Бекарев С.С. Лабораторная работа №2

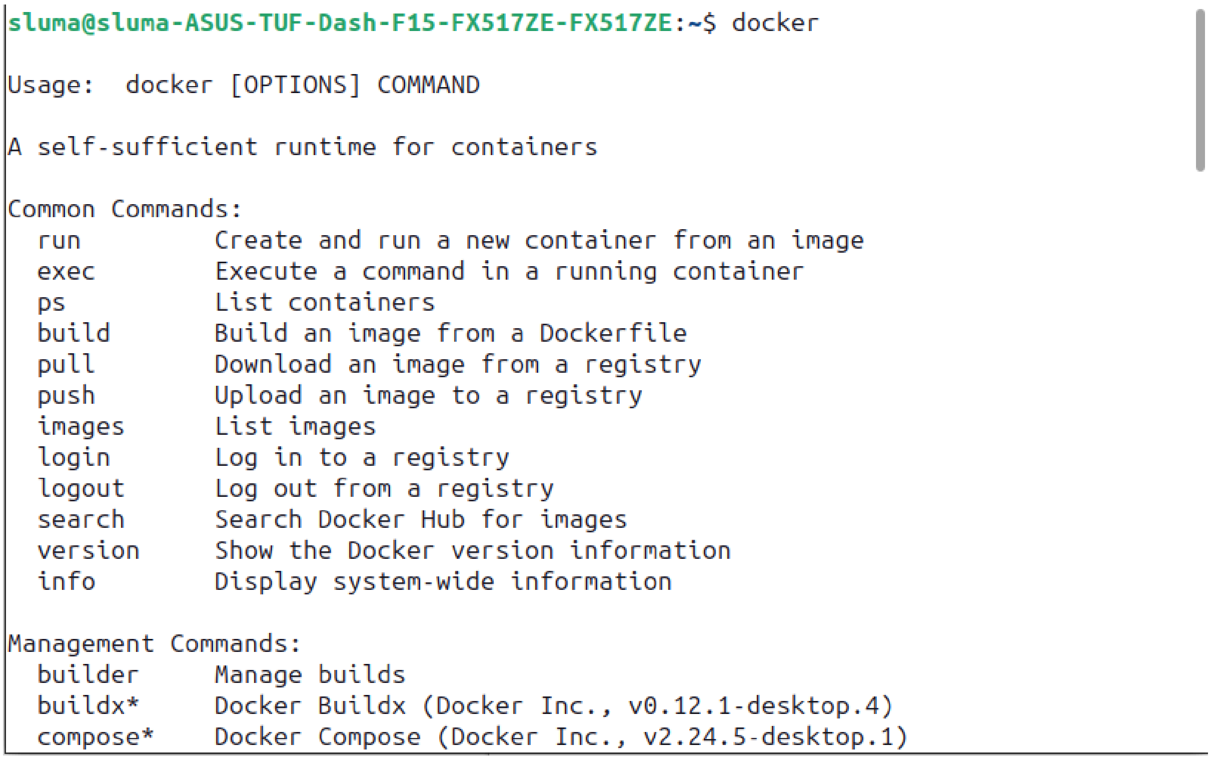
Тема: Работа с Docker.

Цель: Познакомиться с возможностями и получить практические навыки работы с Docker.

Задание:

1. Подготовьте рабочее окружение в соответствии с типом вашей операционной системы

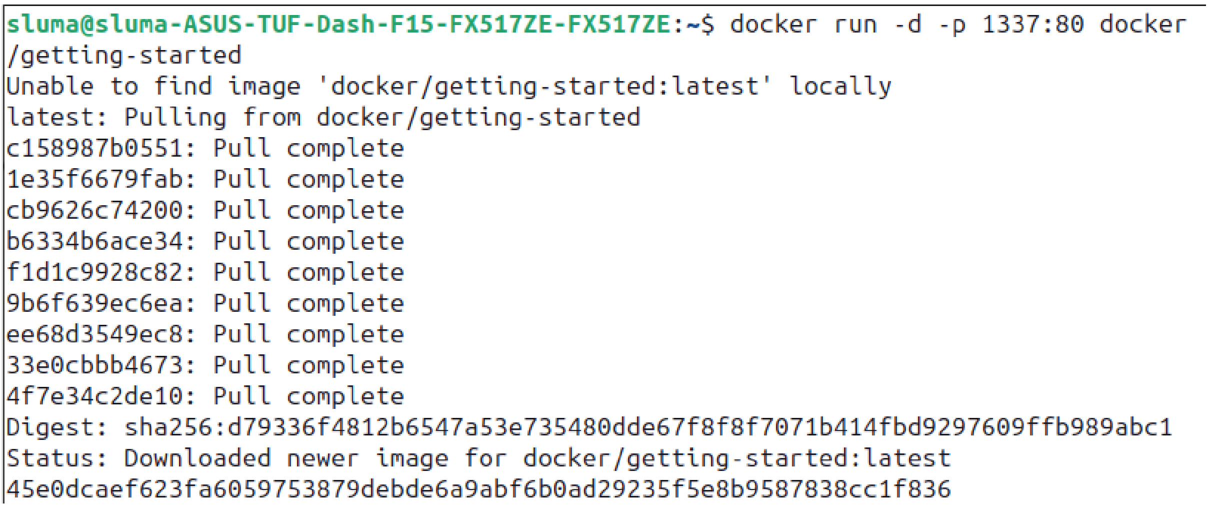
    ○ Установите Docker



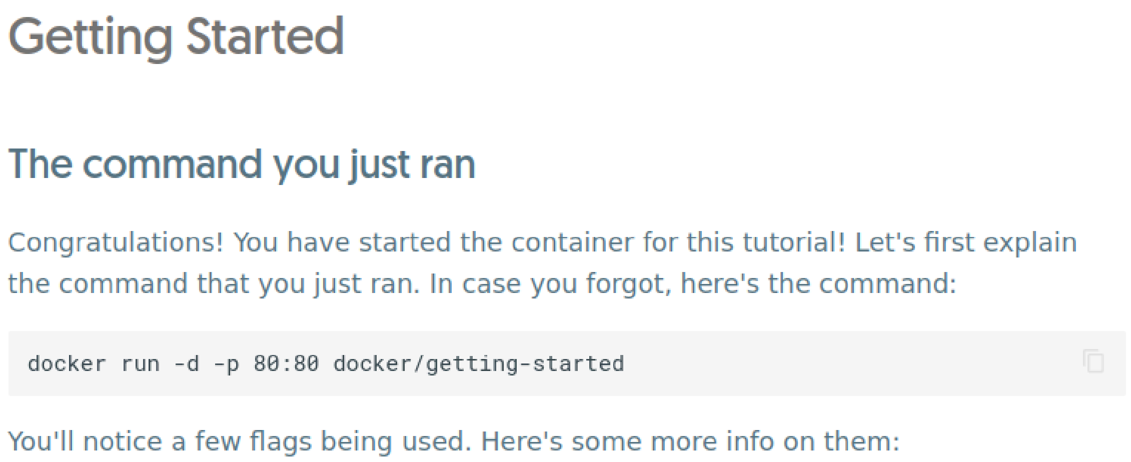
```
sluma@sluma-ASUS-TUF-Dash-F15-FX517ZE-FX517ZE:~$ docker

Usage:  docker [OPTIONS] COMMAND

A self-sufficient runtime for containers

Common Commands:
  run         Create and run a new container from an image
  exec        Execute a command in a running container
  ps          List containers
  build       Build an image from a Dockerfile
  pull        Download an image from a registry
  push        Upload an image to a registry
  images      List images
  login       Log in to a registry
  logout      Log out from a registry
  search      Search Docker Hub for images
  version     Show the Docker version information
  info        Display system-wide information

Management Commands:
  builder     Manage builds
  buildx*     Docker Buildx (Docker Inc., v0.12.1-desktop.4)
  compose*    Docker Compose (Docker Inc., v2.24.5-desktop.1)
```

    ○ Выполните базовую настройку

2. Изучите простейшие консольные команды и возможности Docker Desktop (см. лекцию), создать собственный контейнер docker/getting-started, открыть в браузере и изучить tutorial.



```
sluma@sluma-ASUS-TUF-Dash-F15-FX517ZE-FX517ZE:~$ docker run -d -p 1337:80 docker
/getting-started
Unable to find image 'docker/getting-started:latest' locally
latest: Pulling from docker/getting-started
c158987b0551: Pull complete
1e35f6679fab: Pull complete
cb9626c74200: Pull complete
b6334b6ace34: Pull complete
f1d1c9928c82: Pull complete
9b6f639ec6ea: Pull complete
ee68d3549ec8: Pull complete
33e0cbbb4673: Pull complete
4f7e34c2de10: Pull complete
Digest: sha256:d79336f4812b6547a53e735480dde67f8f8f7071b414fbd9297609ffb989abc1
Status: Downloaded newer image for docker/getting-started:latest
45e0dcaef623fa6059753879debde6a9abf6b0ad29235f5e8b9587838cc1f836
```



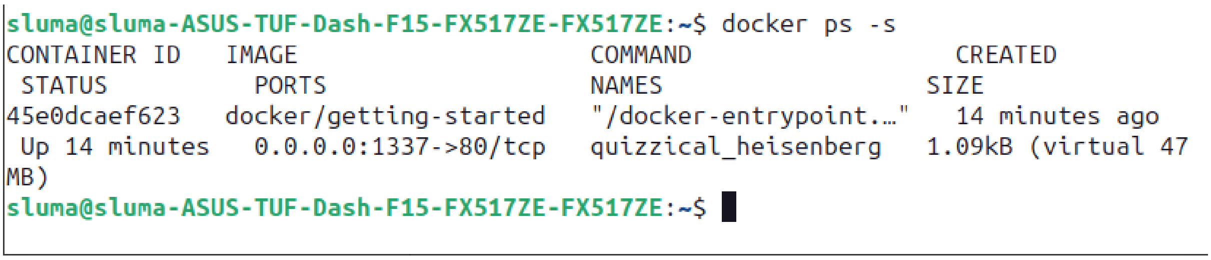
Getting Started

The command you just ran

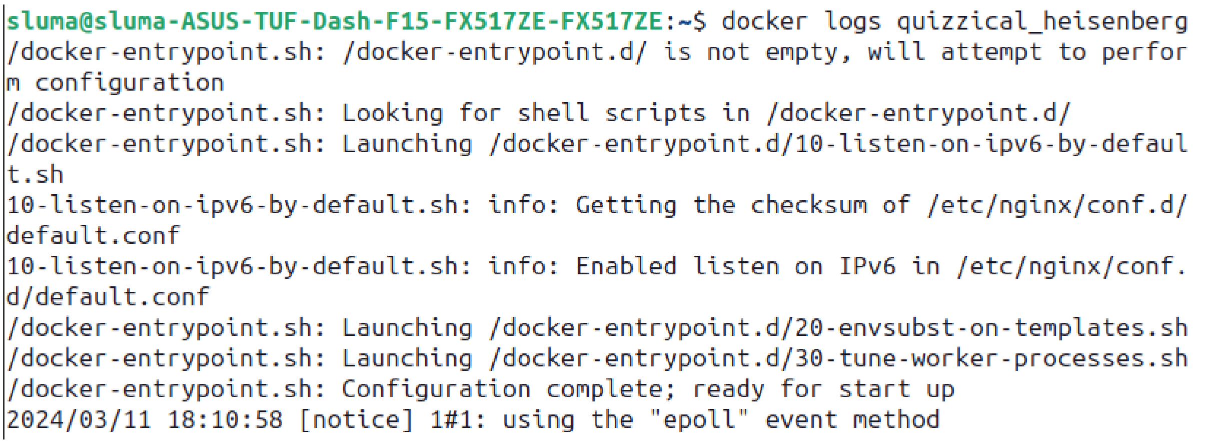
Congratulations! You have started the container for this tutorial! Let's first explain the command that you just ran. In case you forgot, here's the command:

```
docker run -d -p 80:80 docker/getting-started
```

