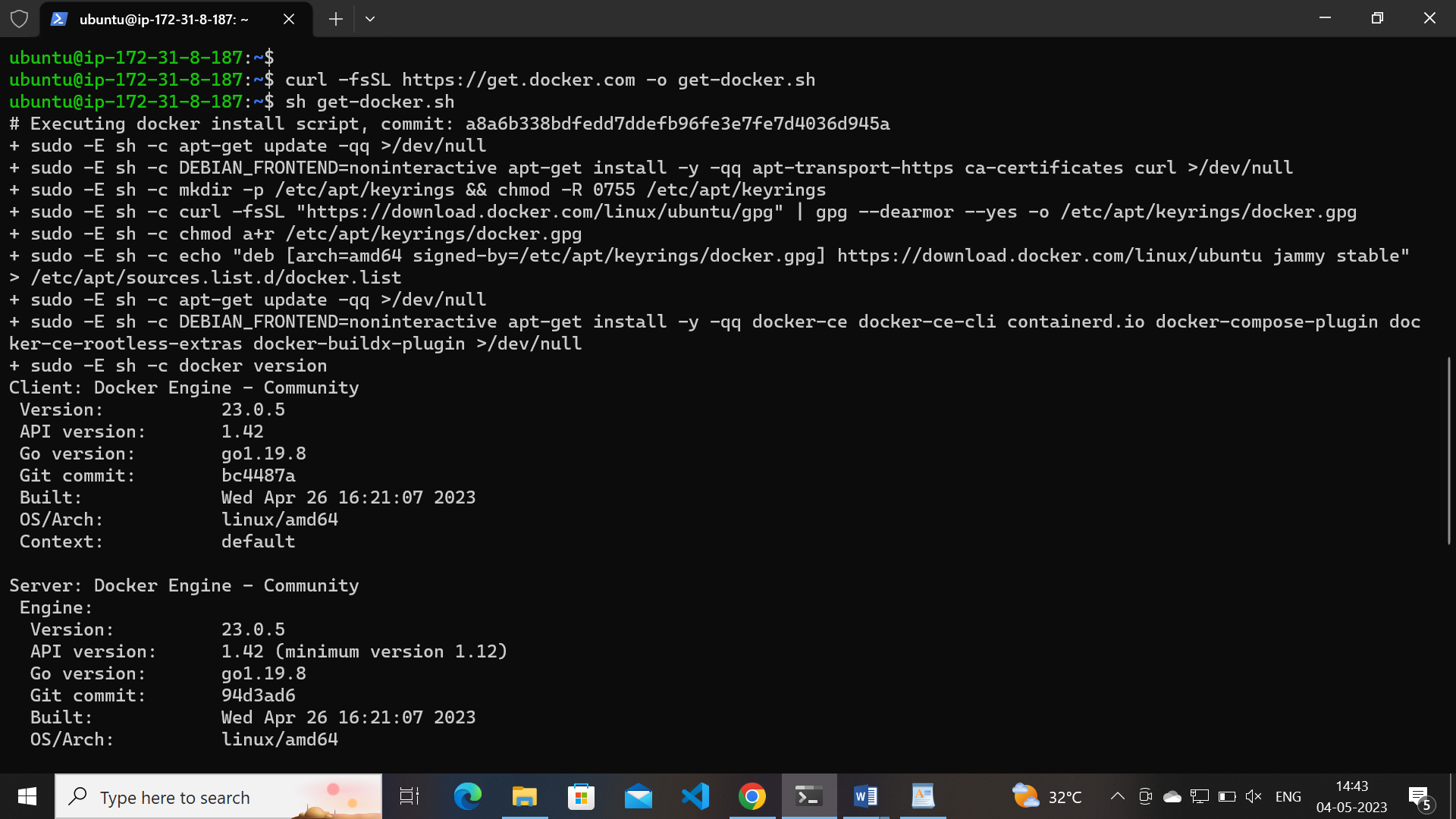
