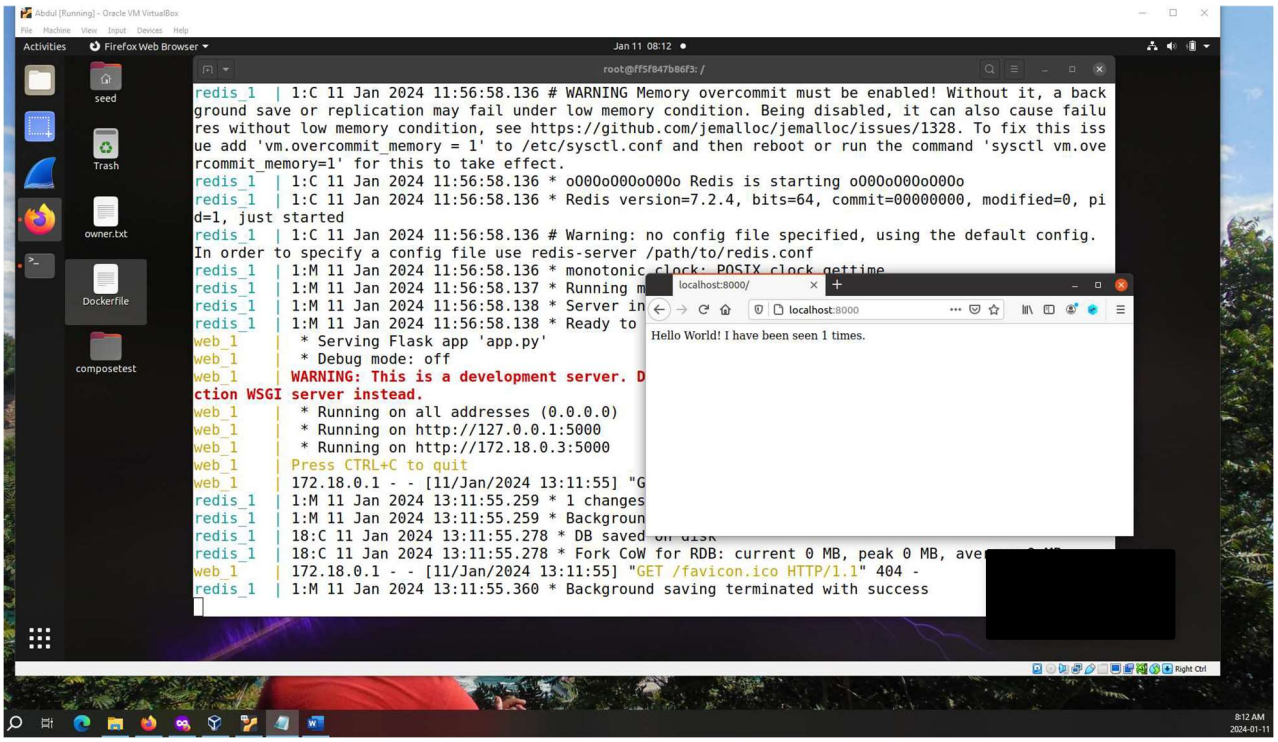
Here, I have created a Docker file which will be used to build a docker image. This file contains all the required dependencies needed for the proper functionality of a Python application, as well as the Python application itself. The services (web and redis) were later defined in a compose file named compose.yaml where the internal and exposed ports were configured and the Dockerfile image (alpine) was used.