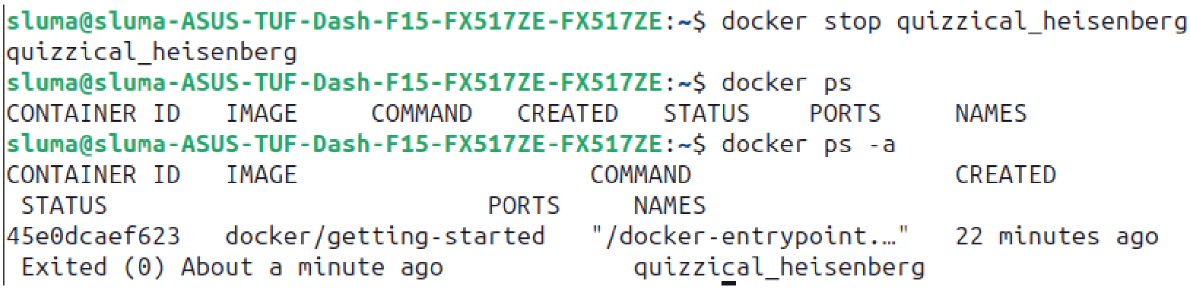
You'll notice a few flags being used. Here's some more info on them:



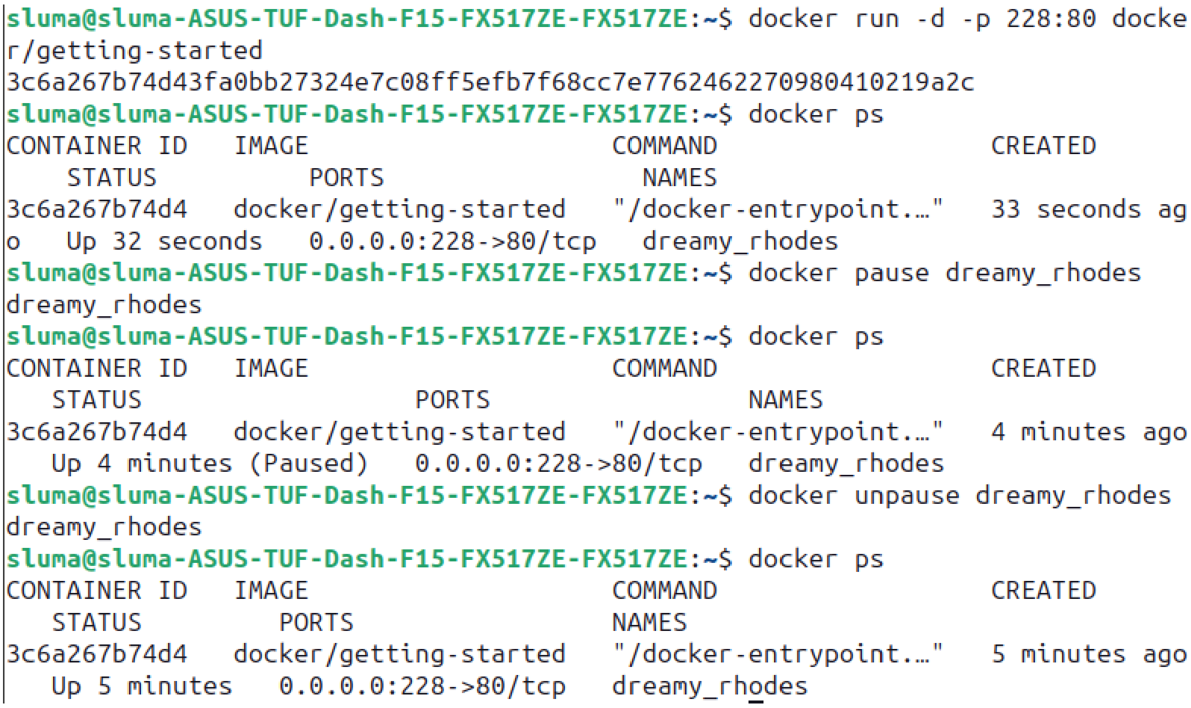
```
sluma@sluma-ASUS-TUF-Dash-F15-FX517ZE-FX517ZE:~$ docker ps -s
CONTAINER ID   IMAGE                     COMMAND                  CREATED
 STATUS          PORTS                   NAMES                SIZE
45e0dcaef623   docker/getting-started   "/docker-entrypoint.…"   14 minutes ago
 Up 14 minutes   0.0.0.0:1337->80/tcp   quizzical_heisenberg   1.09kB (virtual 47
MB)
sluma@sluma-ASUS-TUF-Dash-F15-FX517ZE-FX517ZE:~$ 
```



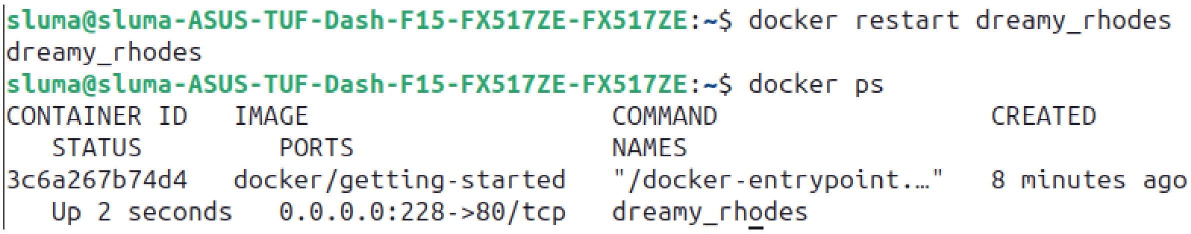
```
sluma@sluma-ASUS-TUF-Dash-F15-FX517ZE-FX517ZE:~$ docker logs quizzical_heisenberg
/docker-entrypoint.sh: /docker-entrypoint.d/ is not empty, will attempt to perfor
m configuration
/docker-entrypoint.sh: Looking for shell scripts in /docker-entrypoint.d/
/docker-entrypoint.sh: Launching /docker-entrypoint.d/10-listen-on-ipv6-by-defaul
t.sh
10-listen-on-ipv6-by-default.sh: info: Getting the checksum of /etc/nginx/conf.d/
default.conf
10-listen-on-ipv6-by-default.sh: info: Enabled listen on IPv6 in /etc/nginx/conf.
d/default.conf
/docker-entrypoint.sh: Launching /docker-entrypoint.d/20-envsubst-on-templates.sh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/30-tune-worker-processes.sh
/docker-entrypoint.sh: Configuration complete; ready for start up
2024/03/11 18:10:58 [notice] 1#1: using the "epoll" event method
```



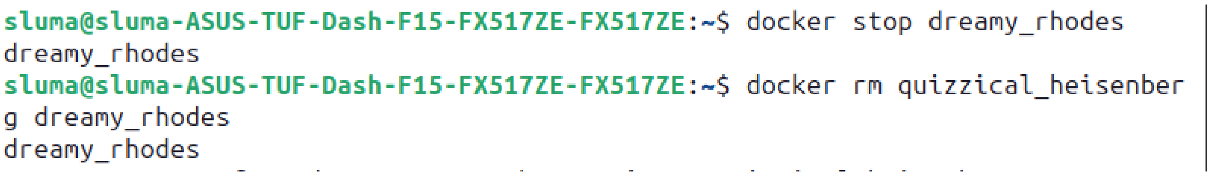
```
sluma@sluma-ASUS-TUF-Dash-F15-FX517ZE-FX517ZE:~$ docker stop quizzical_heisenberg
quizzical_heisenberg
sluma@sluma-ASUS-TUF-Dash-F15-FX517ZE-FX517ZE:~$ docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
sluma@sluma-ASUS-TUF-Dash-F15-FX517ZE-FX517ZE:~$ docker ps -a
CONTAINER ID   IMAGE                    COMMAND                  CREATED
 STATUS                          PORTS     NAMES
45e0dcaef623   docker/getting-started   "/docker-entrypoint.…"   22 minutes ago
 Exited (0) About a minute ago             quizzical_heisenberg
```



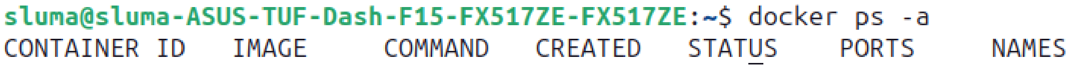
```
sluma@sluma-ASUS-TUF-Dash-F15-FX517ZE-FX517ZE:~$ docker run -d -p 228:80 docke
r/getting-started
3c6a267b74d43fa0bb27324e7c08ff5efb7f68cc7e7762462270980410219a2c
sluma@sluma-ASUS-TUF-Dash-F15-FX517ZE-FX517ZE:~$ docker ps
CONTAINER ID   IMAGE                    COMMAND                  CREATED
    STATUS         PORTS                  NAMES
3c6a267b74d4   docker/getting-started   "/docker-entrypoint.…"   33 seconds ag
o    Up 32 seconds   0.0.0.0:228->80/tcp   dreamy_rhodes
sluma@sluma-ASUS-TUF-Dash-F15-FX517ZE-FX517ZE:~$ docker pause dreamy_rhodes
dreamy_rhodes
sluma@sluma-ASUS-TUF-Dash-F15-FX517ZE-FX517ZE:~$ docker ps
CONTAINER ID   IMAGE                    COMMAND                  CREATED
   STATUS                  PORTS                  NAMES
3c6a267b74d4   docker/getting-started   "/docker-entrypoint.…"   4 minutes ago
   Up 4 minutes (Paused)   0.0.0.0:228->80/tcp   dreamy_rhodes
sluma@sluma-ASUS-TUF-Dash-F15-FX517ZE-FX517ZE:~$ docker unpause dreamy_rhodes
dreamy_rhodes
sluma@sluma-ASUS-TUF-Dash-F15-FX517ZE-FX517ZE:~$ docker ps
CONTAINER ID   IMAGE                    COMMAND                  CREATED
   STATUS         PORTS                  NAMES
3c6a267b74d4   docker/getting-started   "/docker-entrypoint.…"   5 minutes ago
   Up 5 minutes   0.0.0.0:228->80/tcp   dreamy_rhodes
```



