

Задание 1.

Запускаем контейнер mongo-container на основе образа mongo

```
docker run -d -it -e MONGO_INITDB_ROOT_USERNAME=admin \
-e MONGO_INITDB_ROOT_PASSWORD=password \
--name mongo-container mongo
```

The screenshot shows a terminal window titled "Terminal" with the command "andrei@andrei-VirtualBox:~\$". The user runs "docker images" to see available images, then "docker run" to pull the mongo image and create a container named "mongo-container". Finally, "docker images" is run again to show the newly added mongo image.

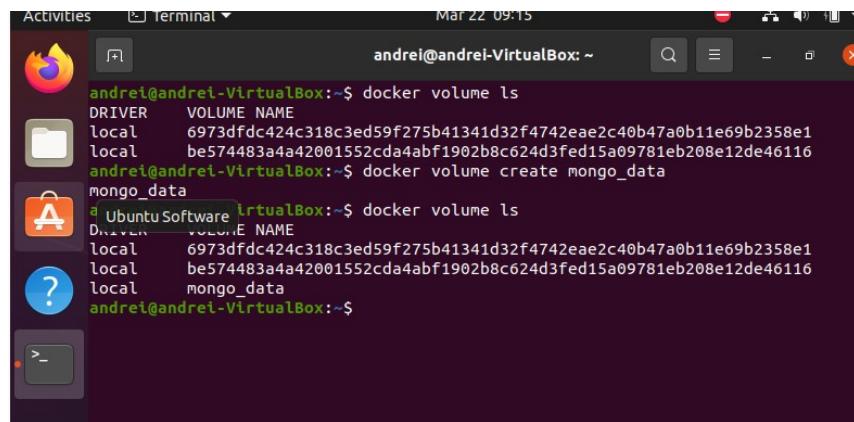
```
andrei@andrei-VirtualBox:~$ docker images
REPOSITORY TAG IMAGE ID CREATED SIZE
nginx latest 53a18edff809 6 weeks ago 192MB
ubuntu latest a04dc4851cbc 7 weeks ago 78.1MB
andrei@andrei-VirtualBox:~$ docker run -d -it -e MONGO_INITDB_ROOT_USERNAME=admin \
-e MONGO_INITDB_ROOT_PASSWORD=password --name mongo-container mongo
Unable to find image 'mongo:latest' locally
latest: Pulling from library/mongo
5a7813e071bf: Already exists
d67c4ebf9460: Pull complete
7afa02f8c09e: Pull complete
4e7ca17a42bd: Pull complete
342a4f4728ff: Pull complete
d5baf14fbe8: Pull complete
0c492c8e8cf0: Pull complete
734719e891c0: Pull complete
Digest: sha256:7bd28e5eea1c5766a084d5818254046f3ebe3b8f20a65e3a274640189e296667
Status: Downloaded newer image for mongo:latest
93b6990fd9456d9d60cdbe410a5c95b567538adde22ca9731859ae3f3515c536
andrei@andrei-VirtualBox:~$ docker images
REPOSITORY TAG IMAGE ID CREATED SIZE
mongo latest b81a621037ef 7 days ago 887MB
nginx latest 53a18edff809 6 weeks ago 192MB
ubuntu latest a04dc4851cbc 7 weeks ago 78.1MB
andrei@andrei-VirtualBox:~$
```

Проверяем, что контейнер mongo-container запущен
docker ps -a

The screenshot shows a terminal window titled "Terminal" with the command "andrei@andrei-VirtualBox:~\$". The user runs "docker ps -a" to list all Docker processes. It shows one container named "mongo-container" running on port 27017/tcp.

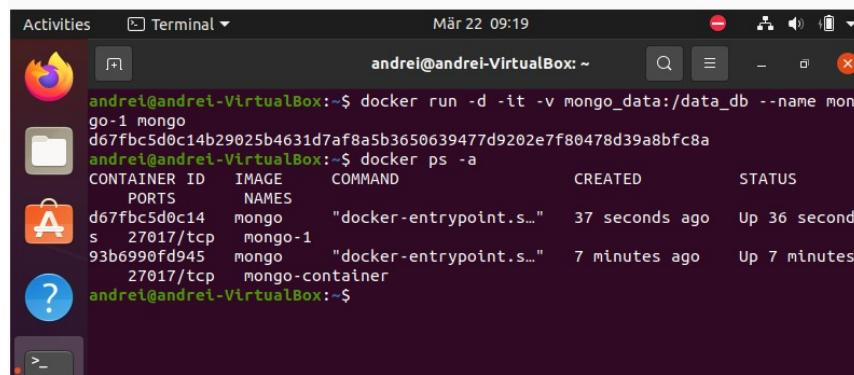
```
andrei@andrei-VirtualBox:~$ docker ps -a
CONTAINER ID IMAGE COMMAND CREATED STATUS
PORTS NAMES
93b6990fd945 mongo "docker-entrypoint.s..." About a minute ago Up About
a minute 27017/tcp mongo-container
andrei@andrei-VirtualBox:~$
```