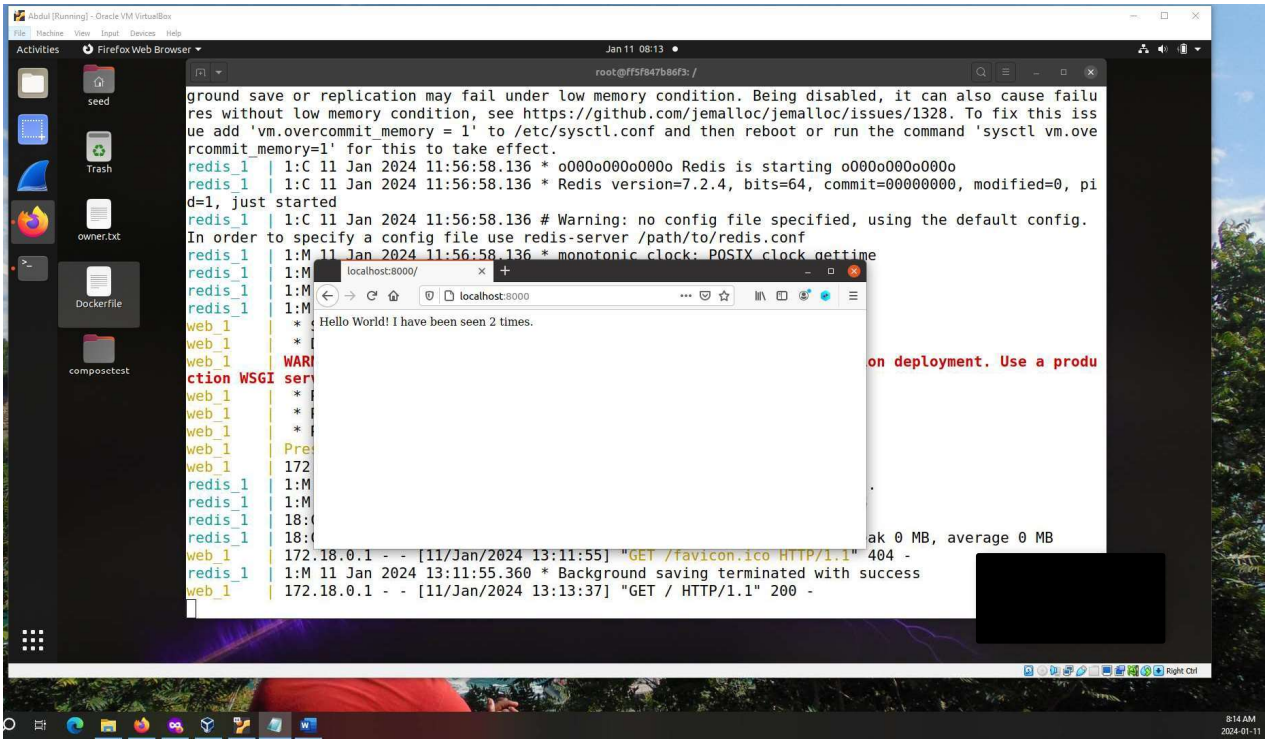
Here, I am starting the application by running the command “docker-compose up”.