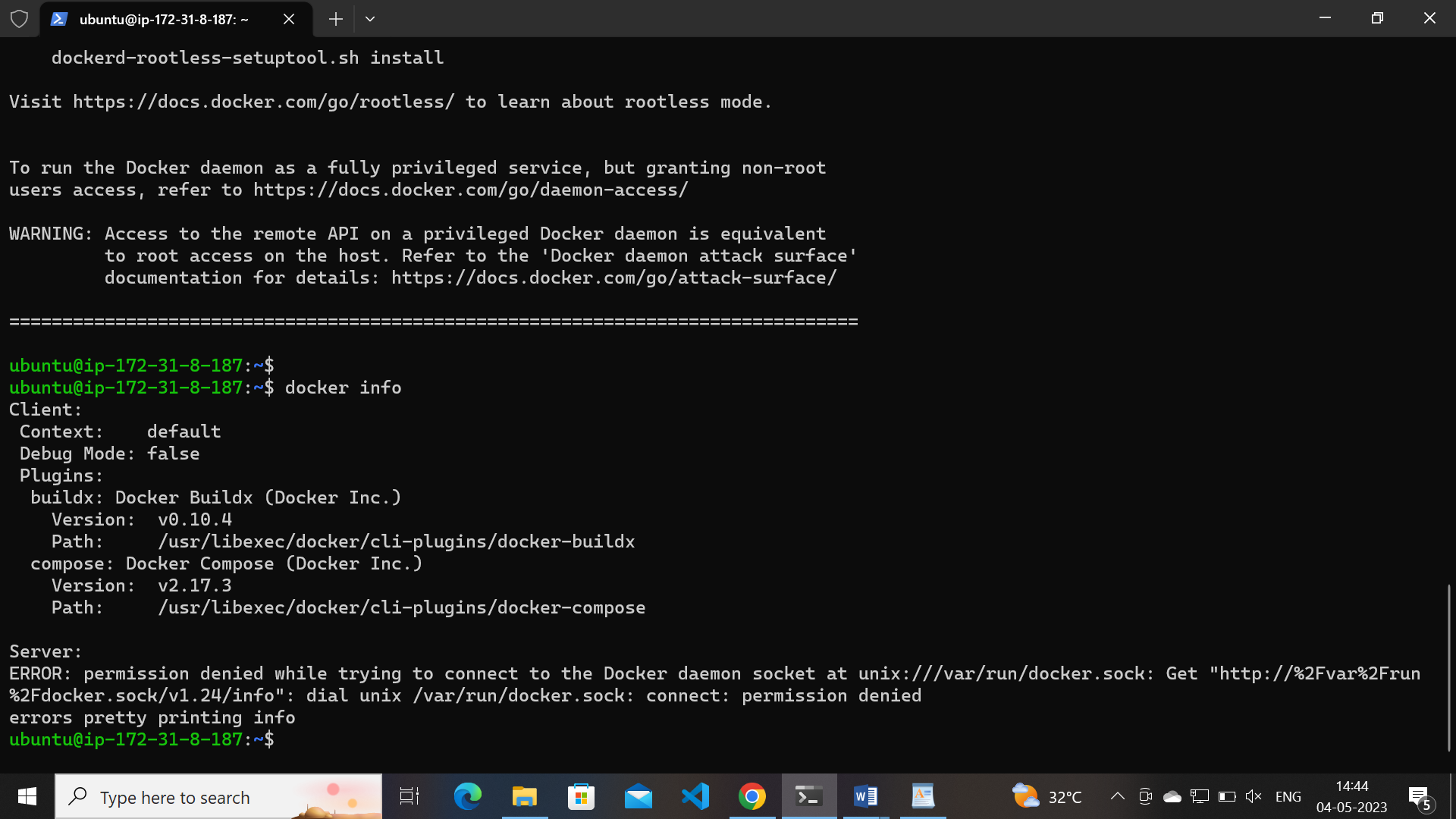