Создаем том mongo_data и проверяем его
docker volume create mongo_data
docker volume ls



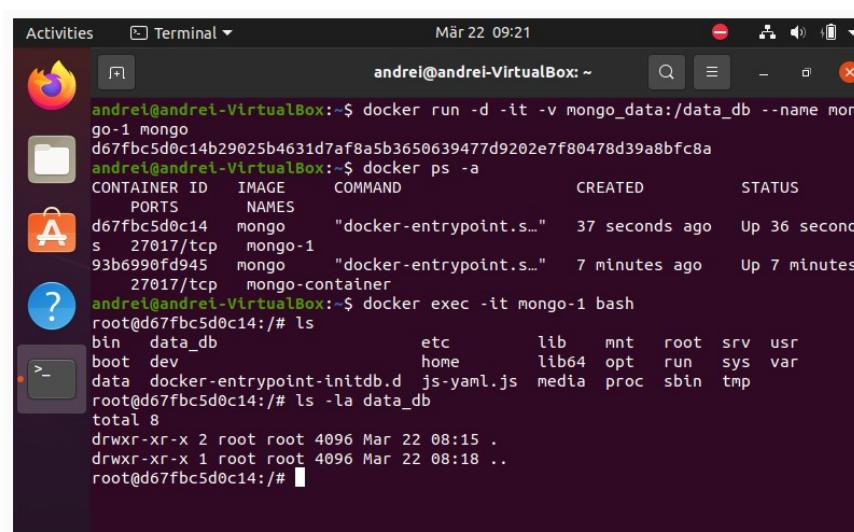
```
Activities Terminal Mar 22 09:15
andrei@andrei-VirtualBox:~$ docker volume ls
DRIVER      VOLUME NAME
local      6973dfdc424c318c3ed59f275b41341d32f4742eae2c40b47a0b11e69b2358e1
local      be574483a4a42001552cd4abf1902b8c624d3fed15a09781eb208e12de46116
andrei@andrei-VirtualBox:~$ docker volume create mongo_data
mongo_data
a UbuntuSoftwareVirtualBox:~$ docker volume ls
DRIVER      VOLUME NAME
local      6973dfdc424c318c3ed59f275b41341d32f4742eae2c40b47a0b11e69b2358e1
local      be574483a4a42001552cd4abf1902b8c624d3fed15a09781eb208e12de46116
local      mongo_data
andrei@andrei-VirtualBox:~$
```

Запускаем новый контейнер и подключаем к нему том mongo_data
docker run -d -it -v mongo_data:/data_db --name mongo-1 mongo



```
Activities Terminal Mär 22 09:19
andrei@andrei-VirtualBox:~$ docker run -d -it -v mongo_data:/data_db --name mongo-1 mongo
d67fb5d0c14b29025b4631d7af8a5b3650639477d9202e7f80478d39a8bfc8a
andrei@andrei-VirtualBox:~$ docker ps -a
CONTAINER ID IMAGE COMMAND CREATED STATUS
PORTS NAMES
d67fb5d0c14 mongo "docker-entrypoint.s..." 37 seconds ago Up 36 seconds
s 27017/tcp mongo-1
93b6990fd945 mongo "docker-entrypoint.s..." 7 minutes ago Up 7 minutes
27017/tcp mongo-container
andrei@andrei-VirtualBox:~$
```

Заходим в контейнер mongo-1 и проверяем подключение docker volume
docker exec -it mongo-1 bash



```
Activities Terminal Mär 22 09:21
andrei@andrei-VirtualBox:~$ docker run -d -it -v mongo_data:/data_db --name mongo-1 mongo
d67fb5d0c14b29025b4631d7af8a5b3650639477d9202e7f80478d39a8bfc8a
andrei@andrei-VirtualBox:~$ docker ps -a
CONTAINER ID IMAGE COMMAND CREATED STATUS
PORTS NAMES
d67fb5d0c14 mongo "docker-entrypoint.s..." 37 seconds ago Up 36 seconds
s 27017/tcp mongo-1
93b6990fd945 mongo "docker-entrypoint.s..." 7 minutes ago Up 7 minutes
27017/tcp mongo-container
andrei@andrei-VirtualBox:~$ docker exec -it mongo-1 bash
root@d67fb5d0c14:/# ls
bin  data_db          etc      lib     mnt   root  srv  usr
boot dev             home    lib64  opt   run   sys  var
data  docker-entrypoint-initdb.d js-yaml.js media  proc sbin tmp
root@d67fb5d0c14:/# ls -la data_db
total 8
drwxr-xr-x 2 root root 4096 Mar 22 08:15 .
drwxr-xr-x 1 root root 4096 Mar 22 08:18 ..
root@d67fb5d0c14:/#
```