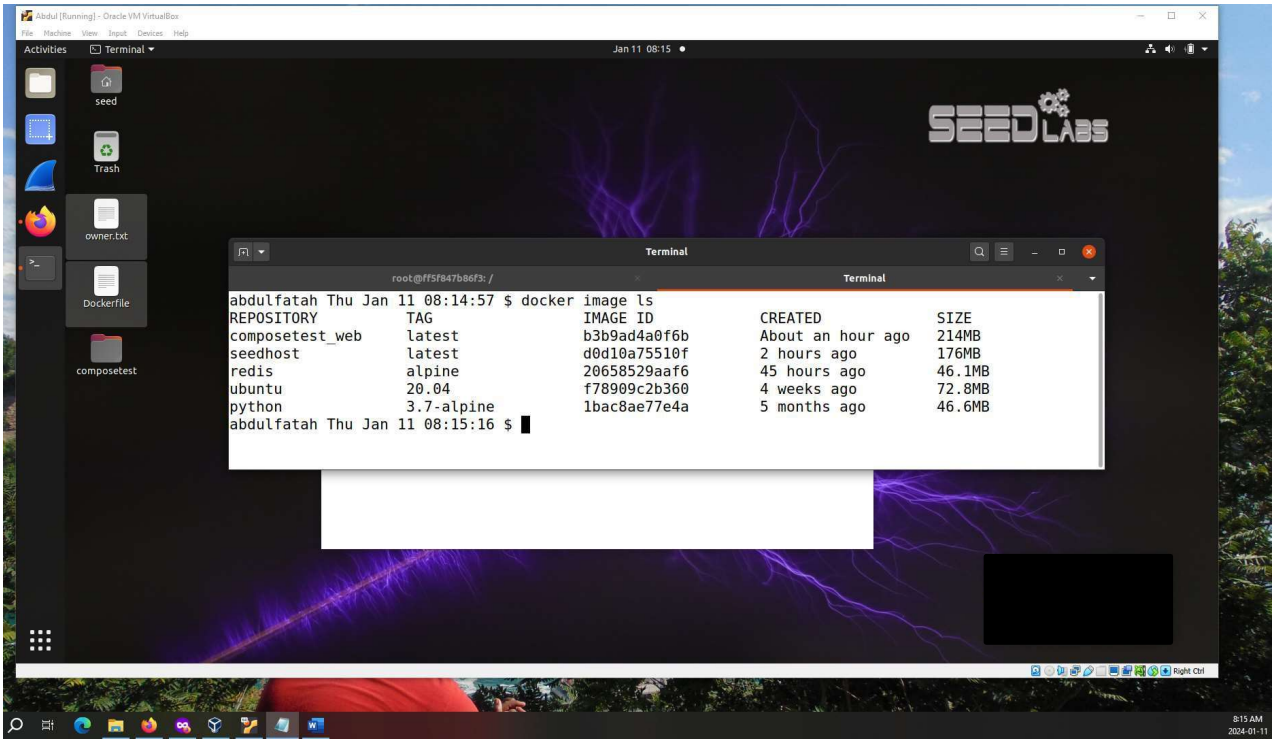
I follow this by browsing to port 8000 on the localhost and this shows that the application works as expected.



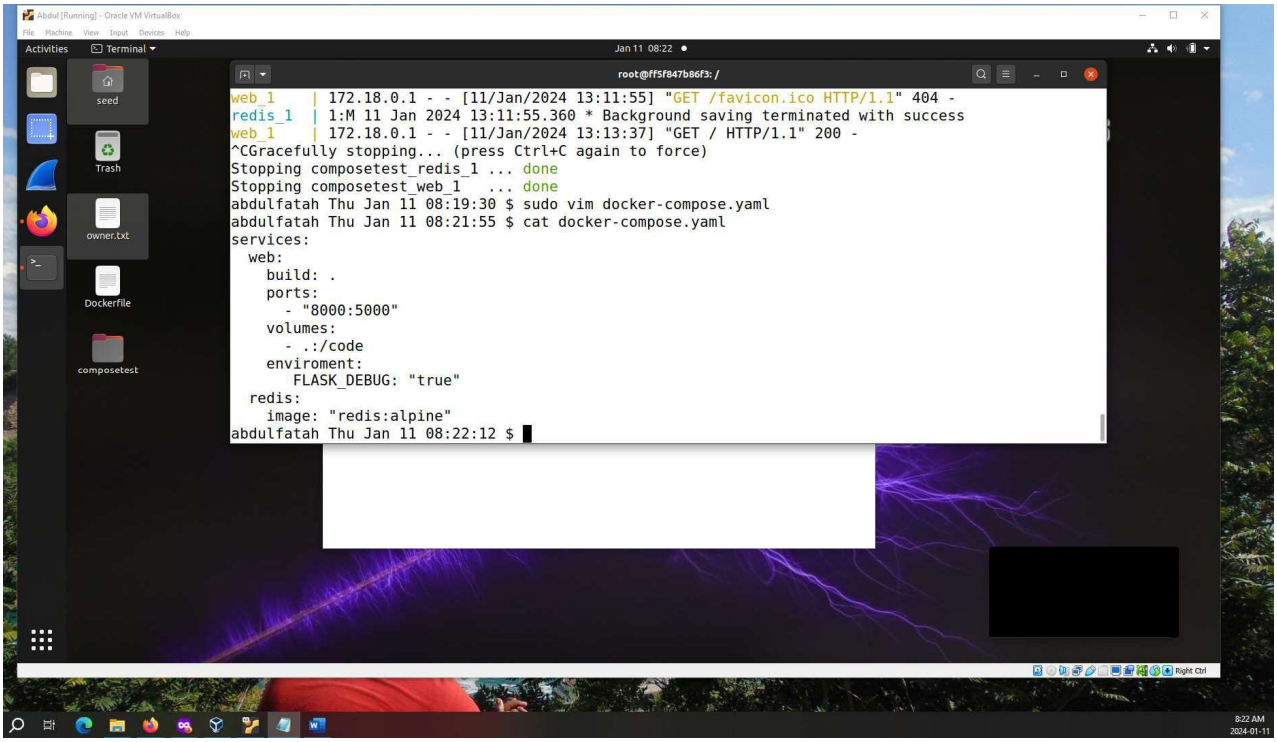
Upon refreshing the page, we see the increment in the number displayed in the message, proving that our Python code is working as we hoped.