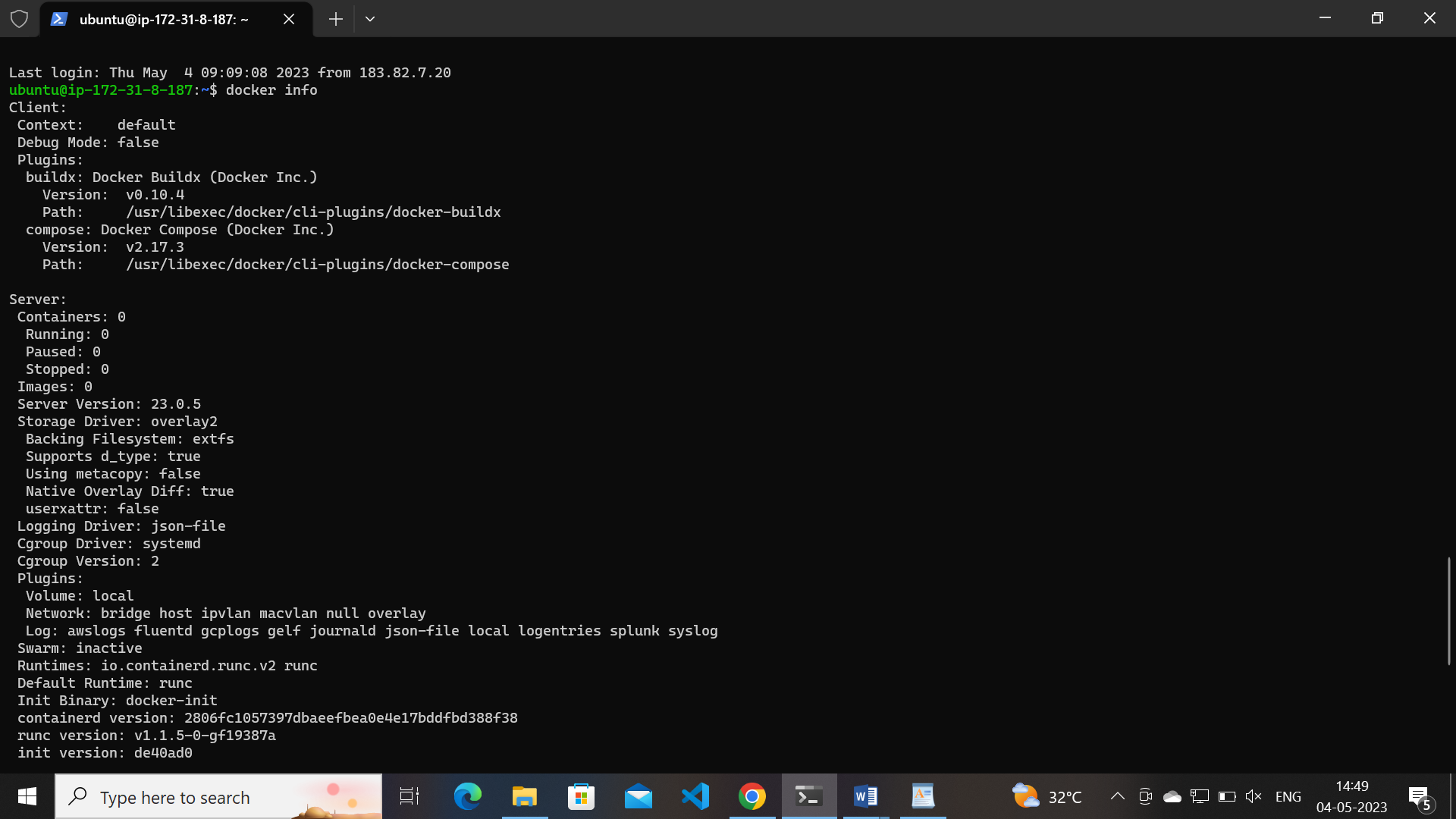