Создаем новую сеть

```
docker network create -d bridge my_network
```

```
andrei@andrei-VirtualBox:~$ docker network create -d bridge my_network
17ffd0225c130e1ec7c9198d44ac37426b57dac4457c2f54e145a547b7a9f13d
andrei@andrei-VirtualBox:~$ docker network ls
NETWORK ID      NAME      DRIVER      SCOPE
4a88cb8721d5    bridge    bridge      local
2e7c4a7b1f2b    host      host       local
17ffd0225c13    my_network    bridge      local
07a5732f89f1    none      null       local
andrei@andrei-VirtualBox:~$
```

Создаем контейнер mongo-2 из образа mongo и подключим к нему my_network и том mongo_data

```
docker run -d -it -v mongo_data:/db_data --name mongo-2 \
--network=my_network mongo
```

```
andrei@andrei-VirtualBox:~$ docker run -d -it -v mongo_data:/db_data --name mongo-2 --network=my_network mongo
4507d6a85474bc123d94a17297b5667d4a1141fa240361ca06d593fe842911cb
andrei@andrei-VirtualBox:~$ docker ps -a
CONTAINER ID      IMAGE      COMMAND      CREATED      STATUS
PORTS      NAMES
4507d6a85474    mongo      "docker-entrypoint.s..."  5 seconds ago  Up 3 seconds
27017/tcp      mongo-2
d67fbc5d0c14    mongo      "docker-entrypoint.s..."  8 minutes ago  Up 8 minutes
27017/tcp      mongo-1
93b6990fd945    mongo      "docker-entrypoint.s..."  14 minutes ago  Up 14 minutes
27017/tcp      mongo-container
andrei@andrei-VirtualBox:~$
```

Проверяем контейнер mongo-2, его сеть my_network и docker volume mongo_data

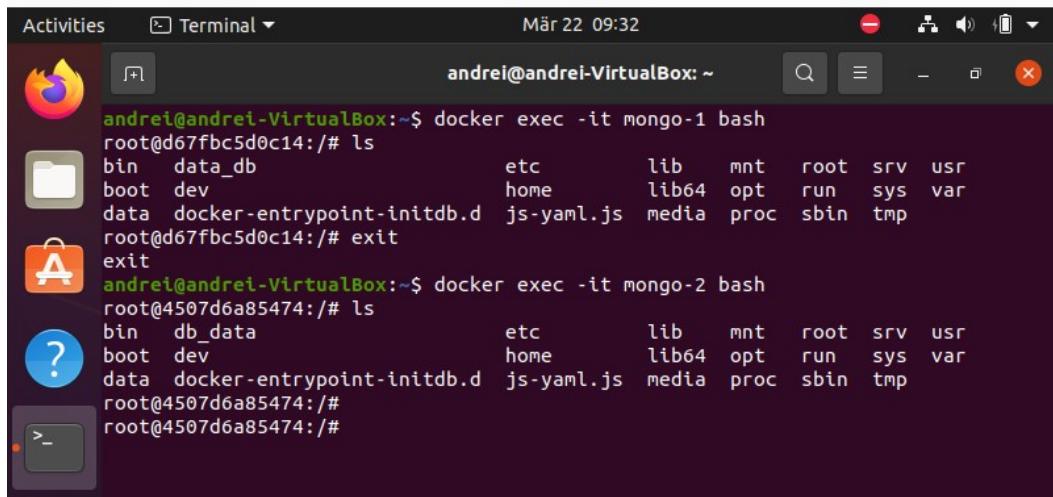
```
docker inspect mongo-2
```

```
"IPv6Gateway": "",  
"MacAddress": "",  
"Networks": {  
    "my_network": {  
        "IPAMConfig": null,  
        "Links": null,  
        "Aliases": null,  
        "MacAddress": "02:42:ac:18:00:02",  
        "NetworkID": "17ffd0225c130e1ec7c9198d44ac37426b57dac4457c2f54e145a547b7a9f13d",  
        "EndpointID": "8d8bd96a74ed4f4bcc72f3a6f608c5510e93fb8d5ad0e87aebc4c41ea2b8fd2",  
        "Gateway": "172.24.0.1",  
        "IPAddress": "172.24.0.2",  
        "IPPrefixLen": 16,  
        "IPv6Gateway": "",  
        "GlobalIPv6Address": "",  
        "GlobalIPv6PrefixLen": 0,  
        "DriverOpts": null,  
        "DNSNames": [  
            "mongo-2",  
            "4507d6a85474"  
        ]  
    }  
},  
}
```

```
29936cca4c47f7a2d09885fb8023ee54e734b599/merged",  
"UpperDir": "/var/lib/docker/overlay2/c42be512e4619c5c0d43b1302  
9936cca4c47f7a2d09885fb8023ee54e734b599/diff",  
"Workdir": "/var/lib/docker/overlay2/c42be512e4619c5c0d43b13029  
936cca4c47f7a2d09885fb8023ee54e734b599/work"  
},  
"Name": "overlay2"  
},  
"Mounts": [  
    {  
        "Type": "volume",  
        "Name": "728306b672b0abdc9e287a6ce8136f3d2f3a52a7b764f0466ca3a5  
7ec061aa43",  
        "Source": "/var/lib/docker/volumes/728306b672b0abdc9e287a6ce8136f3d2f3a52a7b764f0466ca3a5  
6f3d2f3a52a7b764f0466ca3a57ec061aa43/_data",  
        "Destination": "/data/configdb",  
        "Driver": "local",  
        "Mode": "",  
        "RW": true,  
        "Propagation": ""  
    },  

```

Запущенные контейнеры **mongo-1** и **mongo-2** смотрят на один том **mongo_data**.