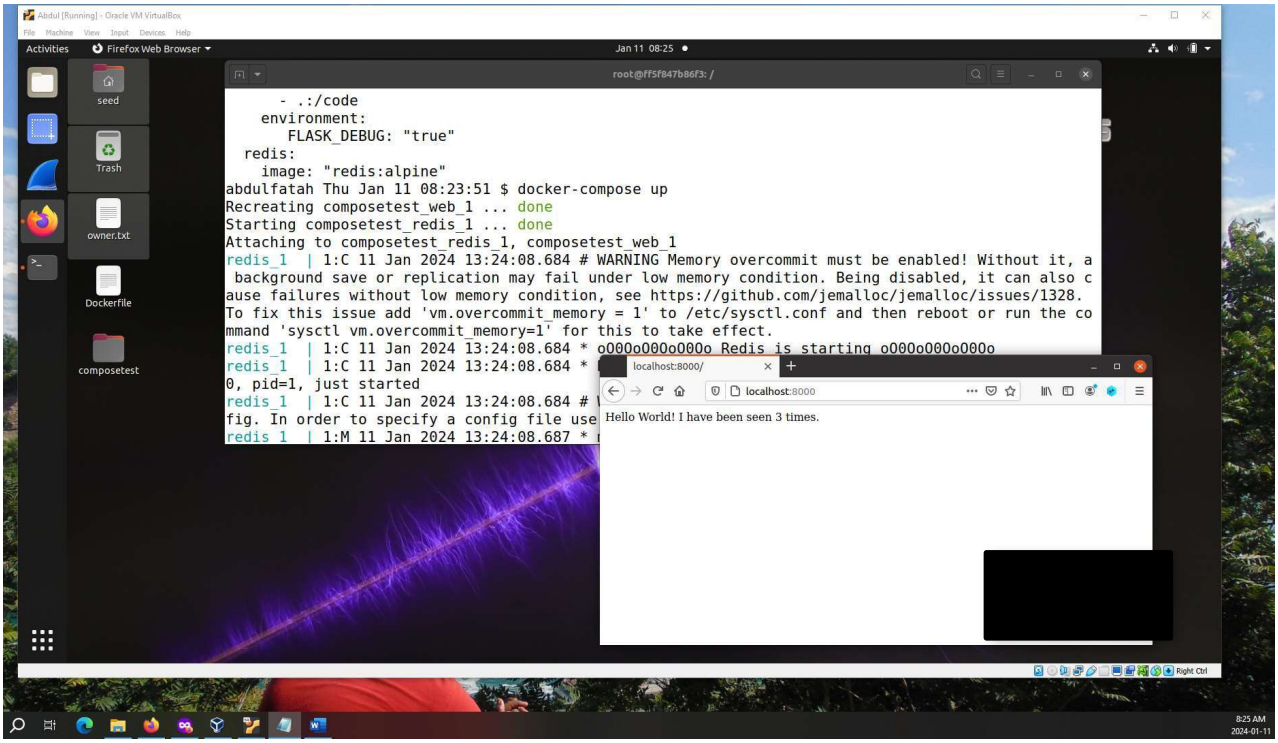
This is showing all the present docker images in the local repository up until this moment.