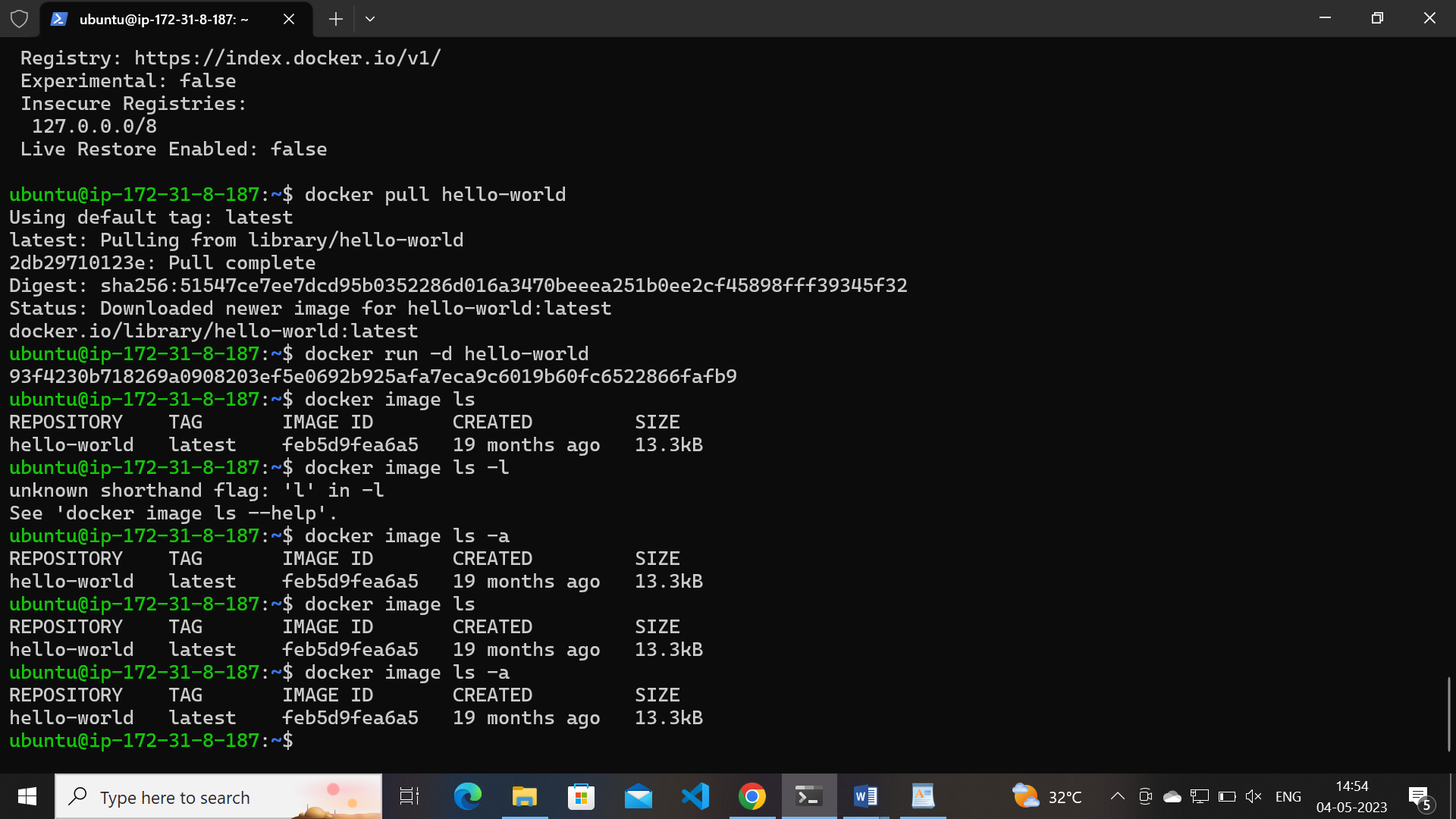
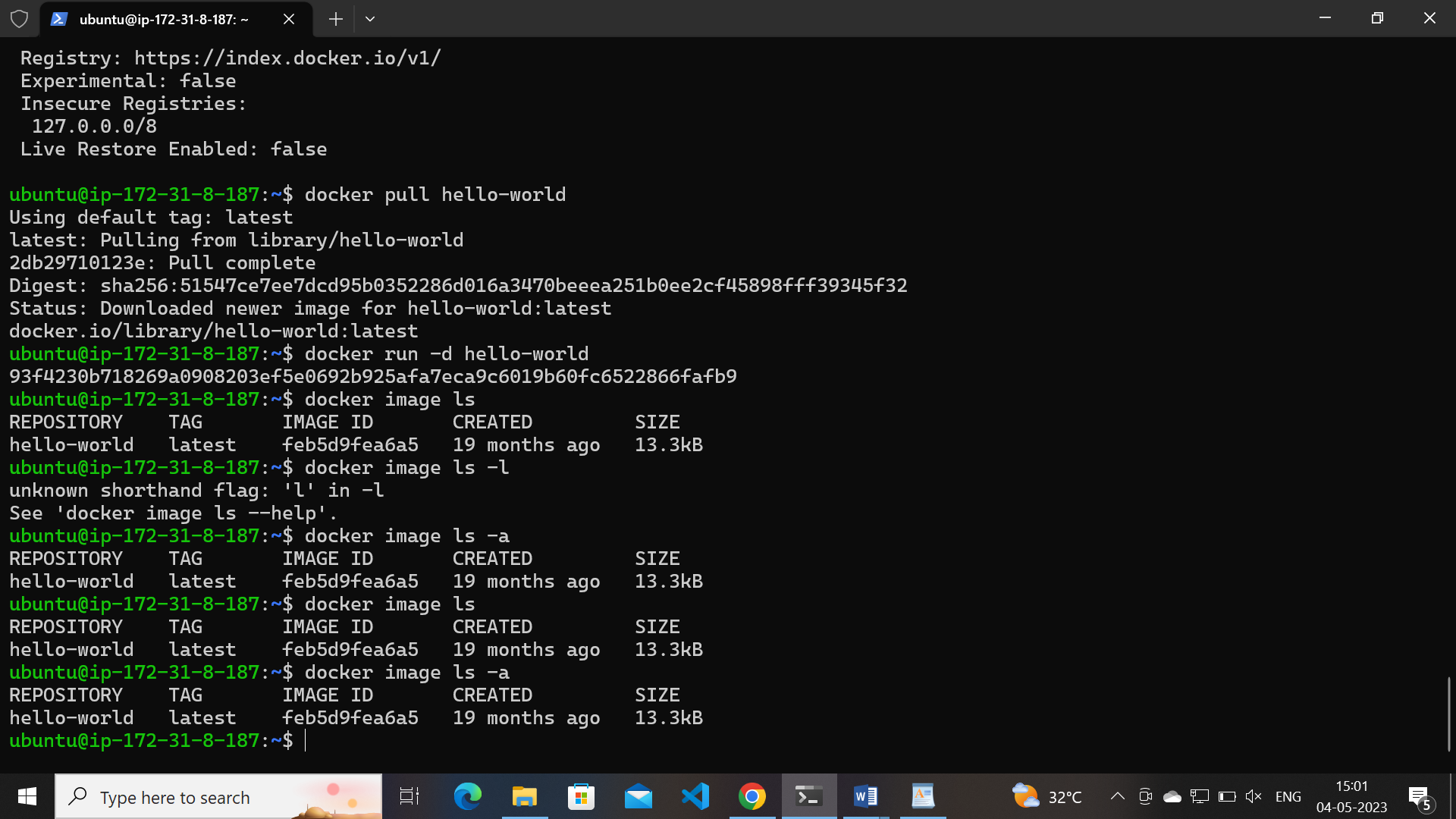
**Docker Practice - Workbook 1**