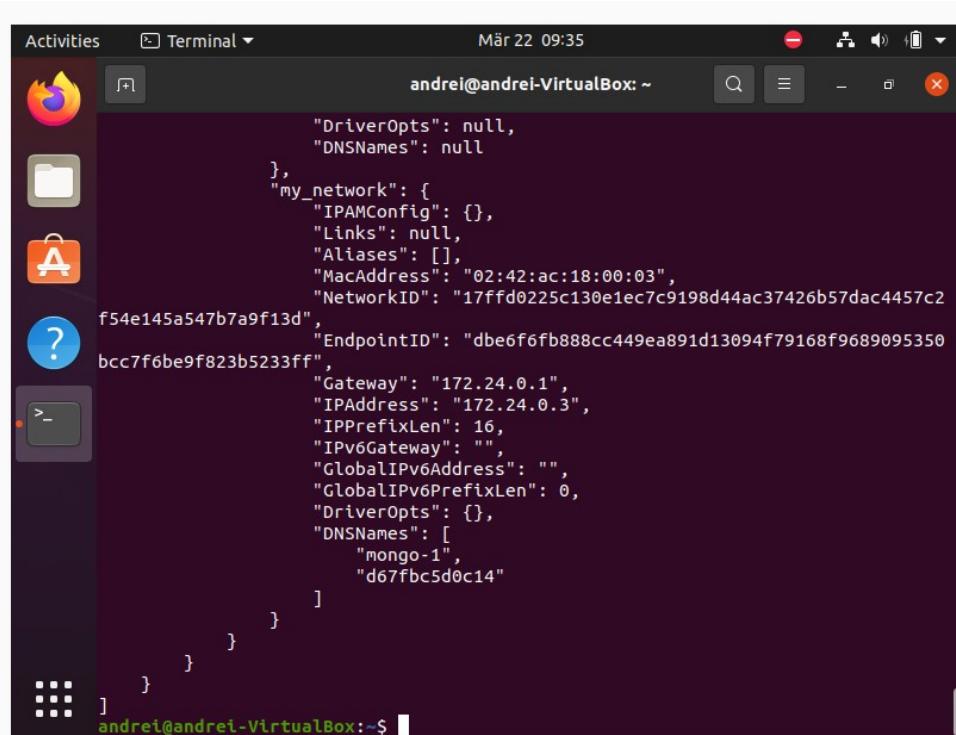
```
Activities Terminal Mär 22 09:32
andrei@andrei-VirtualBox:~$ docker exec -it mongo-1 bash
root@d67fbc5d0c14:/# ls
bin  data_db      etc    lib    mnt   root  srv  usr
boot dev          home   lib64  opt   run   sys  var
data docker-entrypoint-initdb.d js-yaml.js media proc sbin tmp
root@d67fbc5d0c14:/# exit
andrei@andrei-VirtualBox:~$ docker exec -it mongo-2 bash
root@4507d6a85474:/# ls
bin  db_data      etc    lib    mnt   root  srv  usr
boot dev          home   lib64  opt   run   sys  var
data docker-entrypoint-initdb.d js-yaml.js media proc sbin tmp
root@4507d6a85474:/#
root@4507d6a85474:/#
```