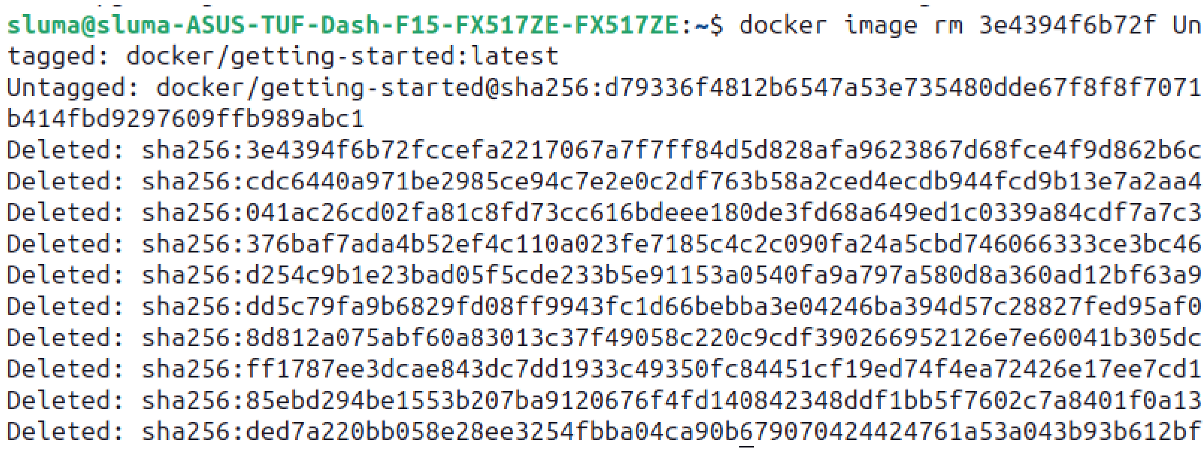
```
sluma@sluma-ASUS-TUF-Dash-F15-FX517ZE-FX517ZE:~$ docker restart dreamy_rhodes
dreamy_rhodes
sluma@sluma-ASUS-TUF-Dash-F15-FX517ZE-FX517ZE:~$ docker ps
CONTAINER ID   IMAGE                    COMMAND                  CREATED
   STATUS         PORTS                  NAMES
3c6a267b74d4   docker/getting-started   "/docker-entrypoint.…"   8 minutes ago
   Up 2 seconds   0.0.0.0:228->80/tcp   dreamy_rhodes
```



```
sluma@sluma-ASUS-TUF-Dash-F15-FX517ZE-FX517ZE:~$ docker stop dreamy_rhodes
dreamy_rhodes
sluma@sluma-ASUS-TUF-Dash-F15-FX517ZE-FX517ZE:~$ docker rm quizzical_heisenber
g dreamy_rhodes
dreamy_rhodes
```



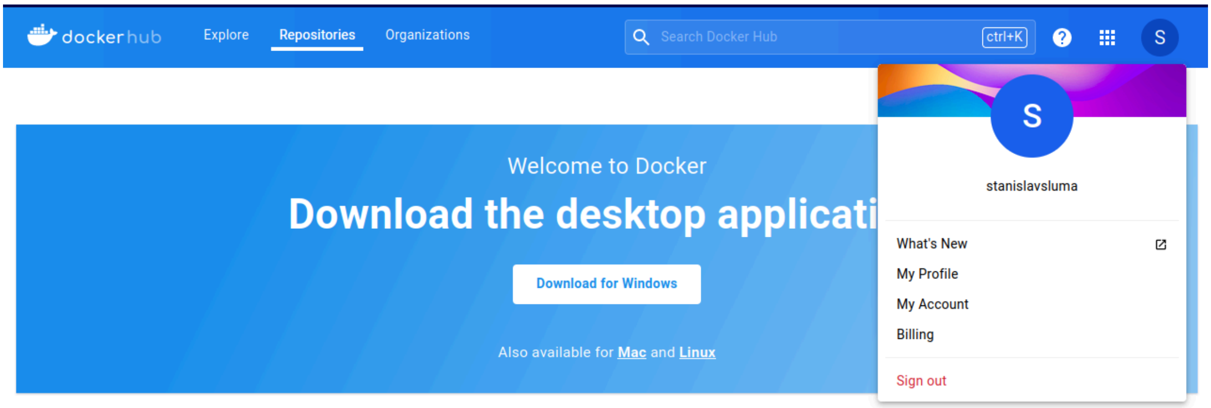
```
sluma@sluma-ASUS-TUF-Dash-F15-FX517ZE-FX517ZE:~$ docker ps -a
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
```



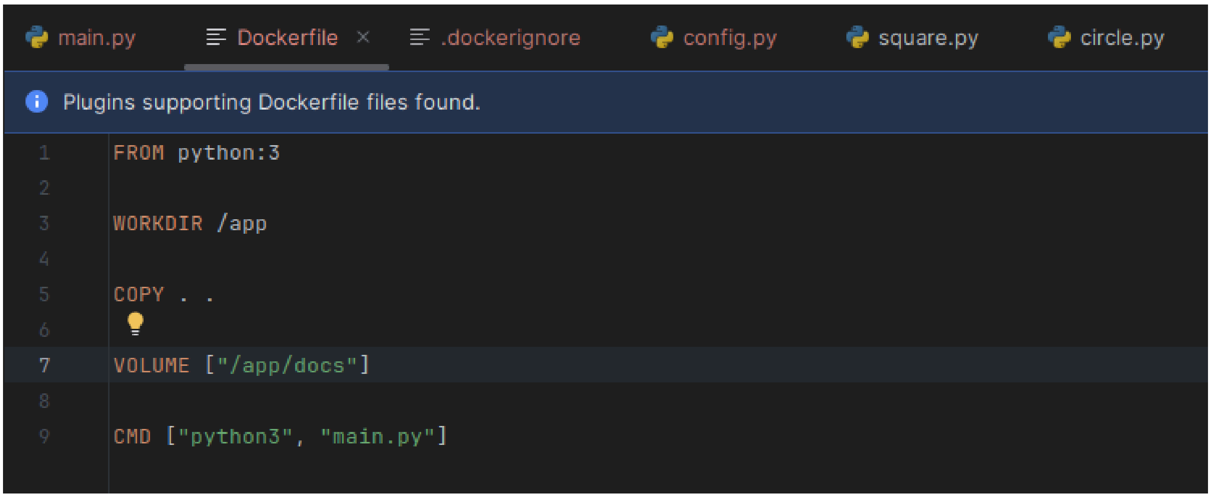
```
sluma@sluma-ASUS-TUF-Dash-F15-FX517ZE-FX517ZE:~$ docker image rm 3e4394f6b72f Un
tagged: docker/getting-started:latest
Untagged: docker/getting-started@sha256:d79336f4812b6547a53e735480dde67f8f8f7071
b414fbd9297609ffb989abc1
Deleted: sha256:3e4394f6b72fccefa2217067a7f7ff84d5d828afa9623867d68fce4f9d862b6c
Deleted: sha256:cdc6440a971be2985ce94c7e2e0c2df763b58a2ced4ecdb944fcd9b13e7a2aa4
Deleted: sha256:041ac26cd02fa81c8fd73cc616bdeee180de3fd68a649ed1c0339a84cdf7a7c3
Deleted: sha256:376baf7ada4b52ef4c110a023fe7185c4c2c090fa24a5cbd746066333ce3bc46
Deleted: sha256:d254c9b1e23bad05f5cde233b5e91153a0540fa9a797a580d8a360ad12bf63a9
Deleted: sha256:dd5c79fa9b6829fd08ff9943fc1d66bebba3e04246ba394d57c28827fed95af0
Deleted: sha256:8d812a075abf60a83013c37f49058c220c9cdf390266952126e7e60041b305dc
Deleted: sha256:ff1787ee3dcae843dc7dd1933c49350fc84451cf19ed74f4ea72426e17ee7cd1
Deleted: sha256:85ebd294be1553b207ba9120676f4fd140842348ddf1bb5f7602c7a8401f0a13
Deleted: sha256:ded7a220bb058e28ee3254fbba04ca90b679070424424761a53a043b93b612bf
```

3. Создайте docker image, который запускает скрипт с использованием функций из https://github.com/smartiqaorg/geometric_lib.

    1. Данные необходимые для работы скрипта передайте любым удобным способом (например: конфиг файл через docker volume, переменные окружения, перенаправление ввода). Изучите простейшие консольные команды для работы с docker(см. лекцию). Зарегистрируйтесь на DockerHub и выберите необходимые для проекта образы

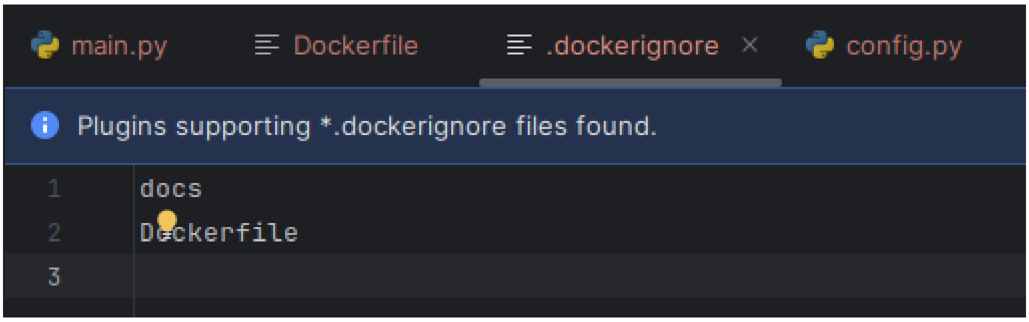


    2. Создать Dockerfile для реализации сборки собственных Docker образов



Plugins supporting Dockerfile files found.

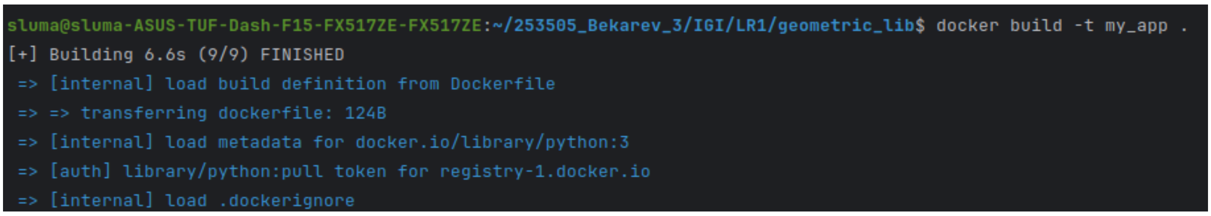
```
FROM python:3

WORKDIR /app

COPY . .

VOLUME ["/app/docs"]

CMD ["python3", "main.py"]
```



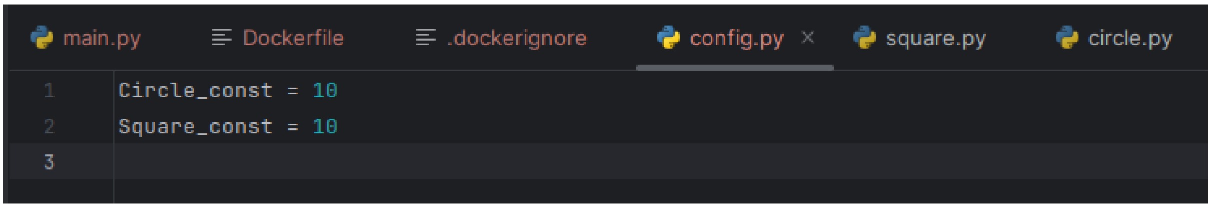
Plugins supporting *.dockerignore files found.

```
docs
Dockerfile
```