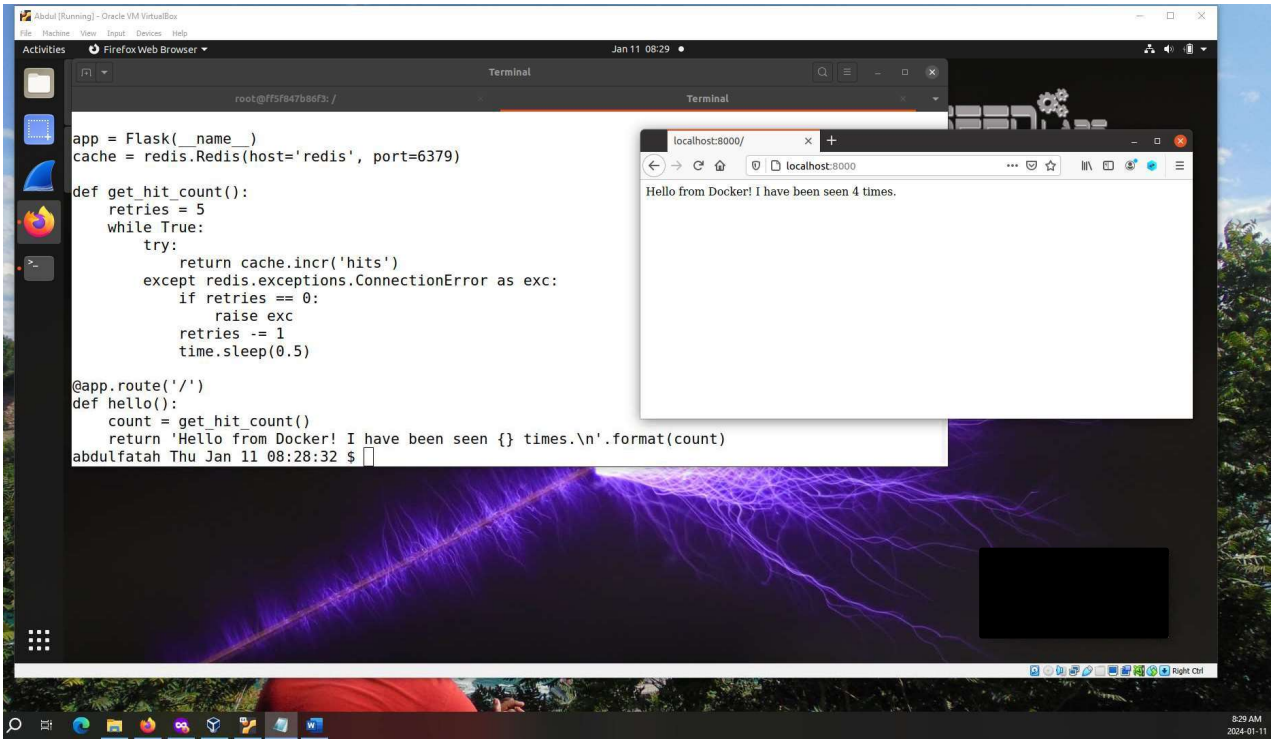
Here, I went back to editing the compose file to include a bind mount. This is used to mount the file on the host machine to a container.