Подключим контейнер mongo-1 также к сети my_network

docker network connect my_network mongo-1

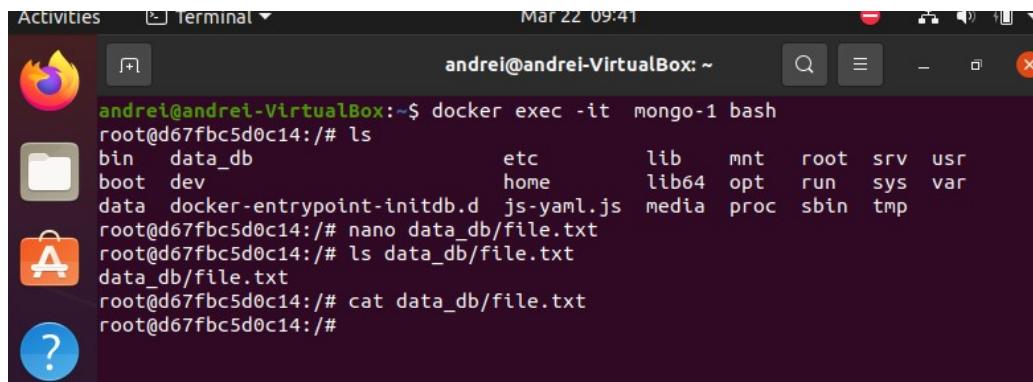
и выполняем проверку

docker inspect mongo-1



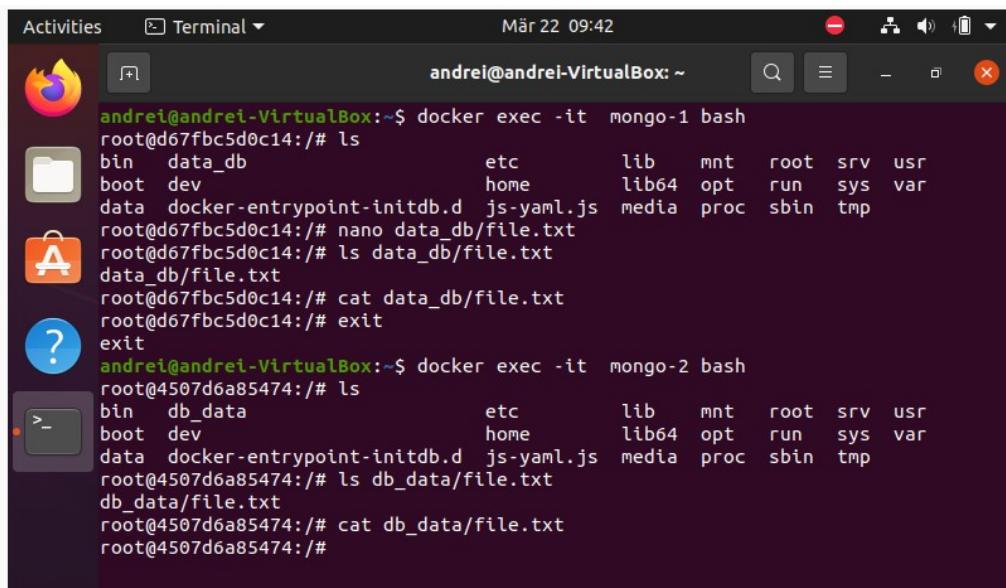
```
Activities Terminal Mär 22 09:35
andrei@andrei-VirtualBox:~$ docker inspect mongo-1
[{"Id": "f54e145a547b7a9f13d", "ContainerId": "d67fbc5d0c14", "HostConfig": {"Binds": ["mongo-1:/var/lib/mongo"], "CgroupParent": "", "Dns": ["172.24.0.3"], "DnsSearch": ["mongo-1"], "ExtraHosts": {}, "IpcMode": "private", "Links": null, "LogConfig": {"Config": {"Level": "info", "MaxLineLength": 1048576, "MaxLogFiles": 10, "MaxLogSize": 10485760}, "Type": "json-file"}, "Memory": 10485760, "MemoryReservation": 10485760, "Mounts": [{"ContainerPath": "/var/lib/mongo", "HostPath": "mongo-1:/var/lib/mongo", "Type": "bind"}], "NetworkMode": "my_network", "PortBindings": {}, "Privileged": false, "ReadonlyRootfs": false, "RestartPolicy": {"Attempts": 1, "Condition": "no-start"}, "ShmSize": 65536, "Ulimits": [{"Name": "nofile", "Soft": 1024, "Hard": 1024}], "User": "root", "VolumeDriver": ""}, "Image": "mongo:3.4", "Labels": {}, "Name": "mongo-1", "NetworkSettings": {"Bridge": "my_network", "GlobalIPv6Address": "", "GlobalIPv6PrefixLen": 0, "Gateway": "172.24.0.1", "IPAddress": "172.24.0.3", "IPPrefixLen": 16, "IPv6Gateway": "", "Links": null, "MacAddress": "02:42:ac:18:00:03", "NetworkID": "17ffd0225c130e1ec7c9198d44ac37426b57dac4457c2bcc7f6be9f823b5233ff", "Ports": {}, "PortsConfig": null, "SecondaryIPAddresses": null, "SecondaryPortRanges": null, "ServerName": null, "Stack": null, "Subnets": null}}, "Path": "mongo-1", "Type": "container"}]
```

Создаем в контейнере mongo-1, в подключенном volume, file.txt
docker exec -it mongo-1 bash
nano data_db/file.txt



```
Activities Terminal Mar 22 09:41
andrei@andrei-VirtualBox:~$ docker exec -it mongo-1 bash
root@d67fbc5d0c14:/# ls
bin  data_db          etc      lib     mnt   root  srv  usr
boot dev              home    lib64  opt   run   sys  var
data  docker-entrypoint-initdb.d js-yaml.js media  proc  sbin  tmp
root@d67fbc5d0c14:/# nano data_db/file.txt
root@d67fbc5d0c14:/# ls data_db/file.txt
data_db/file.txt
root@d67fbc5d0c14:/# cat data_db/file.txt
root@d67fbc5d0c14:/#
```