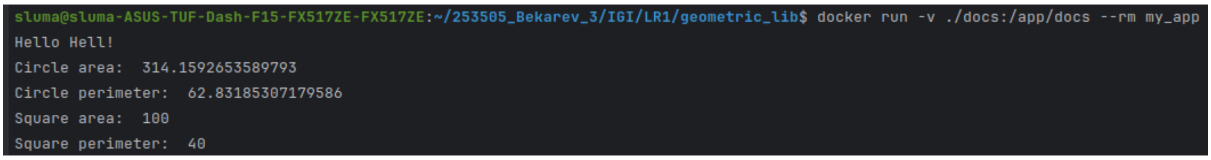
3. Использовать его для создания контейнера. Протестировать использование контейнера



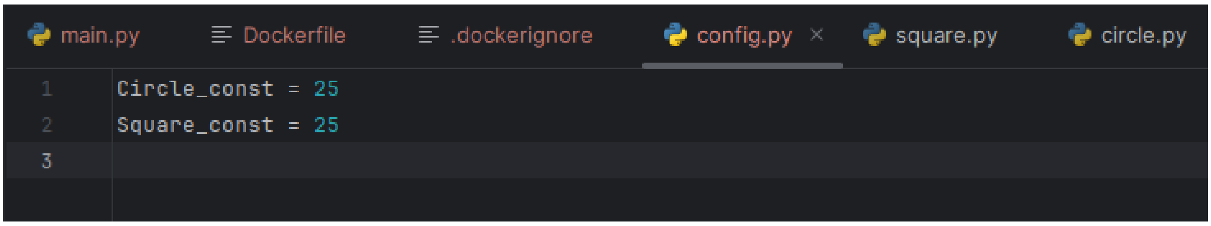
```
sluma@sluma-ASUS-TUF-Dash-F15-FX517ZE-FX517ZE:~/253505_Bekarev_3/IGI/LR1/geometric_lib$ docker build -t my_app .
[+] Building 6.6s (9/9) FINISHED
 => [internal] load build definition from Dockerfile
 => => transferring dockerfile: 124B
 => [internal] load metadata for docker.io/library/python:3
 => [auth] library/python:pull token for registry-1.docker.io
 => [internal] load .dockerignore
```



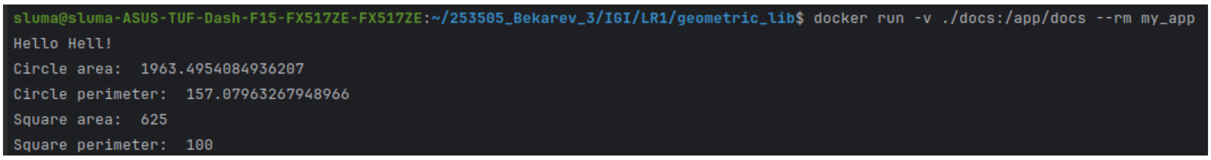
```
Circle_const = 10
Square_const = 10
```



```
sluma@sluma-ASUS-TUF-Dash-F15-FX517ZE-FX517ZE:~/253505_Bekarev_3/IGI/LR1/geometric_lib$ docker run -v ./docs:/app/docs --rm my_app
Hello Hell!
Circle area:  314.1592653589793
Circle perimeter:  62.83185307179586
Square area:  100
Square perimeter:  40
```

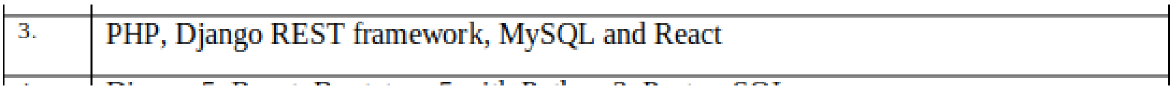


```
Circle_const = 25
Square_const = 25
```

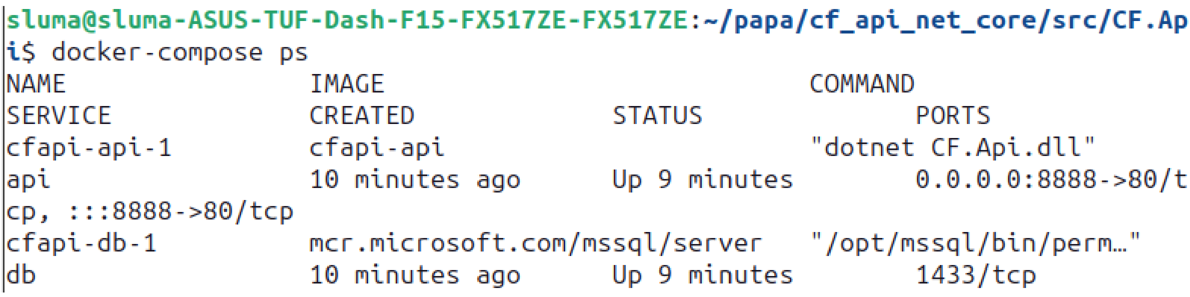


```
sluma@sluma-ASUS-TUF-Dash-F15-FX517ZE-FX517ZE:~/253505_Bekarev_3/IGI/LR1/geometric_lib$ docker run -v ./docs:/app/docs --rm my_app
Hello Hell!
Circle area:  1963.4954084936207
Circle perimeter:  157.07963267948966
Square area:  625
Square perimeter:  100
```

4. Скачать любой доступный проект с GitHub с произвольным стеком технологий (пример – см. индивидуальное задание) или использовать свой, ранее разработанный. Создать для него необходимый контейнер, используя Docker Compose для управления многоконтейнерными приложениями. Запустить проект в контейнере.(Примеры Images: https://hub.docker.com/_/phpmyadmin, https://hub.docker.com/_/mysql, https://hub.docker.com/_/postgres)

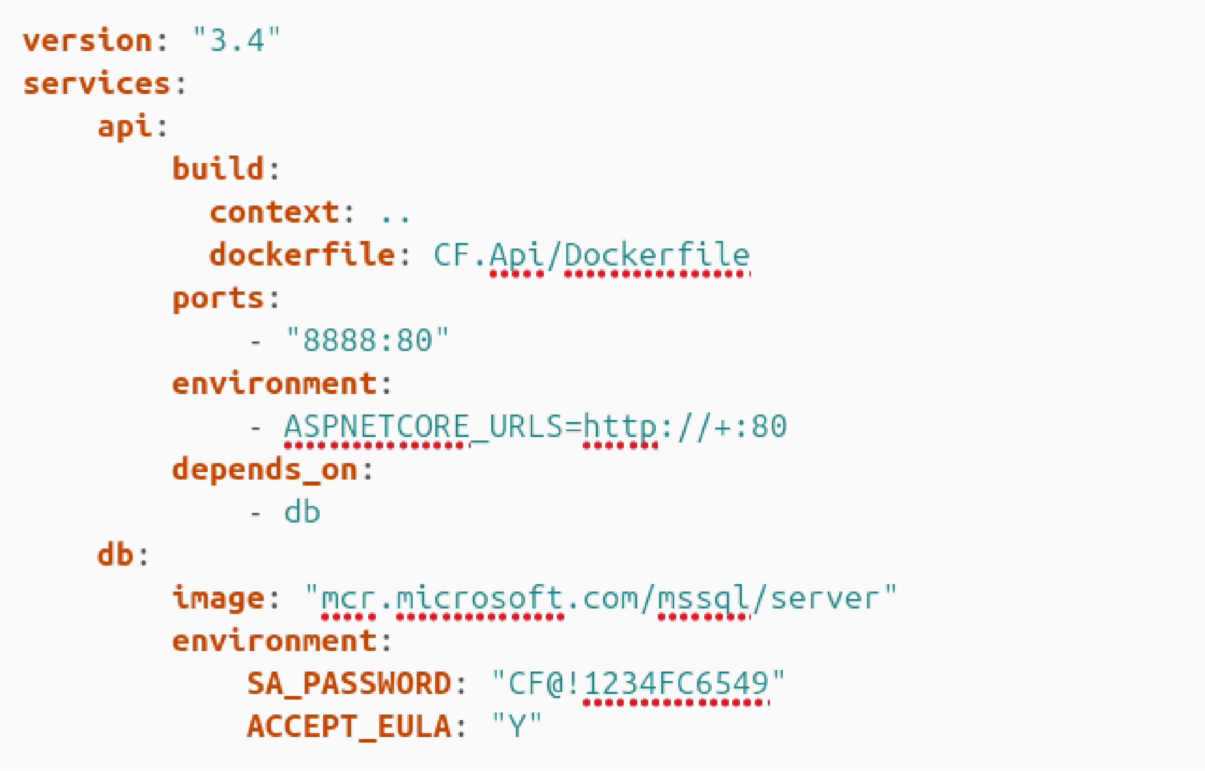


3.	PHP, Django REST framework, MySQL and React

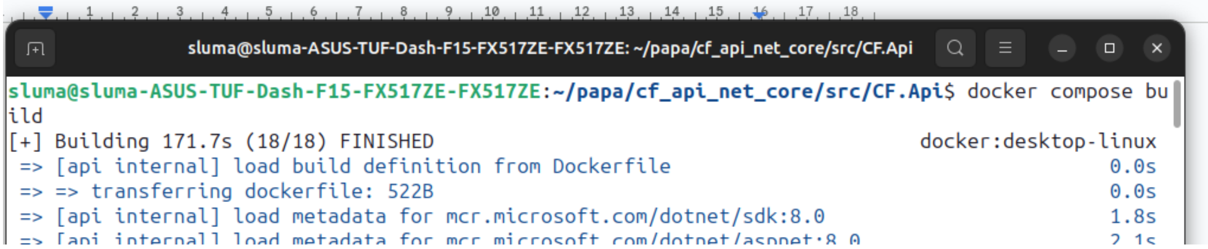


```
sluma@sluma-ASUS-TUF-Dash-F15-FX517ZE-FX517ZE:~/papa/cf_api_net_core/src/CF.Ap
i$ docker-compose ps
NAME                IMAGE                         COMMAND
SERVICE             CREATED             STATUS              PORTS
cfapi-api-1         cfapi-api                     "dotnet CF.Api.dll"
api                 10 minutes ago      Up 9 minutes        0.0.0.0:8888->80/t
cp, :::8888->80/tcp
cfapi-db-1          mcr.microsoft.com/mssql/server   "/opt/mssql/bin/perm…"
db                  10 minutes ago      Up 9 minutes        1433/tcp
```

5. Настроить сети и тома для обеспечения связи между контейнерами и сохранения данных (исходные данные, логин, пароль и т.д.)

