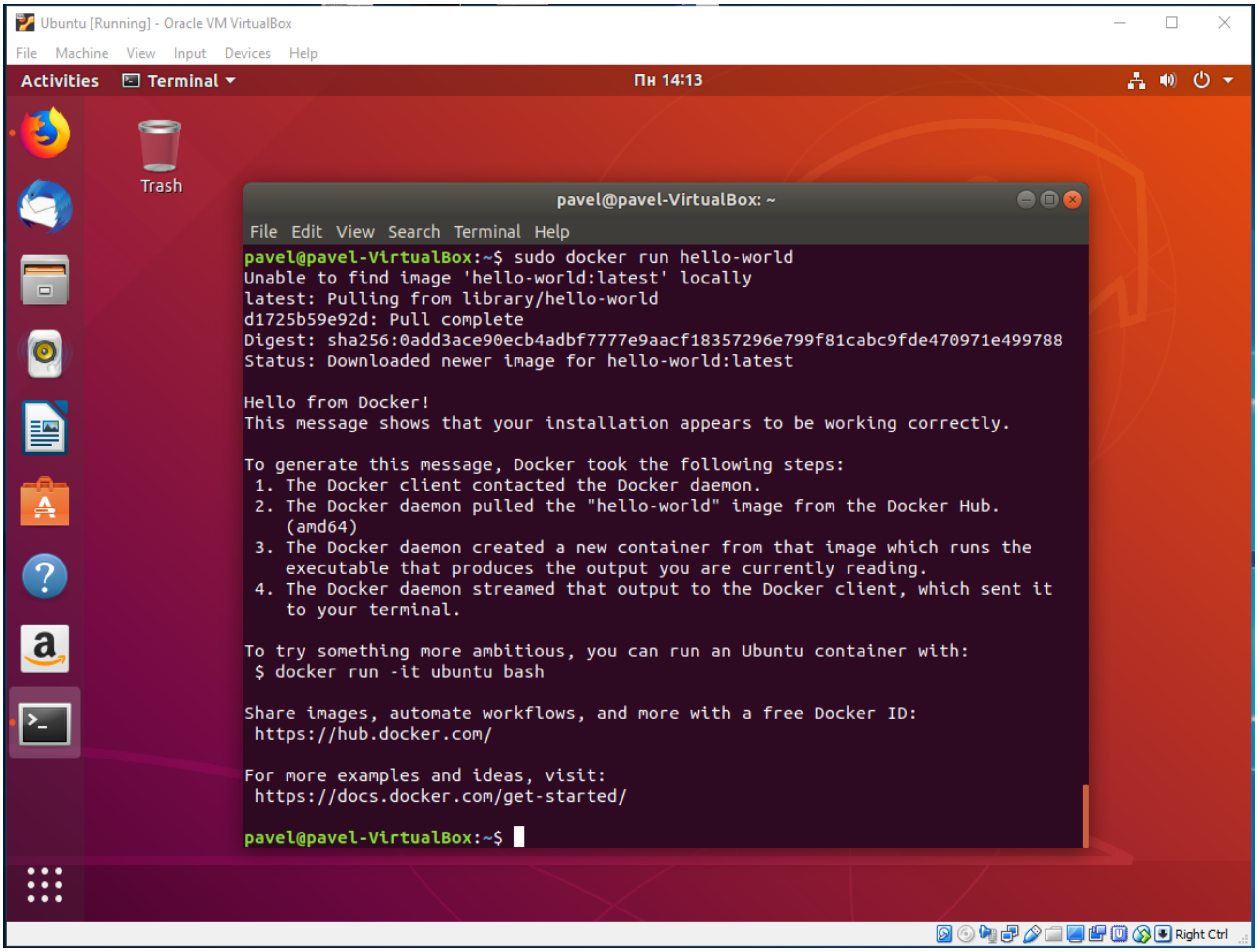
Docker homework.

1) Install docker on your machine

Here is docker running hello-world on my machine:

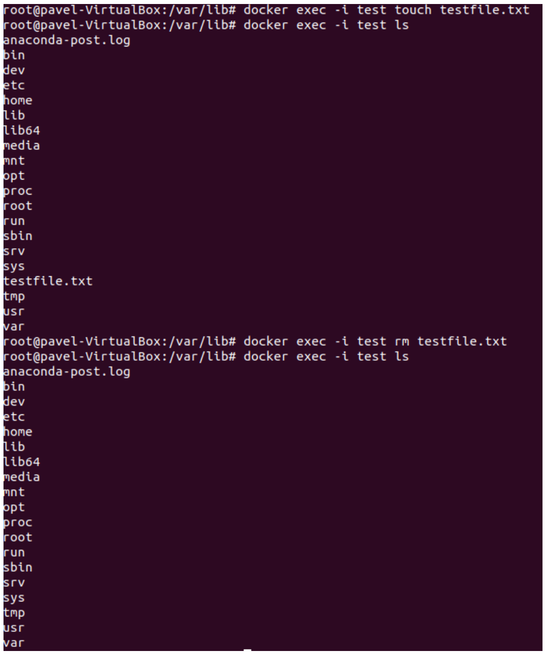

Ubuntu [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities
Terminal
Пн 14:13
Trash
pavel@pavel-VirtualBox: ~
File Edit View Search Terminal Help
Right Ctrl

```
pavel@pavel-VirtualBox:~$ sudo docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
d1725b59e92d: Pull complete
Digest: sha256:0add3ace90ecb4adbf7777e9aacf18357296e799f81cabc9fde470971e499788
Status: Downloaded newer image for hello-world:latest

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (amd64)
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
 $ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
 https://hub.docker.com/

For more examples and ideas, visit:
 https://docs.docker.com/get-started/

pavel@pavel-VirtualBox:~$ 
```

2) Create, edit, delete files within a docker container.

In order to do all of this, we need to use the command docker exec -i container_name our_command

This is how you can create/delete a text file in a docker container:



```
root@pavel-VirtualBox:/var/lib# docker exec -i test touch testfile.txt
root@pavel-VirtualBox:/var/lib# docker exec -i test ls
anaconda-post.log
bin
dev
etc
home
lib
lib64
media
mnt
opt
proc
root
run
sbin
srv
sys
testfile.txt
tmp
usr
var
root@pavel-VirtualBox:/var/lib# docker exec -i test rm testfile.txt
root@pavel-VirtualBox:/var/lib# docker exec -i test ls
anaconda-post.log
bin
dev
etc
home
lib
lib64
media
mnt
opt
proc
root
run
sbin
srv
sys
tmp
usr
var
```

Or you can instead log into the interactive mode by doing docker exec -it container_name bash and execute commands from there.

In order to change a file, you can log into the interactive mode and do vim text_file.txt

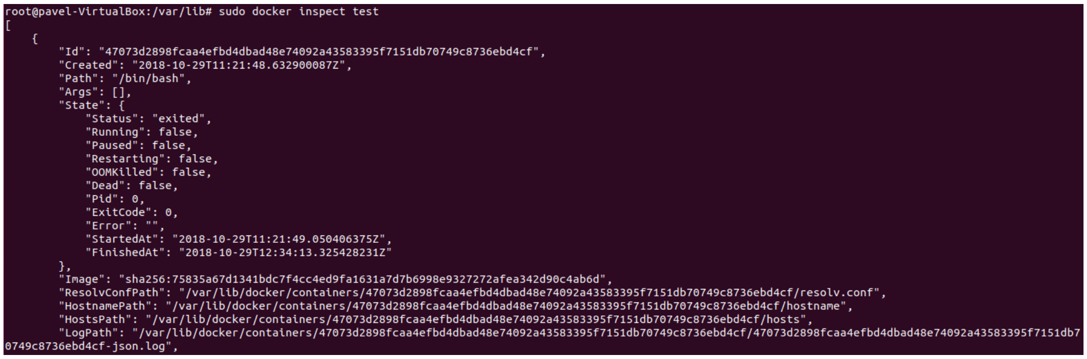
3) Find running docker container log file location by reading configuration (read log file and make screenshot).

All the docker container logs are located at:

/var/lib/docker/containers/<container-id>/<container-id>-json.log

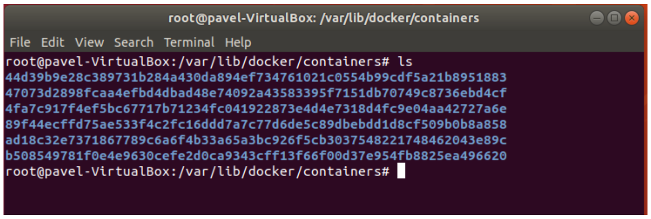
This can be verified by executing the following command:

sudo docker inspect test



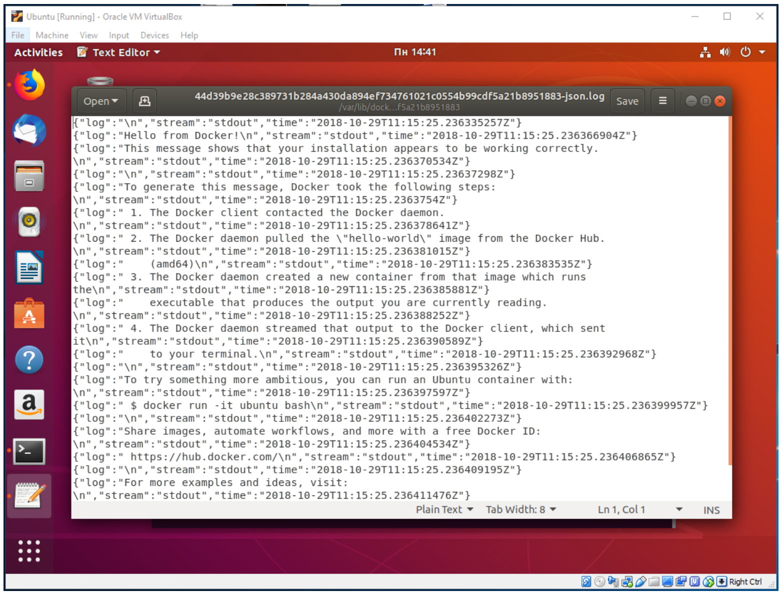
```
root@pavel-VirtualBox:/var/lib# sudo docker inspect test
[
    {
        "Id": "47073d2898fcaa4efbd4dbad48e74092a43583395f7151db70749c8736ebd4cf",
        "Created": "2018-10-29T11:21:48.632900087Z",
        "Path": "/bin/bash",
        "Args": [],
        "State": {
            "Status": "exited",
            "Running": false,
            "Paused": false,
            "Restarting": false,
            "OOMKilled": false,
            "Dead": false,
            "Pid": 0,
            "ExitCode": 0,
            "Error": "",
            "StartedAt": "2018-10-29T11:21:49.050406375Z",
            "FinishedAt": "2018-10-29T12:34:13.325428231Z"
        },
        "Image": "sha256:75835a67d1341bdc7f4cc4ed9fa1631a7d7b6998e9327272afea342d90c4ab6d",
        "ResolvConfPath": "/var/lib/docker/containers/47073d2898fcaa4efbd4dbad48e74092a43583395f7151db70749c8736ebd4cf/resolv.conf",
        "HostnamePath": "/var/lib/docker/containers/47073d2898fcaa4efbd4dbad48e74092a43583395f7151db70749c8736ebd4cf/hostname",
        "HostsPath": "/var/lib/docker/containers/47073d2898fcaa4efbd4dbad48e74092a43583395f7151db70749c8736ebd4cf/hosts",
        "LogPath": "/var/lib/docker/containers/47073d2898fcaa4efbd4dbad48e74092a43583395f7151db70749c8736ebd4cf/47073d2898fcaa4efbd4dbad48e74092a43583395f7151db7
0749c8736ebd4cf-json.log",
```

Let’s navigate there and we will see:


root@pavel-VirtualBox: /var/lib/docker/containers
File Edit View Search Terminal Help

```
root@pavel-VirtualBox:/var/lib/docker/containers# ls
44d39b9e28c389731b284a430da894ef734761021c0554b99cdf5a21b8951883
47073d2898fcaa4efbd4dbad48e74092a43583395f7151db70749c8736ebd4cf
4fa7c917f4ef5bc67717b71234fc041922873e4d4e7318d4fc9e04aa42727a6e
89f44ecffd75ae533f4c2fc16ddd7a7c77d6de5c89dbebdd1d8cf509b0b8a858
ad18c32e7371867789c6a6f4b33a65a3bc926f5cb3037548221748462043e89c
b508549781f0e4e9630cefe2d0ca9343cff13f66f00d37e954fb8825ea496620
root@pavel-VirtualBox:/var/lib/docker/containers# 
```