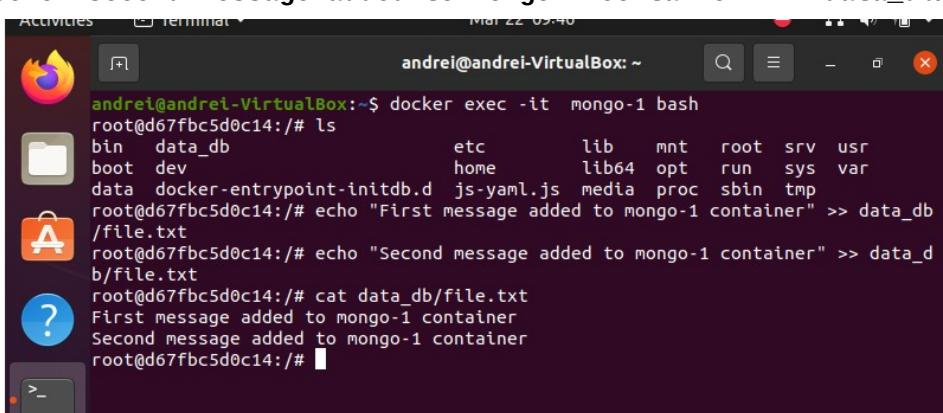
Файл file.txt появляется и в контейнере mongo-2



```
Activities Terminal Mar 22 09:42
andrei@andrei-VirtualBox:~$ docker exec -it mongo-1 bash
root@d67fbc5d0c14:/# ls
bin  data_db          etc      lib     mnt   root  srv  usr
boot dev              home    lib64  opt   run   sys  var
data  docker-entrypoint-initdb.d js-yaml.js media  proc  sbin  tmp
root@d67fbc5d0c14:/# nano data_db/file.txt
root@d67fbc5d0c14:/# ls data_db/file.txt
data_db/file.txt
root@d67fbc5d0c14:/# cat data_db/file.txt
root@d67fbc5d0c14:/# exit
andrei@andrei-VirtualBox:~$ docker exec -it mongo-2 bash
root@4507d6a85474:/# ls
bin  db_data          etc      lib     mnt   root  srv  usr
boot dev              home    lib64  opt   run   sys  var
data  docker-entrypoint-initdb.d js-yaml.js media  proc  sbin  tmp
root@4507d6a85474:/# ls db_data/file.txt
db_data/file.txt
root@4507d6a85474:/# cat db_data/file.txt
root@4507d6a85474:/#
```

Записываем несколько сообщений в созданный файл file.txt в mongo-1

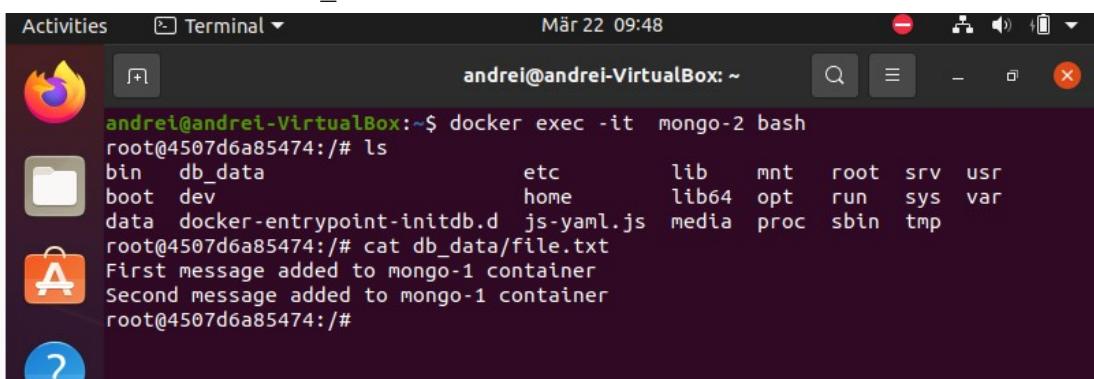
```
docker exec -it mongo-1 bash
echo "First message added to mongo-1 container" >> data_db/file.txt
echo "Second message added to mongo-1 container" >> data_db/file.txt
```



```
Activities Terminal Mar 22 09:40
andrei@andrei-VirtualBox:~$ docker exec -it mongo-1 bash
root@d67fbc5d0c14:/# ls
bin  data_db          etc      lib     mnt   root  srv  usr
boot dev              home    lib64  opt   run   sys  var
data  docker-entrypoint-initdb.d js-yaml.js media  proc  sbin  tmp
root@d67fbc5d0c14:/# echo "First message added to mongo-1 container" >> data_db/file.txt
root@d67fbc5d0c14:/# echo "Second message added to mongo-1 container" >> data_db/file.txt
root@d67fbc5d0c14:/# cat data_db/file.txt
First message added to mongo-1 container
Second message added to mongo-1 container
root@d67fbc5d0c14:/#
```

Проверяем содержание файла file.txt в контейнере mongo-2

```
docker exec -it mongo_vol_container2 bash  
cat db_data/file.txt
```