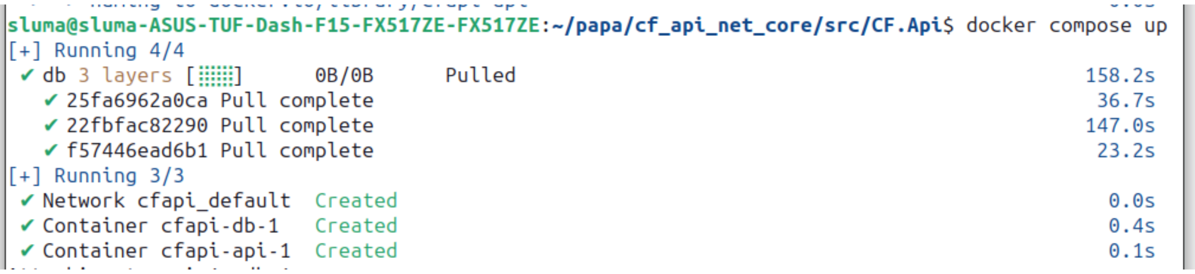


```yaml
version: "3.4"
services:
    api:
        build:
          context: ..
          dockerfile: CF.Api/Dockerfile
        ports:
            - "8888:80"
        environment:
            - ASPNETCORE_URLS=http://+:80
        depends_on:
            - db
    db:
        image: "mcr.microsoft.com/mssql/server"
        environment:
            SA_PASSWORD: "CF@!1234FC6549"
            ACCEPT_EULA: "Y"
```

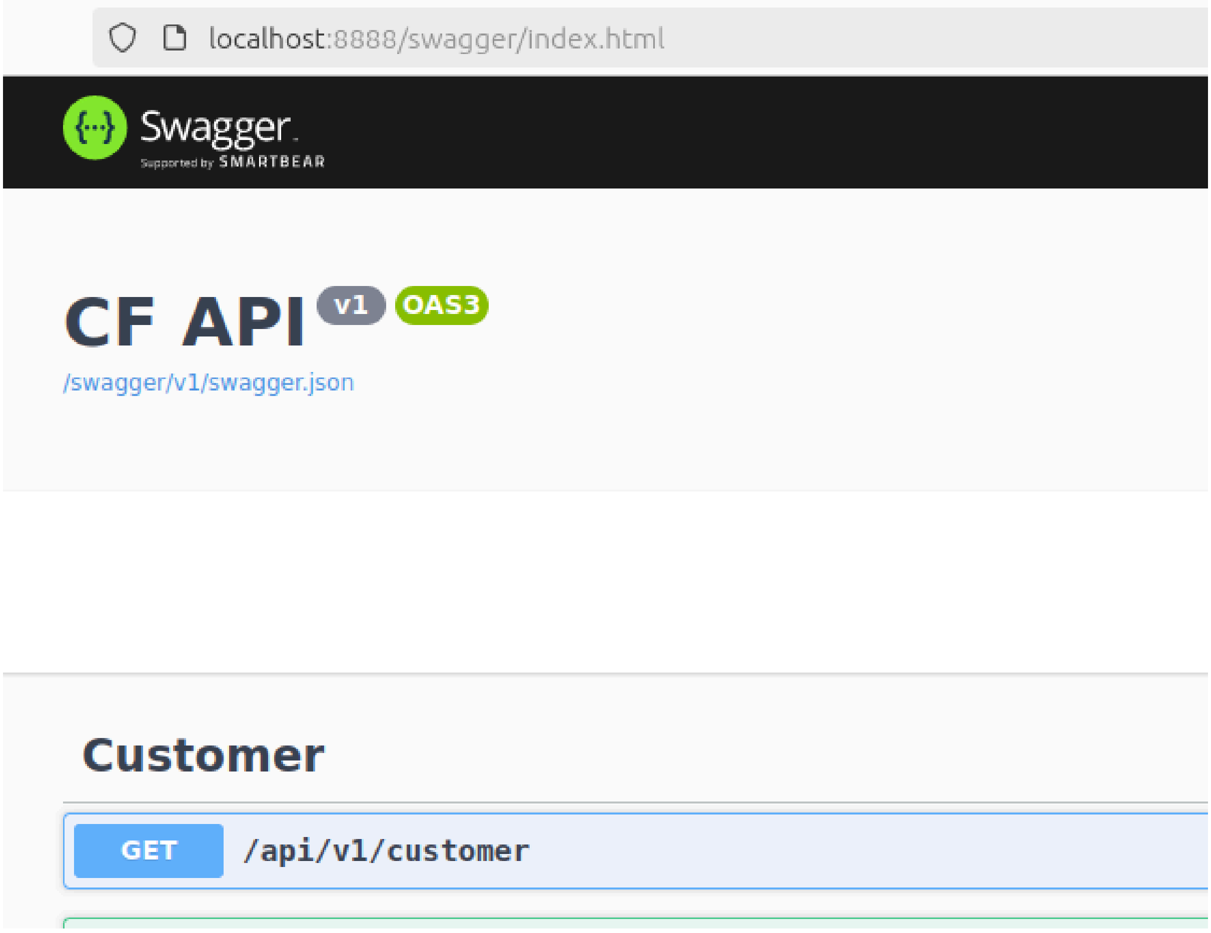


sluma@sluma-ASUS-TUF-Dash-F15-FX517ZE-FX517ZE: ~/papa/cf_api_net_core/src/CF.Api

```
sluma@sluma-ASUS-TUF-Dash-F15-FX517ZE-FX517ZE:~/papa/cf_api_net_core/src/CF.Api$ docker compose bu
ild
[+] Building 171.7s (18/18) FINISHED                                      docker:desktop-linux
 => [api internal] load build definition from Dockerfile                                  0.0s
 => => transferring dockerfile: 522B                                                      0.0s
 => [api internal] load metadata for mcr.microsoft.com/dotnet/sdk:8.0                     1.8s
 => [api internal] load metadata for mcr.microsoft.com/dotnet/aspnet:8.0                  2.1s
```



```
 => => naming to docker.io/library/cfapi-api                                              0.0s
sluma@sluma-ASUS-TUF-Dash-F15-FX517ZE-FX517ZE:~/papa/cf_api_net_core/src/CF.Api$ docker compose up
[+] Running 4/4
 ✔ db 3 layers [⣿⣿⣿]      0B/0B      Pulled                                             158.2s
   ✔ 25fa6962a0ca Pull complete                                                          36.7s
   ✔ 22fbfac82290 Pull complete                                                         147.0s
   ✔ f57446ead6b1 Pull complete                                                          23.2s
[+] Running 3/3
 ✔ Network cfapi_default    Created                                                       0.0s
 ✔ Container cfapi-db-1     Created                                                       0.4s
 ✔ Container cfapi-api-1    Created                                                       0.1s
```



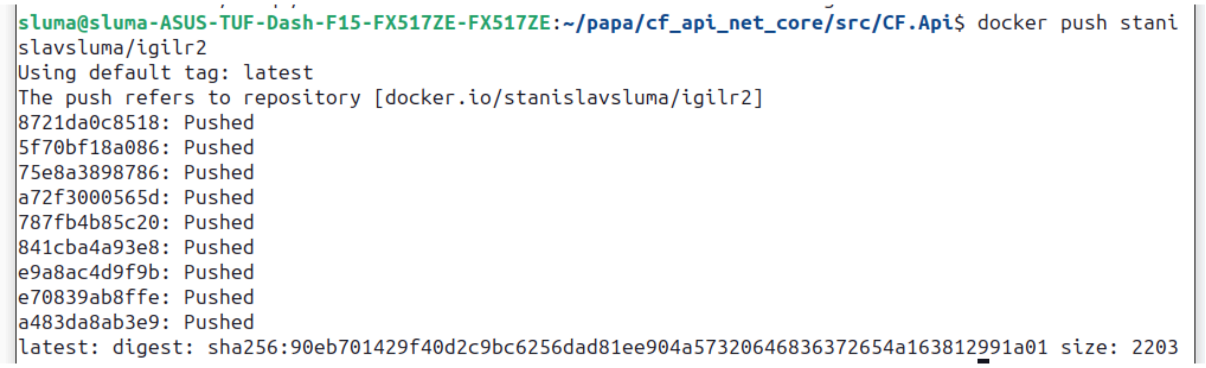
CF API v1 OAS3

/swagger/v1/swagger.json

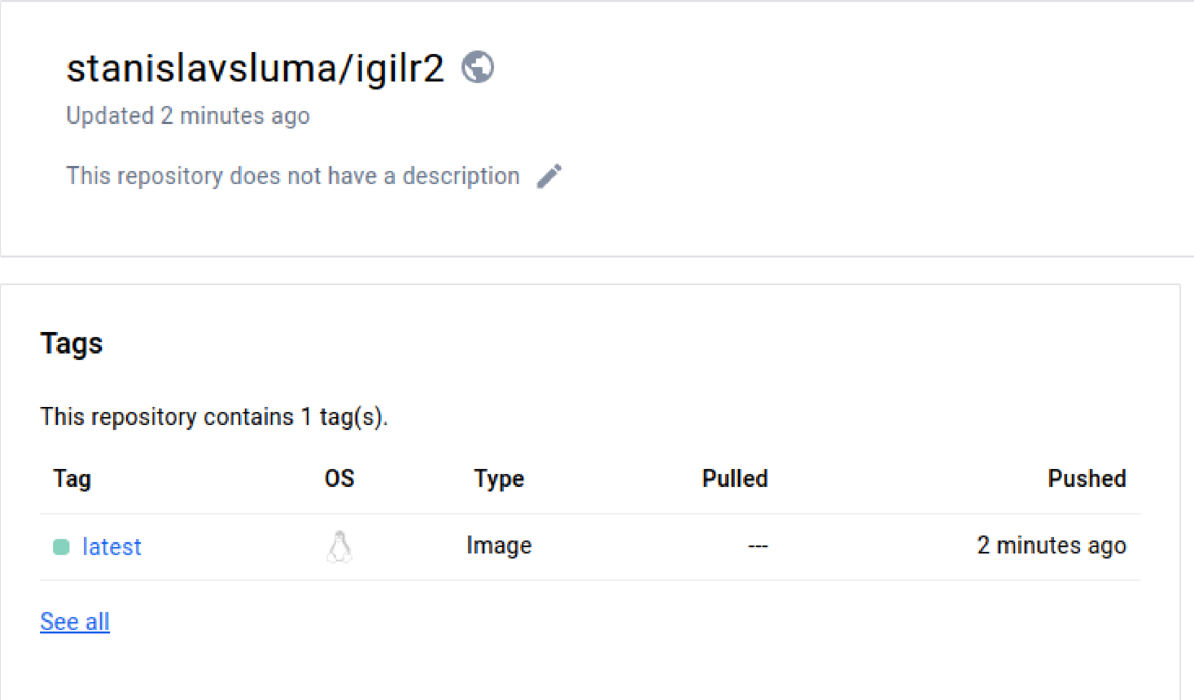
Customer

GET /api/v1/customer

6. Разместите результат в созданный репозиторий в DockerHub



```
sluma@sluma-ASUS-TUF-Dash-F15-FX517ZE-FX517ZE:~/papa/cf_api_net_core/src/CF.Api$ docker push stani
slavsluma/igilr2
Using default tag: latest
The push refers to repository [docker.io/stanislavsluma/igilr2]
8721da0c8518: Pushed
5f70bf18a086: Pushed
75e8a3898786: Pushed
a72f3000565d: Pushed
787fb4b85c20: Pushed
841cba4a93e8: Pushed
e9a8ac4d9f9b: Pushed
e70839ab8ffe: Pushed
a483da8ab3e9: Pushed
latest: digest: sha256:90eb701429f40d2c9bc6256dad81ee904a57320646836372654a163812991a01 size: 2203
```



stanislavsluma/igilr2

Updated 2 minutes ago

This repository does not have a description

Tags

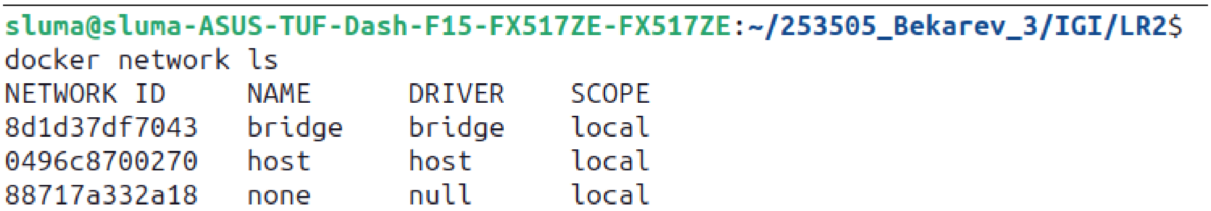
This repository contains 1 tag(s).