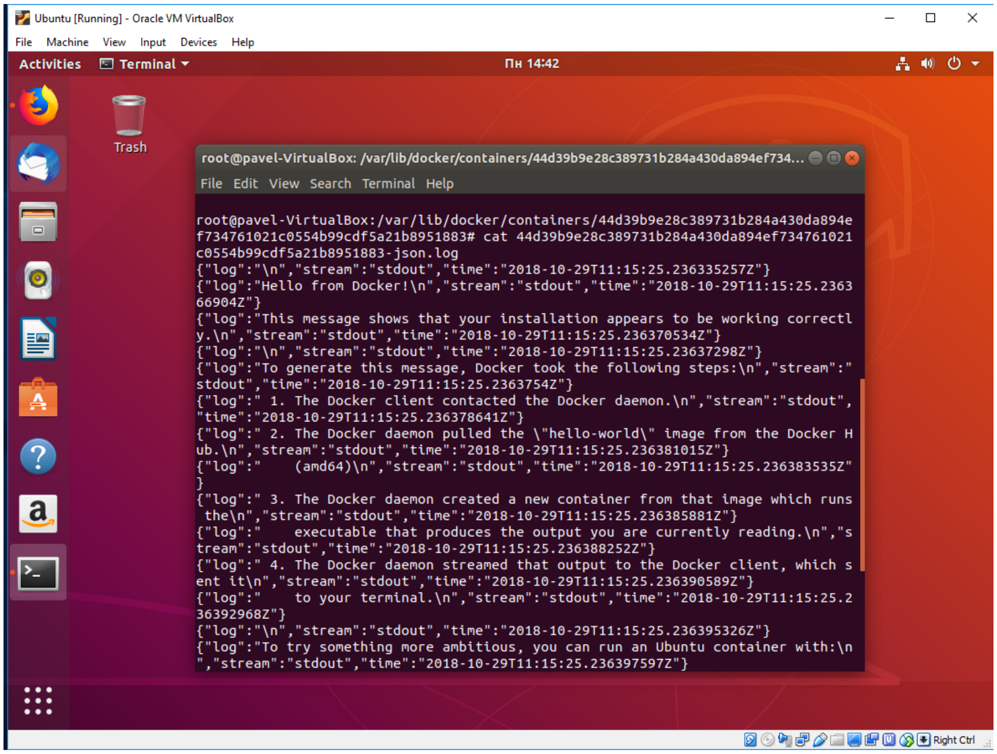
Let’s view the log file for 44d3… which is a container of the hello-world image:


Ubuntu [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities
Text Editor
Пн 14:41
Open
44d39b9e28c389731b284a430da894ef734761021c0554b99cdf5a21b8951883-json.log
/var/lib/dock...f5a21b8951883
Save
Plain Text
Tab Width: 8
Ln 1, Col 1
INS
Right Ctrl

```
{"log":"\n","stream":"stdout","time":"2018-10-29T11:15:25.236335257Z"}
{"log":"Hello from Docker!\n","stream":"stdout","time":"2018-10-29T11:15:25.236366904Z"}
{"log":"This message shows that your installation appears to be working correctly.
\n","stream":"stdout","time":"2018-10-29T11:15:25.236370534Z"}
{"log":"\n","stream":"stdout","time":"2018-10-29T11:15:25.23637298Z"}
{"log":"To generate this message, Docker took the following steps:
\n","stream":"stdout","time":"2018-10-29T11:15:25.2363754Z"}
{"log":" 1. The Docker client contacted the Docker daemon.
\n","stream":"stdout","time":"2018-10-29T11:15:25.236378641Z"}
{"log":" 2. The Docker daemon pulled the \"hello-world\" image from the Docker Hub.
\n","stream":"stdout","time":"2018-10-29T11:15:25.236381015Z"}
{"log":"    (amd64)\n","stream":"stdout","time":"2018-10-29T11:15:25.236383535Z"}
{"log":" 3. The Docker daemon created a new container from that image which runs
the\n","stream":"stdout","time":"2018-10-29T11:15:25.236385881Z"}
{"log":"    executable that produces the output you are currently reading.
\n","stream":"stdout","time":"2018-10-29T11:15:25.236388252Z"}
{"log":" 4. The Docker daemon streamed that output to the Docker client, which sent
it\n","stream":"stdout","time":"2018-10-29T11:15:25.236390589Z"}
{"log":"    to your terminal.\n","stream":"stdout","time":"2018-10-29T11:15:25.236392968Z"}
{"log":"\n","stream":"stdout","time":"2018-10-29T11:15:25.236395326Z"}
{"log":"To try something more ambitious, you can run an Ubuntu container with:
\n","stream":"stdout","time":"2018-10-29T11:15:25.236397597Z"}
{"log":" $ docker run -it ubuntu bash\n","stream":"stdout","time":"2018-10-29T11:15:25.236399957Z"}
{"log":"\n","stream":"stdout","time":"2018-10-29T11:15:25.236402273Z"}
{"log":"Share images, automate workflows, and more with a free Docker ID:
\n","stream":"stdout","time":"2018-10-29T11:15:25.236404534Z"}
{"log":" https://hub.docker.com/\n","stream":"stdout","time":"2018-10-29T11:15:25.236406865Z"}
{"log":"\n","stream":"stdout","time":"2018-10-29T11:15:25.236409195Z"}
{"log":"For more examples and ideas, visit:
\n","stream":"stdout","time":"2018-10-29T11:15:25.236411476Z"}
```

4) Read running docker container log file using cat command (make screenshot).