**======================================================================**

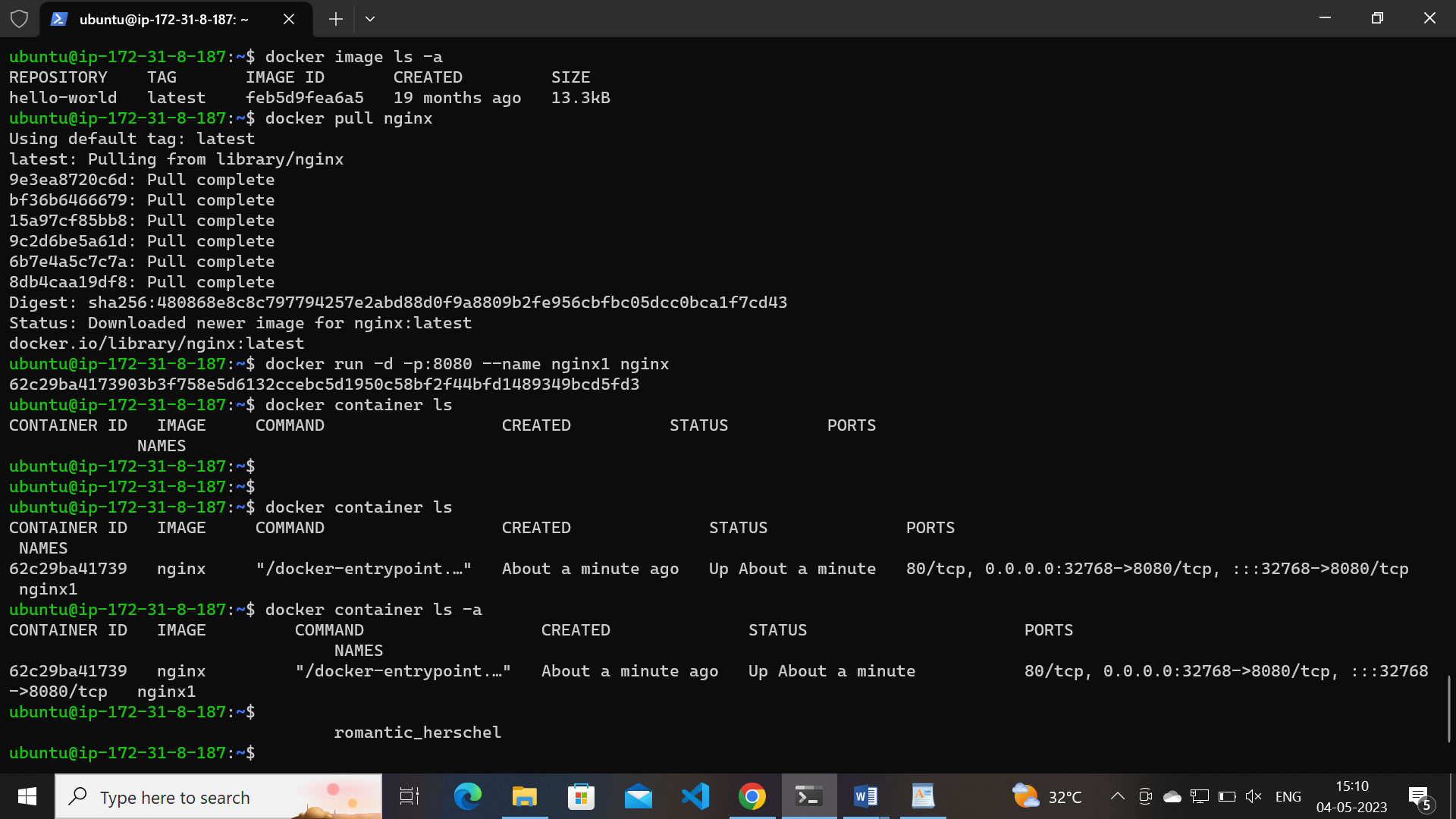
1. **Run hello-world docker container and observe the container status**

* **Login ec2 instance**
* **Script based installation from google**
* **curl -fsSL**[**https://get.docker.com**](https://get.docker.com/)**-o get-docker.sh**
* sh get-docker.sh
* 
* 
* Use command sudo usermod -aG docker <username>
* Exit and relogin the insatcnce check docker info
* 
* Installment done
* pull hello world image from docker hub
* docker pull hello-world
* docker run -d hello-world –to run helloworld container
* to check status docker container ls
* 

**2. Check the docker images and also write down the size of hello-world image**

* 
* Docker image size by command: docker image ls
* Image size 13.3kb

**3. Run the nginx container with name as nginx1 and expose it on 8080 port on docker host**

* Pull image from docker hub : docker pull nginx
* Run image with name nginx1 port:8080 :
* docker run -d -p:8080 --name nginx1 nginx
* 
* P indicates the port number
* d indicates detachmode