Tag	OS	Type	Pulled	Pushed
latest	🐧	Image	---	2 minutes ago

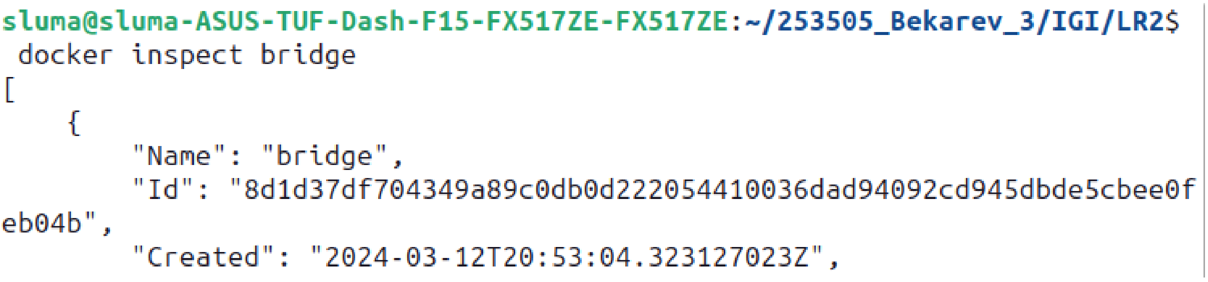
See all

7. Выполните следующие действия с целью изучить особенности сетевого взаимодействия:

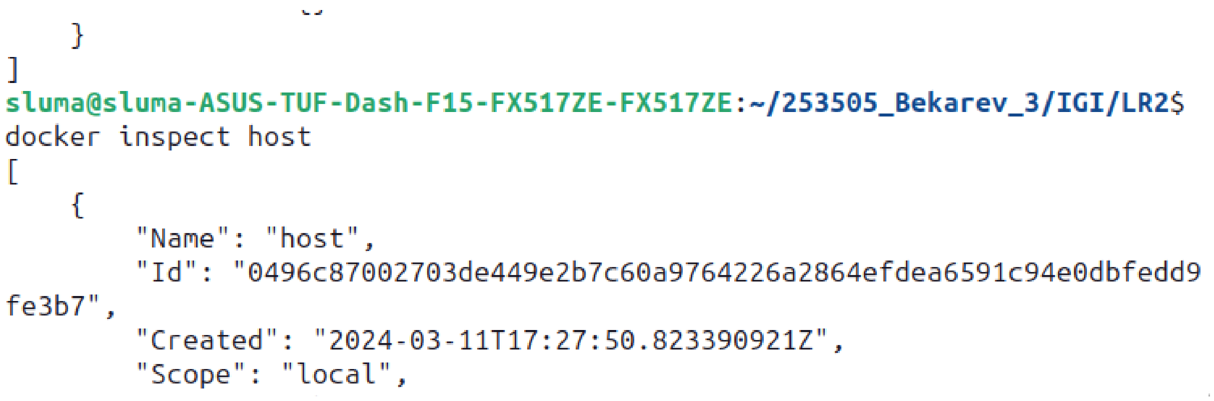
- Получить информацию о всех сетях, работающих на текущем хосте и подробности о каждом типе сети



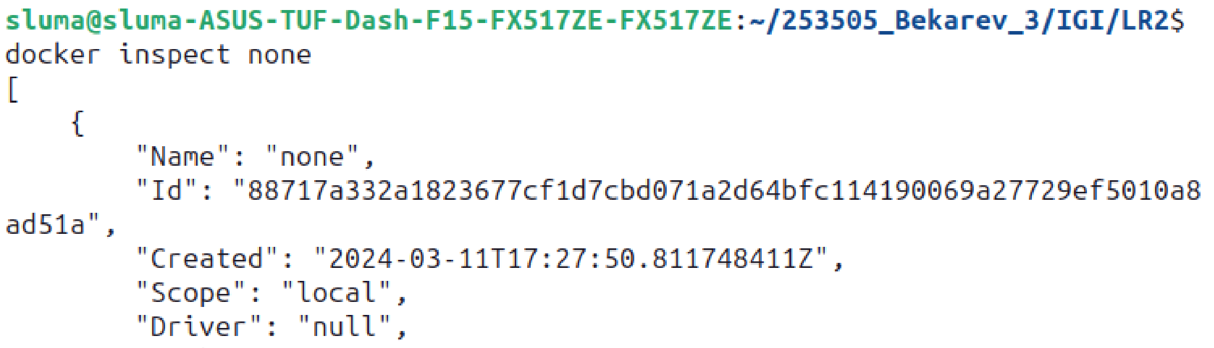
```
sluma@sluma-ASUS-TUF-Dash-F15-FX517ZE-FX517ZE:~/253505_Bekarev_3/IGI/LR2$
docker network ls
NETWORK ID     NAME      DRIVER    SCOPE
8d1d37df7043   bridge    bridge    local
0496c8700270   host      host      local
88717a332a18   none      null      local
```



```
sluma@sluma-ASUS-TUF-Dash-F15-FX517ZE-FX517ZE:~/253505_Bekarev_3/IGI/LR2$
 docker inspect bridge
[
    {
        "Name": "bridge",
        "Id": "8d1d37df704349a89c0db0d222054410036dad94092cd945dbde5cbee0f
eb04b",
        "Created": "2024-03-12T20:53:04.323127023Z",
```

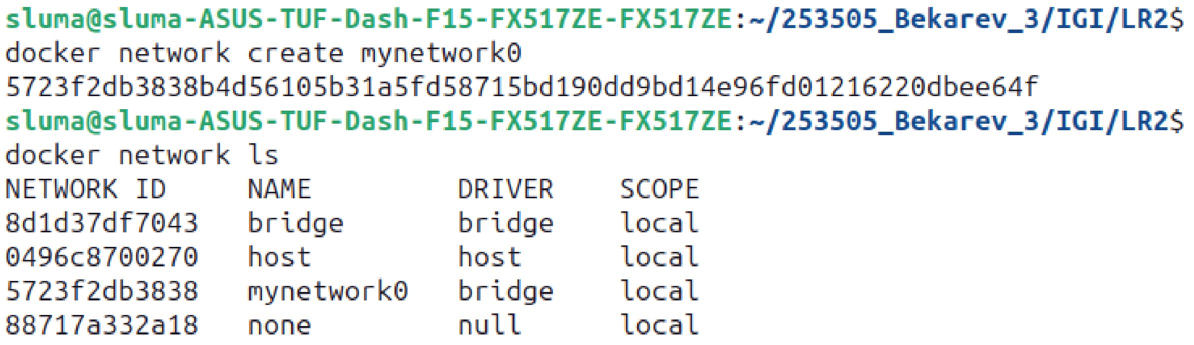


```
    }
]
sluma@sluma-ASUS-TUF-Dash-F15-FX517ZE-FX517ZE:~/253505_Bekarev_3/IGI/LR2$
docker inspect host
[
    {
        "Name": "host",
        "Id": "0496c87002703de449e2b7c60a9764226a2864efdea6591c94e0dbfedd9
fe3b7",
        "Created": "2024-03-11T17:27:50.823390921Z",
        "Scope": "local",
```

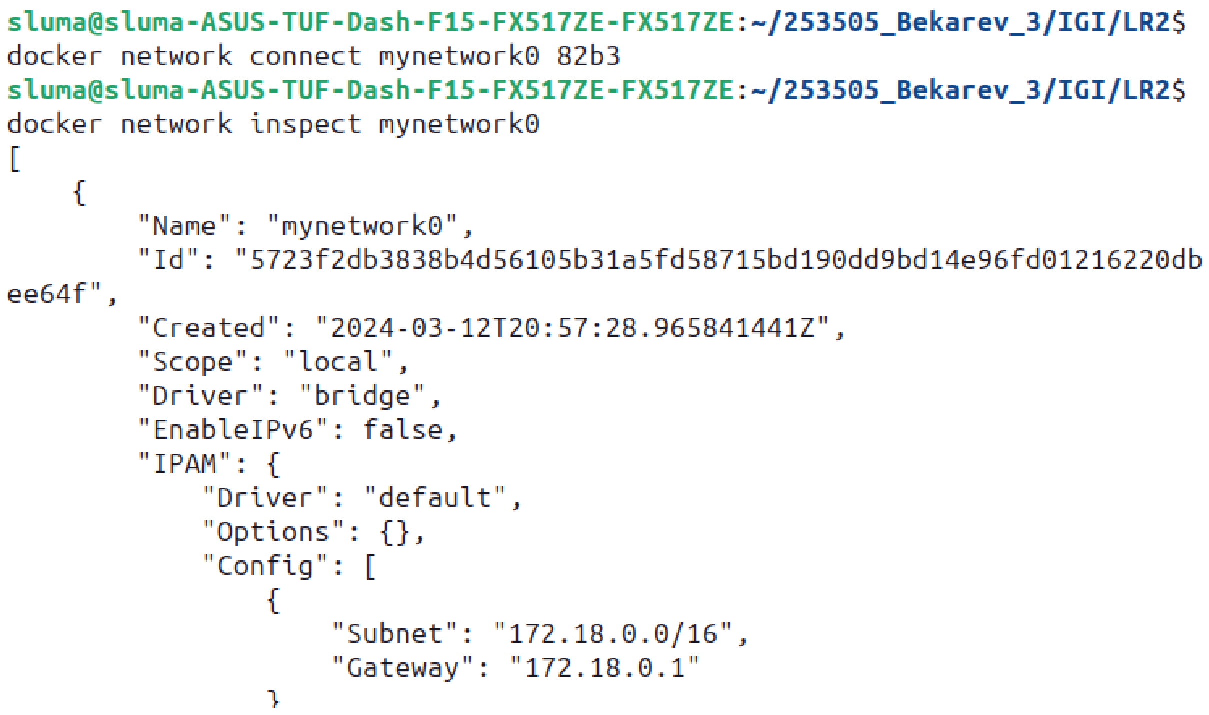


```
sluma@sluma-ASUS-TUF-Dash-F15-FX517ZE-FX517ZE:~/253505_Bekarev_3/IGI/LR2$
docker inspect none
[
    {
        "Name": "none",
        "Id": "88717a332a1823677cf1d7cbd071a2d64bfc114190069a27729ef5010a8
ad51a",
        "Created": "2024-03-11T17:27:50.811748411Z",
        "Scope": "local",
        "Driver": "null",
```