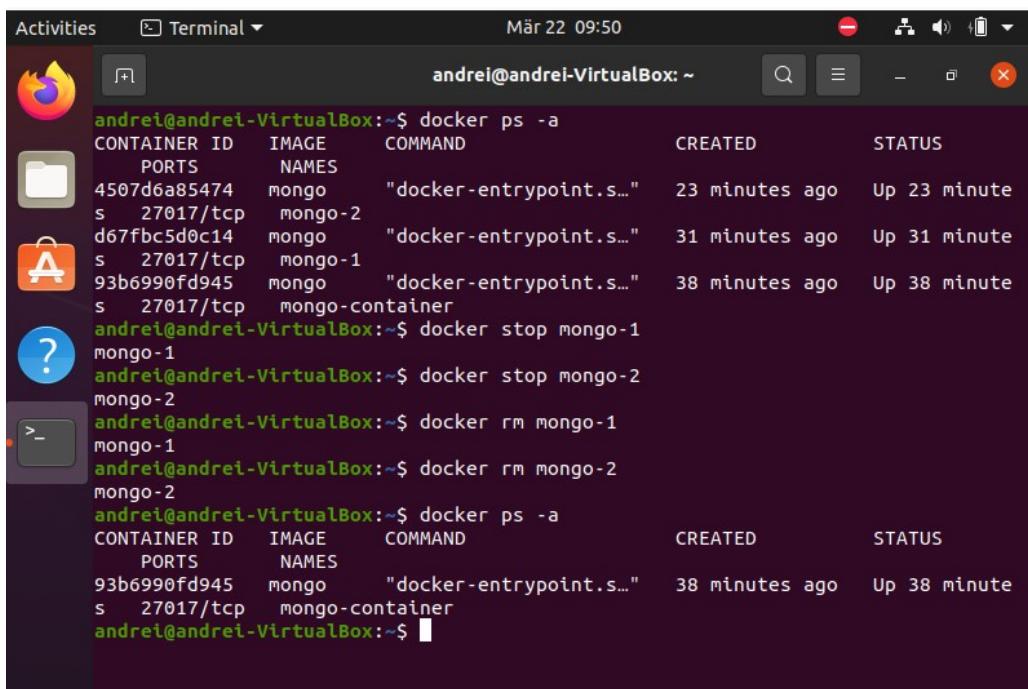


```
andrei@andrei-VirtualBox:~$ docker exec -it mongo-2 bash  
root@4507d6a85474:/# ls  
bin db_data etc lib mnt root srv usr  
boot dev home lib64 opt run sys var  
data docker-entrypoint-initdb.d js-yaml.js media proc sbin tmp  
root@4507d6a85474:/# cat db_data/file.txt  
First message added to mongo-1 container  
Second message added to mongo-1 container  
root@4507d6a85474:/#
```

Останавливаем контейнеры и затем удаляем их

```
docker stop mongo-1  
docker stop mongo-2
```

```
docker rm mongo-1  
docker rm mongo-2
```



```
andrei@andrei-VirtualBox:~$ docker ps -a  
CONTAINER ID IMAGE COMMAND CREATED STATUS  
PORTS NAMES  
4507d6a85474 mongo "docker-entrypoint.s..." 23 minutes ago Up 23 minute  
s 27017/tcp mongo-2  
d67fbc5d0c14 mongo "docker-entrypoint.s..." 31 minutes ago Up 31 minute  
s 27017/tcp mongo-1  
93b6990fd945 mongo "docker-entrypoint.s..." 38 minutes ago Up 38 minute  
s 27017/tcp mongo-container  
andrei@andrei-VirtualBox:~$ docker stop mongo-1  
mongo-1  
andrei@andrei-VirtualBox:~$ docker stop mongo-2  
mongo-2  
andrei@andrei-VirtualBox:~$ docker rm mongo-1  
mongo-1  
andrei@andrei-VirtualBox:~$ docker rm mongo-2  
mongo-2  
andrei@andrei-VirtualBox:~$ docker ps -a  
CONTAINER ID IMAGE COMMAND CREATED STATUS  
PORTS NAMES  
93b6990fd945 mongo "docker-entrypoint.s..." 38 minutes ago Up 38 minute  
s 27017/tcp mongo-container  
andrei@andrei-VirtualBox:~$
```