**4. Explain docker container lifecycle**

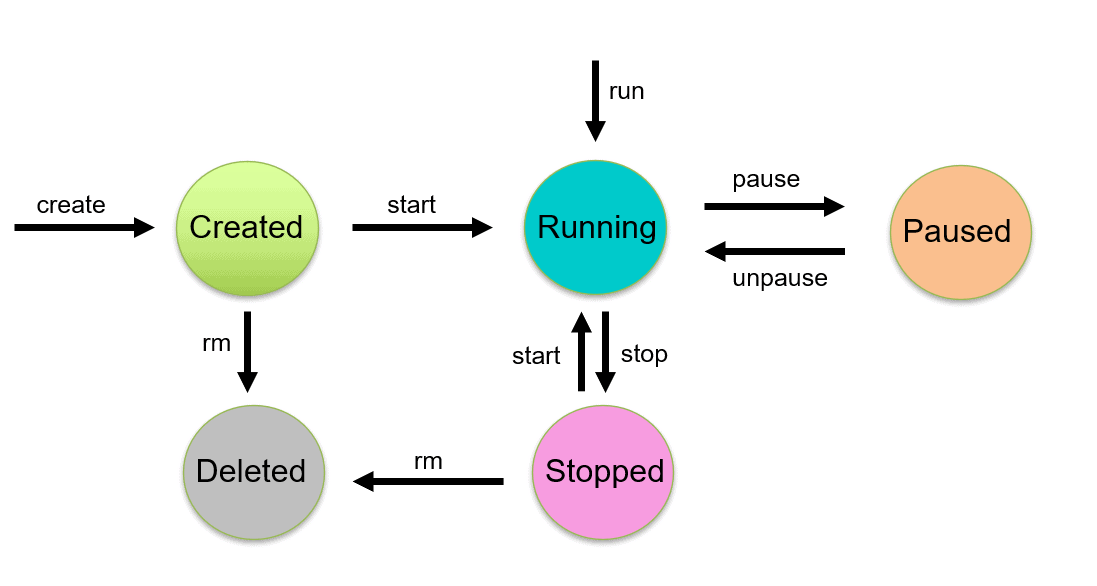
* we have steps in docker container lifecycle from creation to delete container
* from creation to delete or end of container b/w these is called as container life cycle
* steps involved in docker container life cycle
  + Created
  + Running
  + Paused
  + Stopped
  + Deleted
* **Created state**: In the docker image saved in the container state called as created state, image pulled from docker hub repository with out start the state is called created state
* Using docker container create command we can create container by giving image name
* To create container
* Usage: docker container create [OPTIONS] IMAGE [COMMAND] [ARG...]
* **Running:** After image pull while execute run command container image start the service is called as running state
* Usage: docker container run [OPTIONS] IMAGE [COMMAND] [ARG...]
* **Paused state:** while running the container we can pause container running state went paused state
* In this state container just paused(Pause all processes within one or more containers)
* unpause Unpause all processes within one or more containers
* Usage: docker container pause CONTAINER [CONTAINER...]
* By using unpause we can change the state

**Stop and start container :**

* While running the container we can stop and start the container by using start and stop command
* while executing stop command the container state went stopped the service its effected
* while executing start command the container state went start the service started
* Usage: docker container start [OPTIONS] CONTAINER [CONTAINER...]( Start one or more stopped containers)
* Usage: docker container stop [OPTIONS] CONTAINER [CONTAINER...]( Stop one or more running containers)

**Deleted**

* rm is the command Remove one or more containers
* After remove container docker totally switched off and not giving any service



**5. Explain what happens when you run the docker container**

* The docker run command creates running containers from images and can run commands inside them.
* When using the docker run command, a container can run a default action (if it has one), a user specified action, or a shell to be used interactively.
* The docker pull command downloads Docker images from the internet, From docker hub
* The docker image command lists Docker images that are (now) on your computer
* So far, we’ve seen how to download Docker images, use them to run commands inside running containers, and even how to explore a running container from the inside. Next, we’ll take a closer look at all the different kinds of Docker images that are out there.

**6. Explain the Docker Architecture**

## Docker architecture

Docker uses a client-server architecture. The Docker client talks to the Docker daemon, which does the heavy lifting of building, running, and distributing your Docker containers. The Docker client and daemon can run on the same system, or you can connect a Docker client to a remote Docker daemon. The Docker client and daemon communicate using a REST API, over UNIX sockets or a network interface. Another Docker client is Docker Compose, that lets you work with applications consisting of a set of containers.