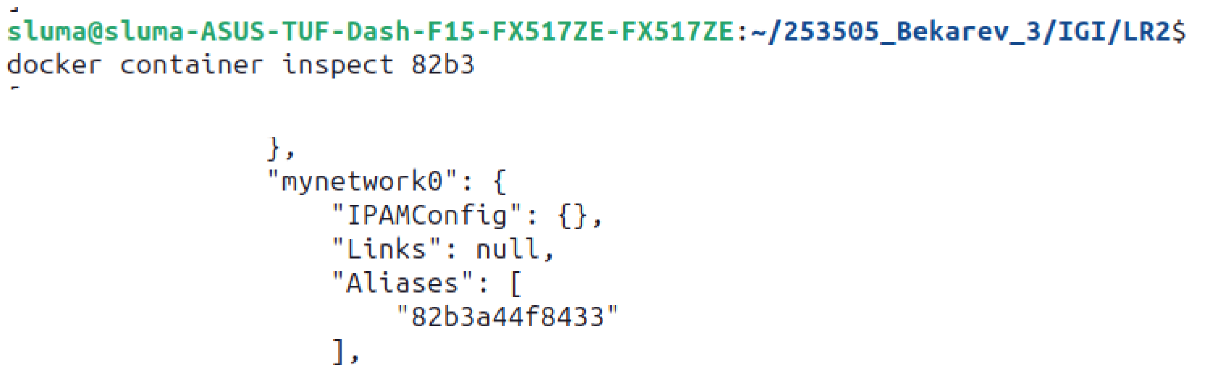
- Создать свою собственную сеть bridge, проверить, создана ли она, запустить Docker-контейнер в созданной сети, вывести о ней всю информацию(включая IP-адрес контейнера), отключить сеть от контейнера



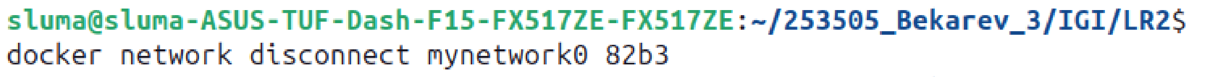
```
sluma@sluma-ASUS-TUF-Dash-F15-FX517ZE-FX517ZE:~/253505_Bekarev_3/IGI/LR2$
docker network create mynetwork0
5723f2db3838b4d56105b31a5fd58715bd190dd9bd14e96fd01216220dbee64f
sluma@sluma-ASUS-TUF-Dash-F15-FX517ZE-FX517ZE:~/253505_Bekarev_3/IGI/LR2$
docker network ls
NETWORK ID     NAME         DRIVER    SCOPE
8d1d37df7043   bridge       bridge    local
0496c8700270   host         host      local
5723f2db3838   mynetwork0   bridge    local
88717a332a18   none         null      local
```



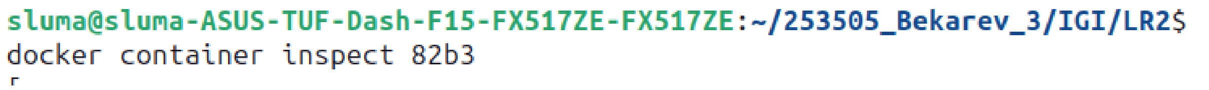
```
sluma@sluma-ASUS-TUF-Dash-F15-FX517ZE-FX517ZE:~/253505_Bekarev_3/IGI/LR2$
docker network connect mynetwork0 82b3
sluma@sluma-ASUS-TUF-Dash-F15-FX517ZE-FX517ZE:~/253505_Bekarev_3/IGI/LR2$
docker network inspect mynetwork0
[
    {
        "Name": "mynetwork0",
        "Id": "5723f2db3838b4d56105b31a5fd58715bd190dd9bd14e96fd01216220db
ee64f",
        "Created": "2024-03-12T20:57:28.965841441Z",
        "Scope": "local",
        "Driver": "bridge",
        "EnableIPv6": false,
        "IPAM": {
            "Driver": "default",
            "Options": {},
            "Config": [
                {
                    "Subnet": "172.18.0.0/16",
                    "Gateway": "172.18.0.1"
                }
```



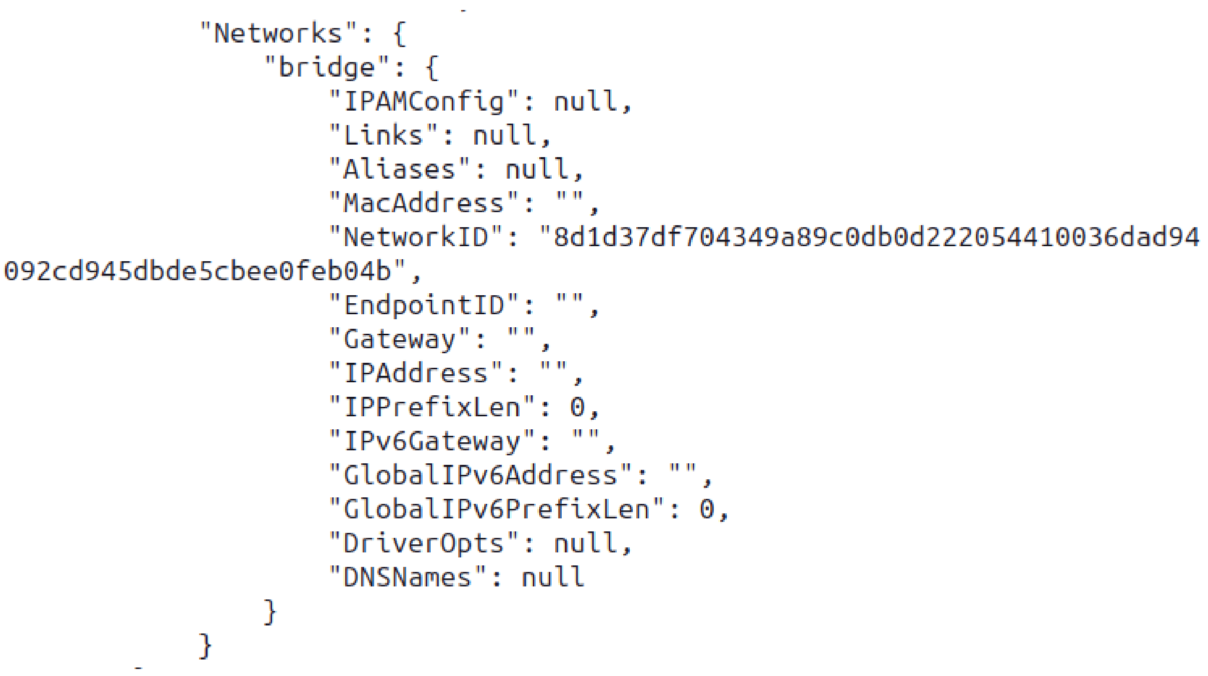
```
sluma@sluma-ASUS-TUF-Dash-F15-FX517ZE-FX517ZE:~/253505_Bekarev_3/IGI/LR2$
docker container inspect 82b3

                },
                "mynetwork0": {
                    "IPAMConfig": {},
                    "Links": null,
                    "Aliases": [
                        "82b3a44f8433"
                    ],
```



```
sluma@sluma-ASUS-TUF-Dash-F15-FX517ZE-FX517ZE:~/253505_Bekarev_3/IGI/LR2$
docker network disconnect mynetwork0 82b3
```

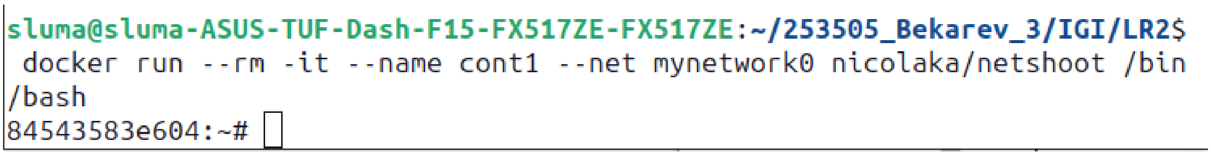


```
sluma@sluma-ASUS-TUF-Dash-F15-FX517ZE-FX517ZE:~/253505_Bekarev_3/IGI/LR2$
docker container inspect 82b3
```

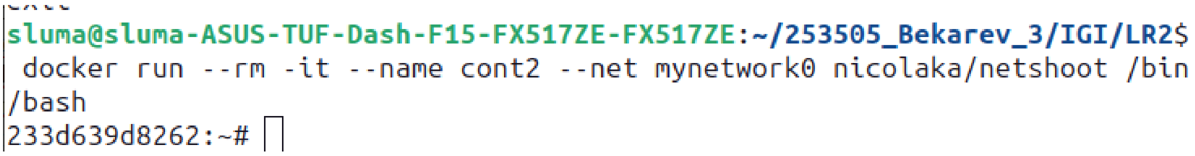


```
            "Networks": {
                "bridge": {
                    "IPAMConfig": null,
                    "Links": null,
                    "Aliases": null,
                    "MacAddress": "",
                    "NetworkID": "8d1d37df704349a89c0db0d222054410036dad94
092cd945dbde5cbee0feb04b",
                    "EndpointID": "",
                    "Gateway": "",
                    "IPAddress": "",
                    "IPPrefixLen": 0,
                    "IPv6Gateway": "",
                    "GlobalIPv6Address": "",
                    "GlobalIPv6PrefixLen": 0,
                    "DriverOpts": null,
                    "DNSNames": null
                }
            }
```

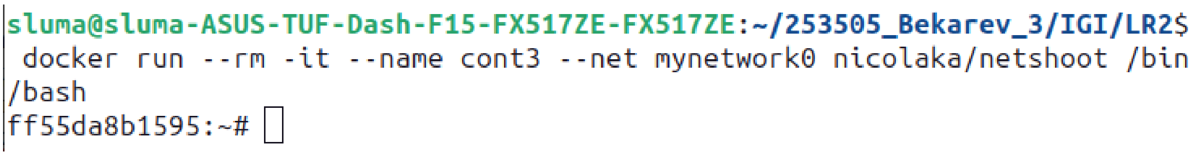
- Создать еще одну сеть bridge, вывести о ней всю информацию, запустить в ней три контейнера, подключиться к любому из контейнеров и пропинговать два других из оболочки контейнера, убедиться, что между контейнерами происходит общение по IP-адресу



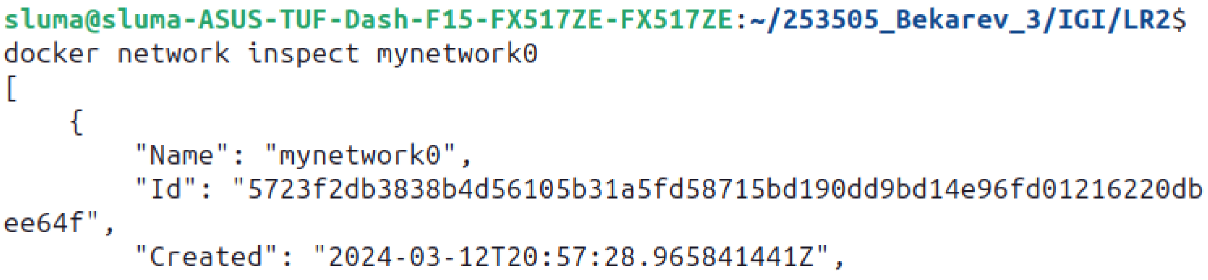
```
sluma@sluma-ASUS-TUF-Dash-F15-FX517ZE-FX517ZE:~/253505_Bekarev_3/IGI/LR2$
 docker run --rm -it --name cont1 --net mynetwork0 nicolaka/netshoot /bin
/bash
84543583e604:~#
```