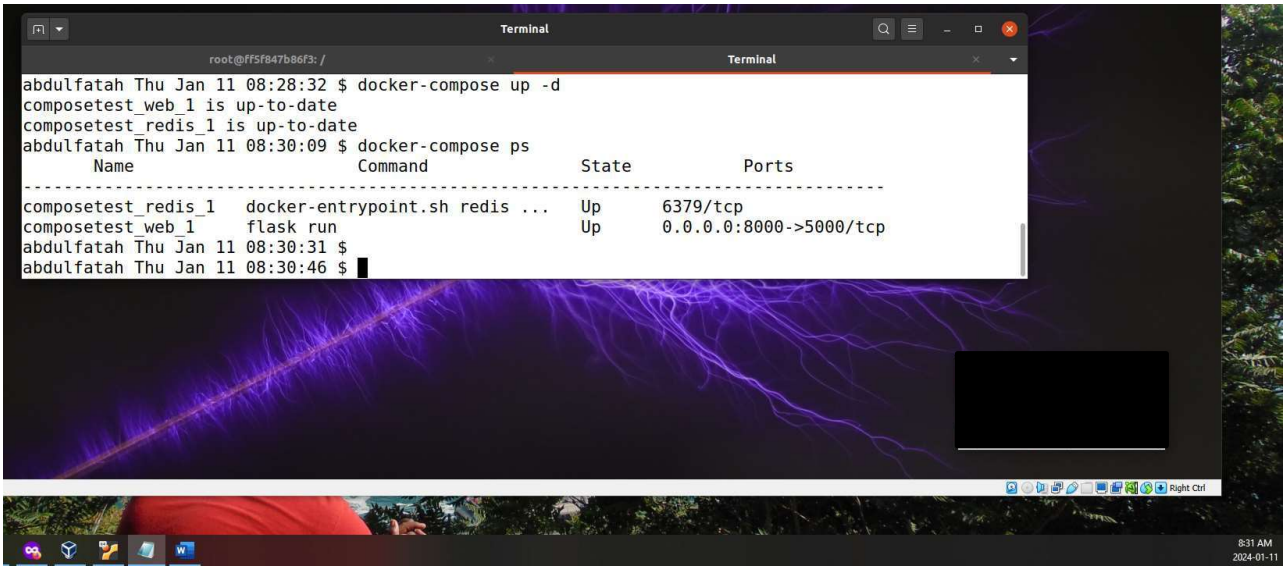
I am now running “docker-compose up” to build the app with the updated compose file. I then refresh the page and see the count increment.



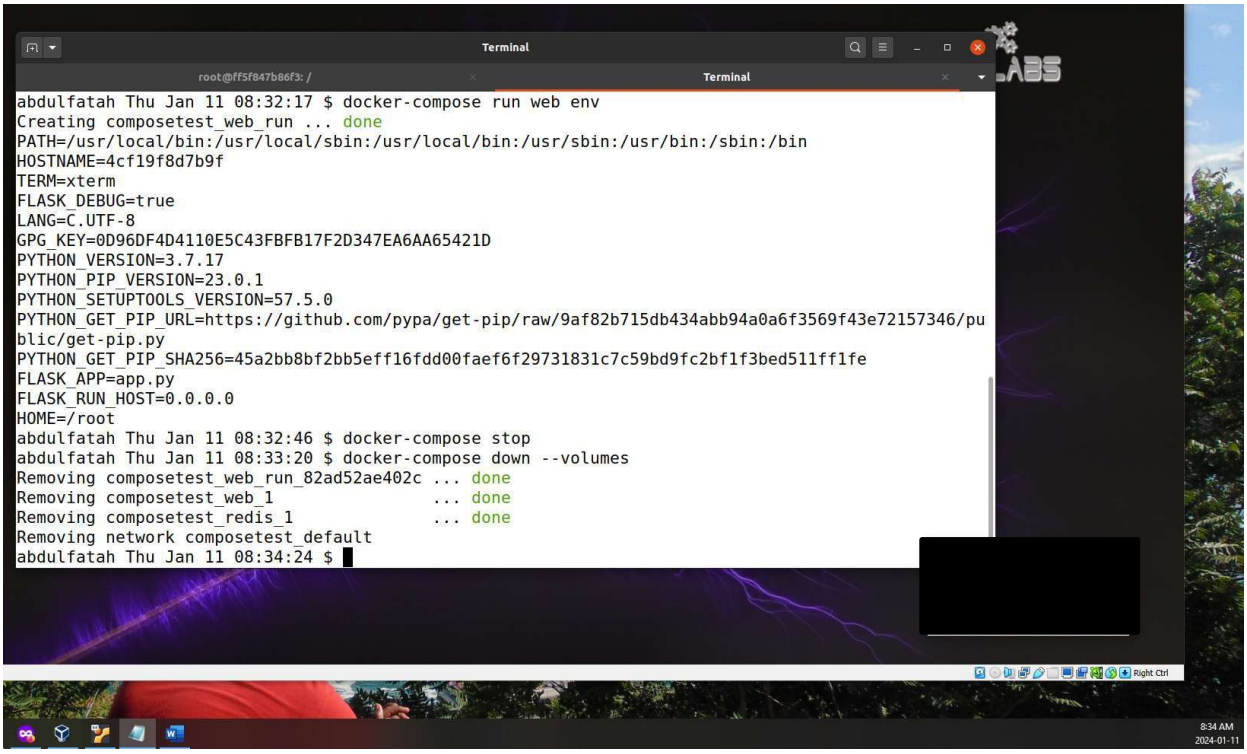
Here, the displayed message in the Python code was updated from “Hello World!” to “Hello from Docker!”. And since we have previously configured the bind mount, we are no longer required to update the image after making changes to the code as the changes will take place instantly.