Executing the cat command, we get the same result:



Ubuntu [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities
Terminal
Пн 14:42
Trash
root@pavel-VirtualBox: /var/lib/docker/containers/44d39b9e28c389731b284a430da894ef734...
File Edit View Search Terminal Help
Right Ctrl

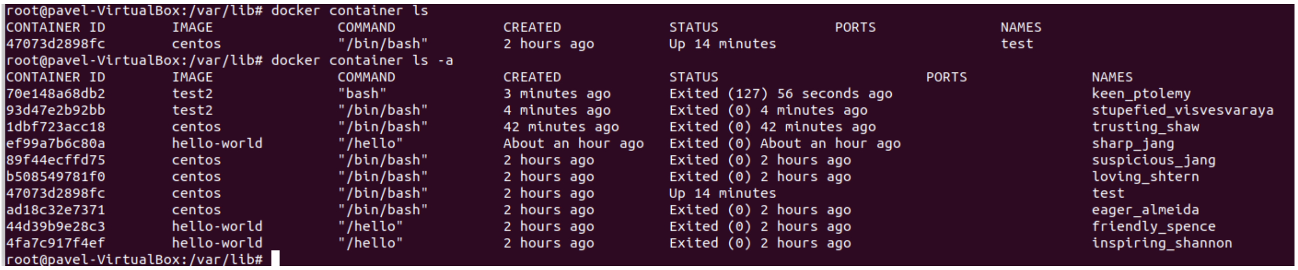
```
root@pavel-VirtualBox:/var/lib/docker/containers/44d39b9e28c389731b284a430da894e
f734761021c0554b99cdf5a21b8951883# cat 44d39b9e28c389731b284a430da894ef734761021
c0554b99cdf5a21b8951883-json.log
{"log":"\n","stream":"stdout","time":"2018-10-29T11:15:25.236335257Z"}
{"log":"Hello from Docker!\n","stream":"stdout","time":"2018-10-29T11:15:25.2363
66904Z"}
{"log":"This message shows that your installation appears to be working correctl
y.\n","stream":"stdout","time":"2018-10-29T11:15:25.236370534Z"}
{"log":"\n","stream":"stdout","time":"2018-10-29T11:15:25.23637298Z"}
{"log":"To generate this message, Docker took the following steps:\n","stream":"
stdout","time":"2018-10-29T11:15:25.2363754Z"}
{"log":" 1. The Docker client contacted the Docker daemon.\n","stream":"stdout",
"time":"2018-10-29T11:15:25.236378641Z"}
{"log":" 2. The Docker daemon pulled the \"hello-world\" image from the Docker H
ub.\n","stream":"stdout","time":"2018-10-29T11:15:25.236381015Z"}
{"log":"    (amd64)\n","stream":"stdout","time":"2018-10-29T11:15:25.236383535Z"
}
{"log":" 3. The Docker daemon created a new container from that image which runs
 the\n","stream":"stdout","time":"2018-10-29T11:15:25.236385881Z"}
{"log":"    executable that produces the output you are currently reading.\n","s
tream":"stdout","time":"2018-10-29T11:15:25.236388252Z"}
{"log":" 4. The Docker daemon streamed that output to the Docker client, which s
ent it\n","stream":"stdout","time":"2018-10-29T11:15:25.236390589Z"}
{"log":"    to your terminal.\n","stream":"stdout","time":"2018-10-29T11:15:25.2
36392968Z"}
{"log":"\n","stream":"stdout","time":"2018-10-29T11:15:25.236395326Z"}
{"log":"To try something more ambitious, you can run an Ubuntu container with:\n
","stream":"stdout","time":"2018-10-29T11:15:25.236397597Z"}
```

5) Add file to running docker container and save container. Next time docker container runs it should have file present. (make several screenshots).

In order to create a new image from a container you have to execute this command:

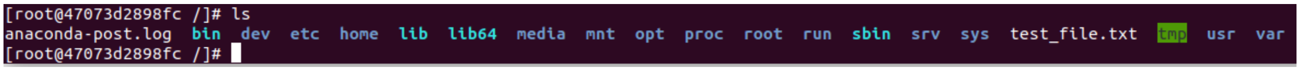
docker commit container_name new_image_name

Let's list all of my containers and their images:



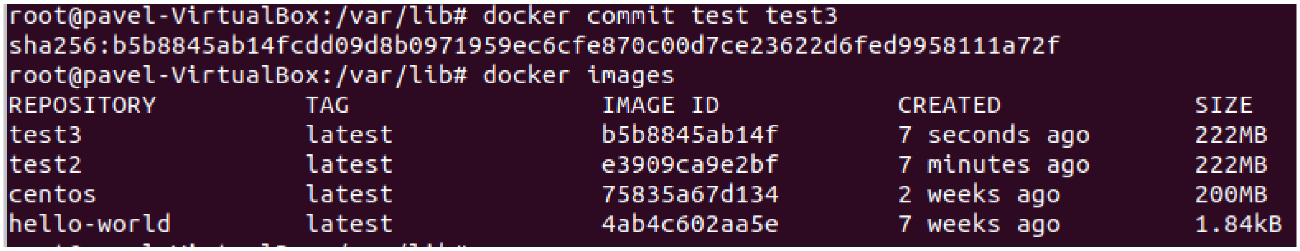
```
root@pavel-VirtualBox:/var/lib# docker container ls
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS               NAMES
47073d2898fc        centos              "/bin/bash"         2 hours ago         Up 14 minutes                           test
root@pavel-VirtualBox:/var/lib# docker container ls -a
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS                        PORTS               NAMES
70e148a68db2        test2               "bash"              3 minutes ago       Exited (127) 56 seconds ago                       keen_ptolemy
93d47e2b92bb        test2               "/bin/bash"         4 minutes ago       Exited (0) 4 minutes ago                          stupefied_visvesvaraya
1dbf723acc18        centos              "/bin/bash"         42 minutes ago      Exited (0) 42 minutes ago                         trusting_shaw
ef99a7b6c80a        hello-world         "/hello"            About an hour ago   Exited (0) About an hour ago                      sharp_jang
89f44ecffd75        centos              "/bin/bash"         2 hours ago         Exited (0) 2 hours ago                            suspicious_jang
b508549781f0        centos              "/bin/bash"         2 hours ago         Exited (0) 2 hours ago                            loving_shtern
47073d2898fc        centos              "/bin/bash"         2 hours ago         Up 14 minutes                                     test
ad18c32e7371        centos              "/bin/bash"         2 hours ago         Exited (0) 2 hours ago                            eager_almeida
44d39b9e28c3        hello-world         "/hello"            2 hours ago         Exited (0) 2 hours ago                            friendly_spence
4fa7c917f4ef        hello-world         "/hello"            2 hours ago         Exited (0) 2 hours ago                            inspiring_shannon
root@pavel-VirtualBox:/var/lib#
```

As you can see I have a container called "test", corresponding to the "centos" image. Let's look at it's file system:



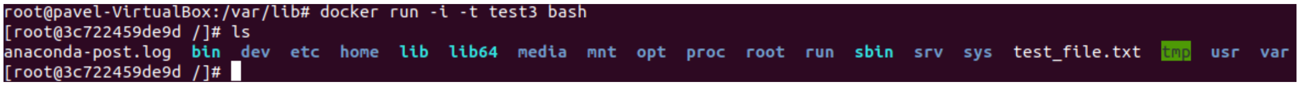
```
[root@47073d2898fc /]# ls
anaconda-post.log  bin  dev  etc  home  lib  lib64  media  mnt  opt  proc  root  run  sbin  srv  sys  test_file.txt  tmp  usr  var
[root@47073d2898fc /]#
```

As you can see, I have a file, called "test_file.txt" in there. Let's now create an image "test3" from this container:



```
root@pavel-VirtualBox:/var/lib# docker commit test test3
sha256:b5b8845ab14fcdd09d8b0971959ec6cfe870c00d7ce23622d6fed9958111a72f
root@pavel-VirtualBox:/var/lib# docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
test3               latest              b5b8845ab14f        7 seconds ago       222MB
test2               latest              e3909ca9e2bf        7 minutes ago       222MB
centos              latest              75835a67d134        2 weeks ago         200MB
hello-world         latest              4ab4c602aa5e        7 weeks ago         1.84kB
```

Now, "test3" is among all the other images, we can now run it as a container and look at it's file system:



```
root@pavel-VirtualBox:/var/lib# docker run -i -t test3 bash
[root@3c722459de9d /]# ls
anaconda-post.log  bin  dev  etc  home  lib  lib64  media  mnt  opt  proc  root  run  sbin  srv  sys  test_file.txt  tmp  usr  var
[root@3c722459de9d /]#
```

We can see that "test_file.txt" is present in the copy of the image.