Удаляем образ mongo

```
docker rmi mongo
```

The terminal window shows the command `docker rmi mongo` being run, which successfully removes the mongo image. The output also lists other untagged images.

```
Activities Terminal M&P andrei@andrei-VirtualBox: ~
REPOSITORY TAG IMAGE ID CREATED SIZE
mongo latest b81a621937ef 7 days ago 887MB
nginx latest 53a18edff809 6 weeks ago 192MB
ubuntu latest a04dc4851cbc 7 weeks ago 78.1MB
andrei@andrei-VirtualBox: $ docker rmi mongo
Untagged: mongo:latest
Deleted: mongo@sha256:7bd28e5eea1c5766a084d5818254046f3ebe3b8f20a65e3a27464018
9e296667
Deleted: sha256:b81a621037ef33c357805810a4995dd1bf81b8bc81f7640857d10f1dd7434f26
Deleted: sha256:316fc4387bc37ed0bf2b90e6ab828a86c652073dd6a5dda921159d1e1268ce77
Deleted: sha256:fb015276a81803226e30ae392649e98a6d24a6a3534bae129f21b3ac58b46218
Deleted: sha256:ad2510f1d0adddfa9d68c0b8b13408b4c90ba4585b0365ddf359b6bdc66bd
Deleted: sha256:062b3e903f0cedf78e555e583842aab319f06a54c20ddd640c3a878b1a79147b
Deleted: sha256:5cc6a70be5b22d92d86c4b206674652bbe3d36629d2455fde160540459353077
Deleted: sha256:2ab8b4d001269c4249e70c017607af4f8f42b828cb44b2977a0026fc4b43d9e9
Deleted: sha256:d473b26149c562a3cef58aad0e2e7b18b556ab3fe542d3de582611542bb67f
andrei@andrei-VirtualBox: $ docker images
REPOSITORY TAG IMAGE ID CREATED SIZE
nginx latest 53a18edff809 6 weeks ago 192MB
ubuntu latest a04dc4851cbc 7 weeks ago 78.1MB
andrei@andrei-VirtualBox: $
```

Удаляем том mongo_data

```
docker volume rm mongo_data
```

The terminal window shows the command `docker volume rm mongo_data` being run, which successfully removes the mongo_data volume. The output also lists other volumes.

```
andrei@andrei-VirtualBox: ~$ docker volume ls
DRIVER VOLUME NAME
local 0fd82f10019e891661234225f6984c148659f0d4c82922a8fb5f347bf0f3c874
local 6973dfdc424c318c3ed59f275b41341d32f4742eae2c40b47a0b11e69b2358e1
local 23798d80b3d65e177dc5a50b3a99e8555852a6957bf61723f98ff4e4224a1b41
local 728306b672b0abdc9e287a6ce8136f3d2f3a52a7b764f0466ca3a57ec061aa43
local 5660456afc2573bfa6d8af515b3ac91ba736365ffc33cdf2098b94ffa9869539
local be574483a4a42001552cda4abf1902b8c624d3fed15a09781eb208e12de46116
local mongo_data
andrei@andrei-VirtualBox: ~$ docker volume rm mongo_data
mongo_data
andrei@andrei-VirtualBox: ~$ docker volume ls
DRIVER VOLUME NAME
local 0fd82f10019e891661234225f6984c148659f0d4c82922a8fb5f347bf0f3c874
local 6973dfdc424c318c3ed59f275b41341d32f4742eae2c40b47a0b11e69b2358e1
local 23798d80b3d65e177dc5a50b3a99e8555852a6957bf61723f98ff4e4224a1b41
local 728306b672b0abdc9e287a6ce8136f3d2f3a52a7b764f0466ca3a57ec061aa43
local 5660456afc2573bfa6d8af515b3ac91ba736365ffc33cdf2098b94ffa9869539
local be574483a4a42001552cda4abf1902b8c624d3fed15a09781eb208e12de46116
andrei@andrei-VirtualBox: ~$
```

Удаляем network

```
docker network rm my_network
```

The terminal window shows the command `docker network rm my_network` being run, which fails because the network was not found. The output also lists other networks.

```
andrei@andrei-VirtualBox: ~$ docker network ls
NETWORK ID NAME DRIVER SCOPE
4a88cb8721d5 bridge bridge local
2e7c4a7b1f2b host host local
17ffd0225c13 my_network bridge local
07a5732f89f1 none null local
andrei@andrei-VirtualBox: ~$ docker network rm my_network
my_network
andrei@andrei-VirtualBox: ~$ Error response from daemon: network my_network not found
andrei@andrei-VirtualBox: ~$ docker volume ls
DRIVER VOLUME NAME
local 0fd82f10019e891661234225f6984c148659f0d4c82922a8fb5f347bf0f3c874
local 6973dfdc424c318c3ed59f275b41341d32f4742eae2c40b47a0b11e69b2358e1
local 23798d80b3d65e177dc5a50b3a99e8555852a6957bf61723f98ff4e4224a1b41
local 728306b672b0abdc9e287a6ce8136f3d2f3a52a7b764f0466ca3a57ec061aa43
local 5660456afc2573bfa6d8af515b3ac91ba736365ffc33cdf2098b94ffa9869539
local be574483a4a42001552cda4abf1902b8c624d3fed15a09781eb208e12de46116
andrei@andrei-VirtualBox: ~$ docker network ls
NETWORK ID NAME DRIVER SCOPE
4a88cb8721d5 bridge bridge local
2e7c4a7b1f2b host host local
07a5732f89f1 none null local
andrei@andrei-VirtualBox: ~$
```