Here, we use the -d flag with compose-up to run the services in detached mode and leave them in the background. I also ran the command “docker-compose ps” to see all any active services defined in the compose file.



Finally, I ran the command “docker compose run web env” to see what environmental variables are available on the web service. I then ran the command “docker compose stop” to eliminate all the services running in the background (in detached mode). Also, with the “docker compose down --volumes” command all containers have been completely removed and the data volume used by any service (Redis) is also removed.


Terminal
root@ff5f847b86f3: /
Terminal
abdulfatah Thu Jan 11 08:32:17 $ docker-compose run web env
Creating composetest_web_run ... done
PATH=/usr/local/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
HOSTNAME=4cf19f8d7b9f
TERM=xterm
FLASK_DEBUG=true
LANG=C.UTF-8
GPG_KEY=0D96DF4D4110E5C43FBFB17F2D347EA6AA65421D
PYTHON_VERSION=3.7.17
PYTHON_PIP_VERSION=23.0.1
PYTHON_SETUPTOOLS_VERSION=57.5.0
PYTHON_GET_PIP_URL=https://github.com/pypa/get-pip/raw/9af82b715db434abb94a0a6f3569f43e72157346/public/get-pip.py
PYTHON_GET_PIP_SHA256=45a2bb8bf2bb5eff16fdd00faef6f29731831c7c59bd9fc2bf1f3bed511ff1fe
FLASK_APP=app.py
FLASK_RUN_HOST=0.0.0.0
HOME=/root
abdulfatah Thu Jan 11 08:32:46 $ docker-compose stop
abdulfatah Thu Jan 11 08:33:20 $ docker-compose down --volumes
Removing composetest_web_run_82ad52ae402c ... done
Removing composetest_web_1 ... done
Removing composetest_redis_1 ... done
Removing network composetest_default
abdulfatah Thu Jan 11 08:34:24 $
Right Ctrl
8:34 AM
2024-01-11

References

[1] “seed-labs/seed-labs,” *GitHub*, Jan. 11, 2023. https://github.com/seed-labs/seed-labs/blob/master/manuals/docker/docker.md

[2] “Get started with Docker Compose,” *Docker Documentation*, Jul. 22, 2020. https://docs.docker.com/compose/gettingstarted/ (accessed Jan. 12, 2024).

[3] “seed-labs/seed-labs,” *GitHub*, Jan. 12, 2020. https://github.com/seed-labs/seed-labs/blob/master/manuals/vm/seedvm-manual.md (accessed Jan. 12, 2024).