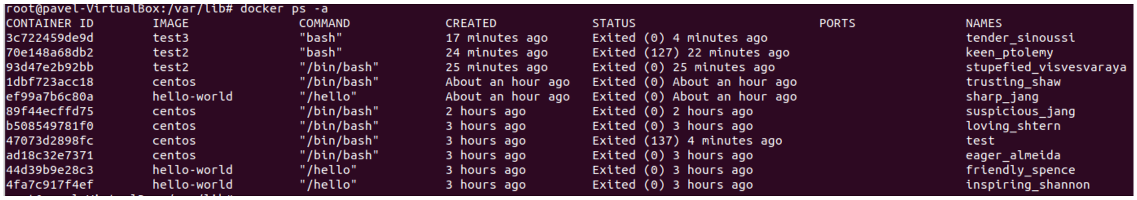
6) Stop docker container and clean up resources.

In order to stop a docker container you need to execute this command: docker stop container_name

To clean up, or remove all stopped containers, you can execute this command: docker container prune

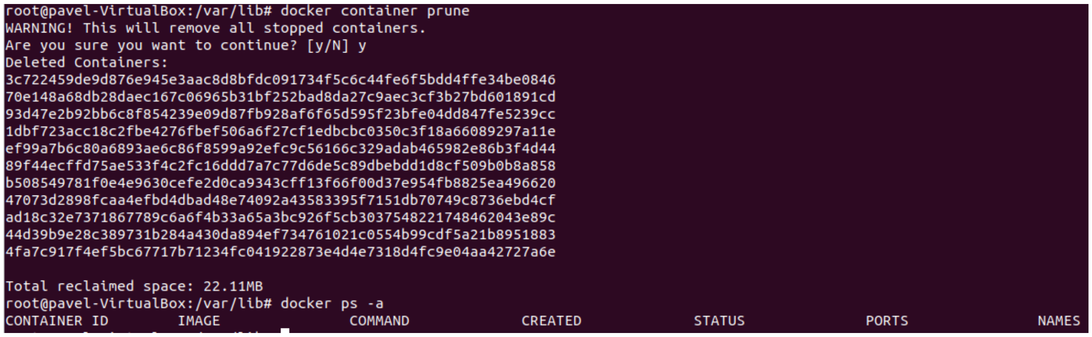
Also, as a means of cleanup, you can remove all dangling images (these are images that don’t have containers) by executing: docker image prune -a

Let’s list our containers before we remove them:



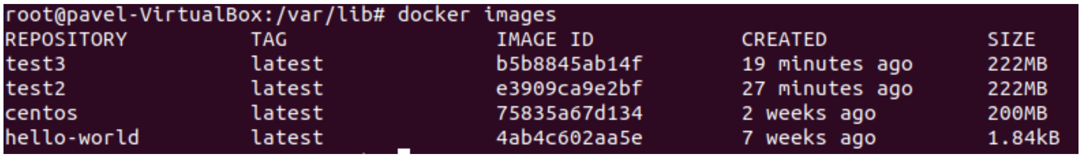
```
root@pavel-VirtualBox:/var/lib# docker ps -a
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS                        PORTS               NAMES
3c722459de9d        test3               "bash"              17 minutes ago      Exited (0) 4 minutes ago                          tender_sinoussi
70e148a68db2        test2               "bash"              24 minutes ago      Exited (127) 22 minutes ago                       keen_ptolemy
93d47e2b92bb        test2               "/bin/bash"         25 minutes ago      Exited (0) 25 minutes ago                         stupefied_visvesvaraya
1dbf723acc18        centos              "/bin/bash"         About an hour ago   Exited (0) About an hour ago                      trusting_shaw
ef99a7b6c80a        hello-world         "/hello"            About an hour ago   Exited (0) About an hour ago                      sharp_jang
89f44ecffd75        centos              "/bin/bash"         2 hours ago         Exited (0) 2 hours ago                            suspicious_jang
b508549781f0        centos              "/bin/bash"         3 hours ago         Exited (0) 3 hours ago                            loving_shtern
47073d2898fc        centos              "/bin/bash"         3 hours ago         Exited (137) 4 minutes ago                        test
ad18c32e7371        centos              "/bin/bash"         3 hours ago         Exited (0) 3 hours ago                            eager_almeida
44d39b9e28c3        hello-world         "/hello"            3 hours ago         Exited (0) 3 hours ago                            friendly_spence
4fa7c917f4ef        hello-world         "/hello"            3 hours ago         Exited (0) 3 hours ago                            inspiring_shannon
```

And after:



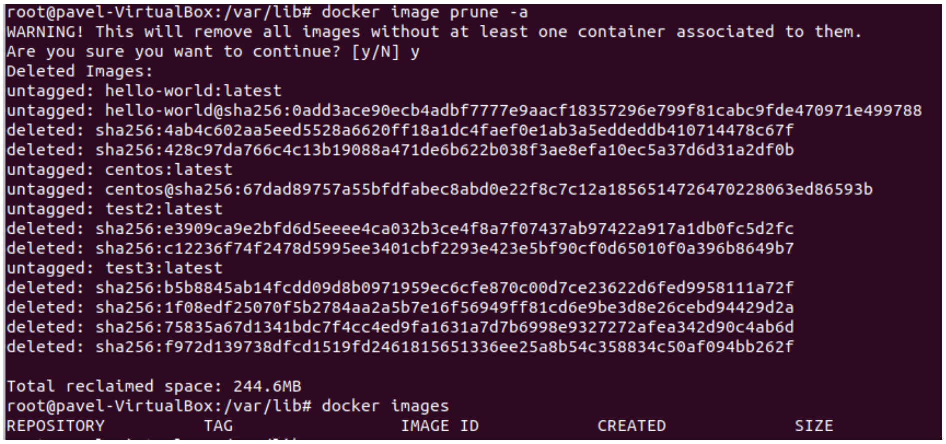
```
root@pavel-VirtualBox:/var/lib# docker container prune
WARNING! This will remove all stopped containers.
Are you sure you want to continue? [y/N] y
Deleted Containers:
3c722459de9d876e945e3aac8d8bfdc091734f5c6c44fe6f5bdd4ffe34be0846
70e148a68db28daec167c06965b31bf252bad8da27c9aec3cf3b27bd601891cd
93d47e2b92bb6c8f854239e09d87fb928af6f65d595f23bfe04dd847fe5239cc
1dbf723acc18c2fbe4276fbef506a6f27cf1edbcbc0350c3f18a66089297a11e
ef99a7b6c80a6893ae6c86f8599a92efc9c56166c329adab465982e86b3f4d44
89f44ecffd75ae533f4c2fc16ddd7a7c77d6de5c89dbebdd1d8cf509b0b8a858
b508549781f0e4e9630cefe2d0ca9343cff13f66f00d37e954fb8825ea496620
47073d2898fcaa4efbd4dbad48e74092a43583395f7151db70749c8736ebd4cf
ad18c32e7371867789c6a6f4b33a65a3bc926f5cb30375482217484620433e89c
44d39b9e28c389731b284a430da894ef734761021c0554b99cdf5a21b8951883
4fa7c917f4ef5bc67717b71234fc041922873e4d4e7318d4fc9e04aa42727a6e

Total reclaimed space: 22.11MB
root@pavel-VirtualBox:/var/lib# docker ps -a
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS               NAMES
```

Let’s list our images before we remove them:



```
root@pavel-VirtualBox:/var/lib# docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
test3               latest              b5b8845ab14f        19 minutes ago      222MB
test2               latest              e3909ca9e2bf        27 minutes ago      222MB
centos              latest              75835a67d134        2 weeks ago         200MB
hello-world         latest              4ab4c602aa5e        7 weeks ago         1.84kB
```

And after:



```
root@pavel-VirtualBox:/var/lib# docker image prune -a
WARNING! This will remove all images without at least one container associated to them.
Are you sure you want to continue? [y/N] y
Deleted Images:
untagged: hello-world:latest
untagged: hello-world@sha256:0add3ace90ecb4adbf7777e9aacf18357296e799f81cabc9fde470971e499788
deleted: sha256:4ab4c602aa5eed5528a6620ff18a1dc4faef0e1ab3a5eddeddb410714478c67f
deleted: sha256:428c97da766c4c13b19088a471de6b622b038f3ae8efa10ec5a37d6d31a2df0b
untagged: centos:latest
untagged: centos@sha256:67dad89757a55bfdfabec8abd0e22f8c7c12a1856514726470228063ed86593b
untagged: test2:latest
deleted: sha256:e3909ca9e2bfd6d5eeee4ca032b3ce4f8a7f07437ab97422a917a1db0fc5d2fc
deleted: sha256:c12236f74f2478d5995ee3401cbf2293e423e5bf90cf0d65010f0a396b8649b7
untagged: test3:latest
deleted: sha256:b5b8845ab14fcdd09d8b0971959ec6cfe870c00d7ce23622d6fed9958111a72f
deleted: sha256:1f08edf25070f5b2784aa2a5b7e16f56949ff81cd6e9be3d8e26cebd94429d2a
deleted: sha256:75835a67d1341bdc7f4cc4ed9fa1631a7d7b6998e9327272afea342d90c4ab6d
deleted: sha256:f972d139738dfcd1519fd2461815651336ee25a8b54c358834c50af094bb262f

Total reclaimed space: 244.6MB
root@pavel-VirtualBox:/var/lib# docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
```