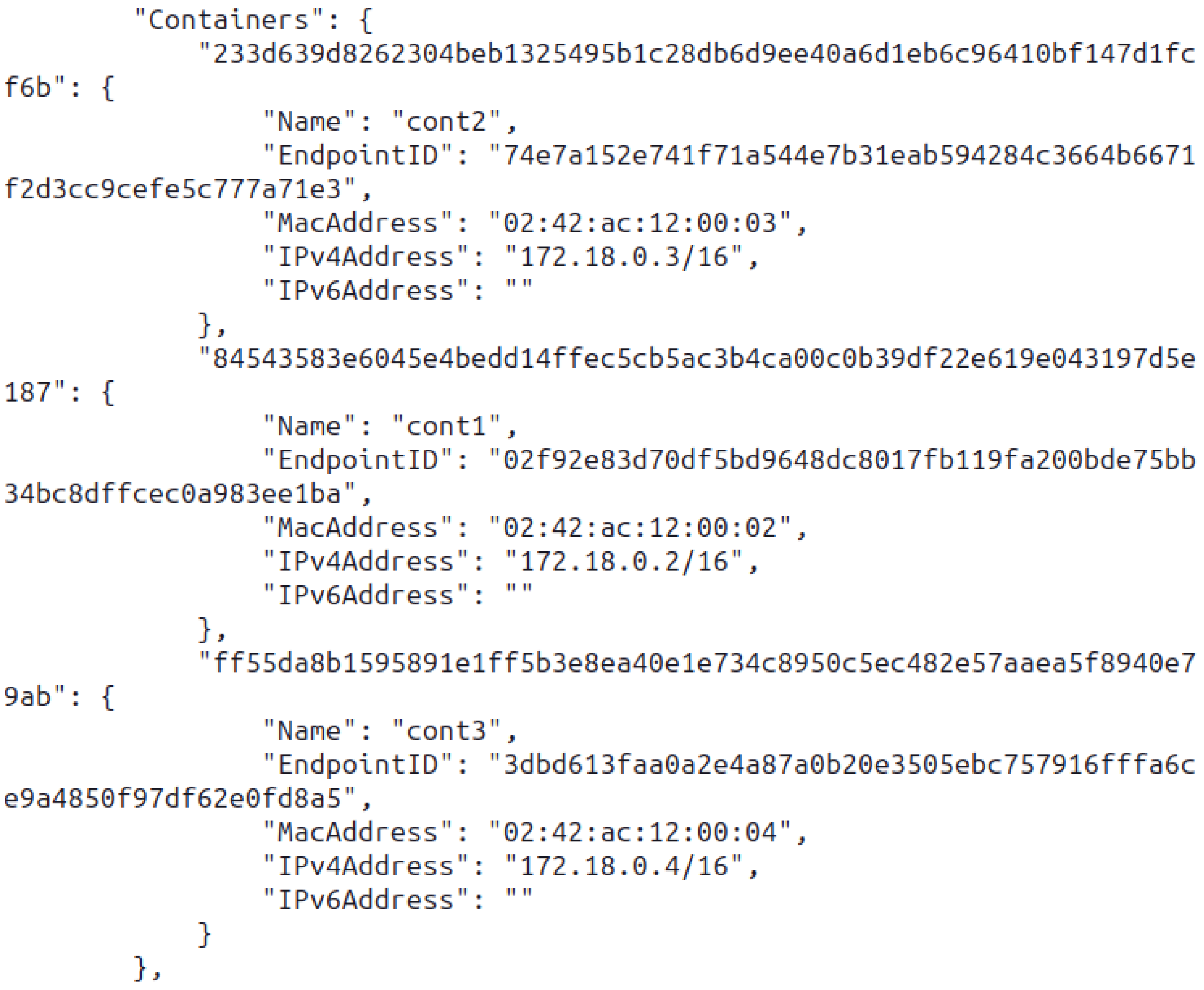
```
sluma@sluma-ASUS-TUF-Dash-F15-FX517ZE-FX517ZE:~/253505_Bekarev_3/IGI/LR2$
 docker run --rm -it --name cont2 --net mynetwork0 nicolaka/netshoot /bin
/bash
233d639d8262:~#
```



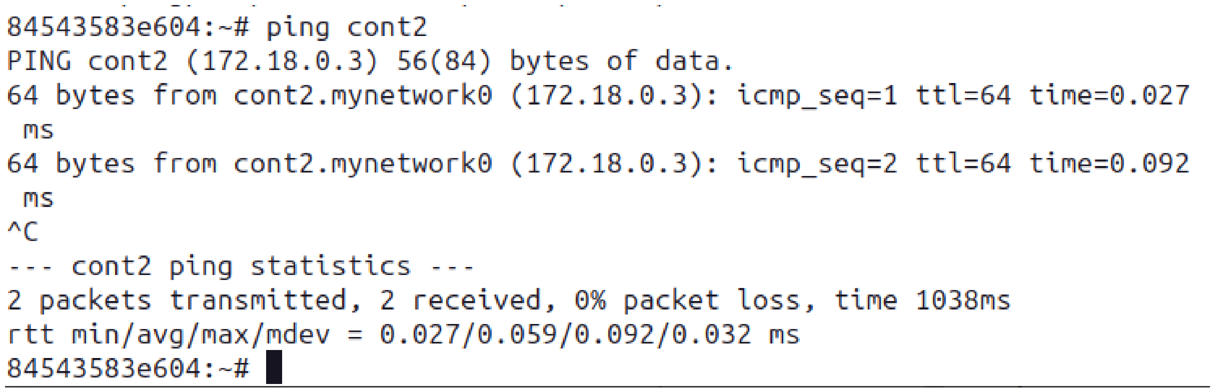
```
sluma@sluma-ASUS-TUF-Dash-F15-FX517ZE-FX517ZE:~/253505_Bekarev_3/IGI/LR2$
 docker run --rm -it --name cont3 --net mynetwork0 nicolaka/netshoot /bin
/bash
ff55da8b1595:~#
```



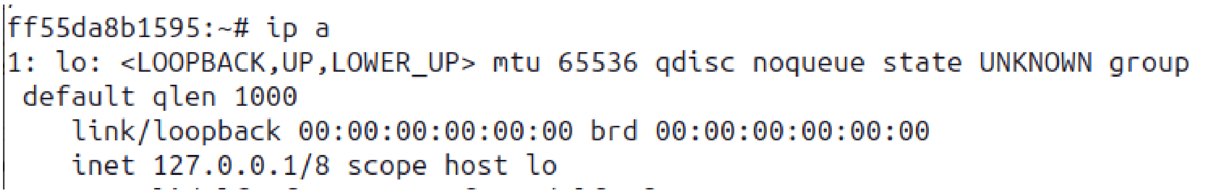
```
sluma@sluma-ASUS-TUF-Dash-F15-FX517ZE-FX517ZE:~/253505_Bekarev_3/IGI/LR2$
docker network inspect mynetwork0
[
    {
        "Name": "mynetwork0",
        "Id": "5723f2db3838b4d56105b31a5fd58715bd190dd9bd14e96fd01216220db
ee64f",
        "Created": "2024-03-12T20:57:28.965841441Z",
```



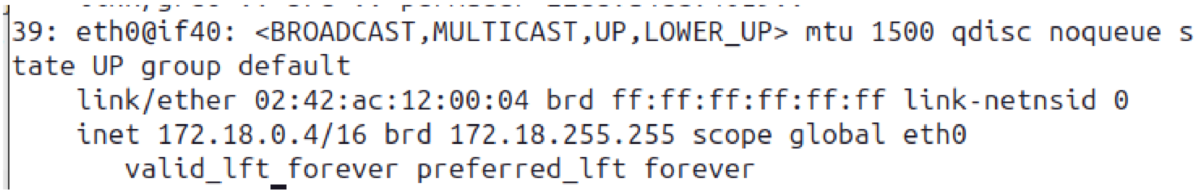
```
        "Containers": {
            "233d639d8262304beb1325495b1c28db6d9ee40a6d1eb6c96410bf147d1fc
f6b": {
                "Name": "cont2",
                "EndpointID": "74e7a152e741f71a544e7b31eab594284c3664b6671
f2d3cc9cefe5c777a71e3",
                "MacAddress": "02:42:ac:12:00:03",
                "IPv4Address": "172.18.0.3/16",
                "IPv6Address": ""
            },
            "84543583e6045e4bedd14ffec5cb5ac3b4ca00c0b39df22e619e043197d5e
187": {
                "Name": "cont1",
                "EndpointID": "02f92e83d70df5bd9648dc8017fb119fa200bde75bb
34bc8dffcec0a983ee1ba",
                "MacAddress": "02:42:ac:12:00:02",
                "IPv4Address": "172.18.0.2/16",
                "IPv6Address": ""
            },
            "ff55da8b1595891e1ff5b3e8ea40e1e734c8950c5ec482e57aaea5f8940e7
9ab": {
                "Name": "cont3",
                "EndpointID": "3dbd613faa0a2e4a87a0b20e3505ebc757916fffa6c
e9a4850f97df62e0fd8a5",
                "MacAddress": "02:42:ac:12:00:04",
                "IPv4Address": "172.18.0.4/16",
                "IPv6Address": ""
            }
        },
```



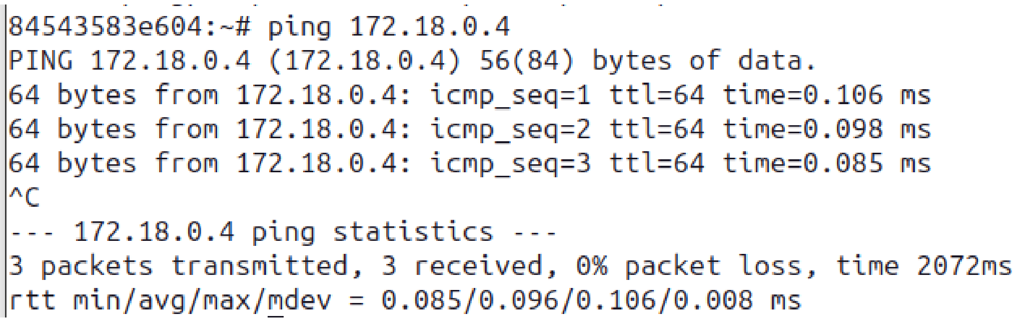
```
84543583e604:~# ping cont2
PING cont2 (172.18.0.3) 56(84) bytes of data.
64 bytes from cont2.mynetwork0 (172.18.0.3): icmp_seq=1 ttl=64 time=0.027
 ms
64 bytes from cont2.mynetwork0 (172.18.0.3): icmp_seq=2 ttl=64 time=0.092
 ms
^C
--- cont2 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1038ms
rtt min/avg/max/mdev = 0.027/0.059/0.092/0.032 ms
84543583e604:~# 
```



```
ff55da8b1595:~# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group
 default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
```

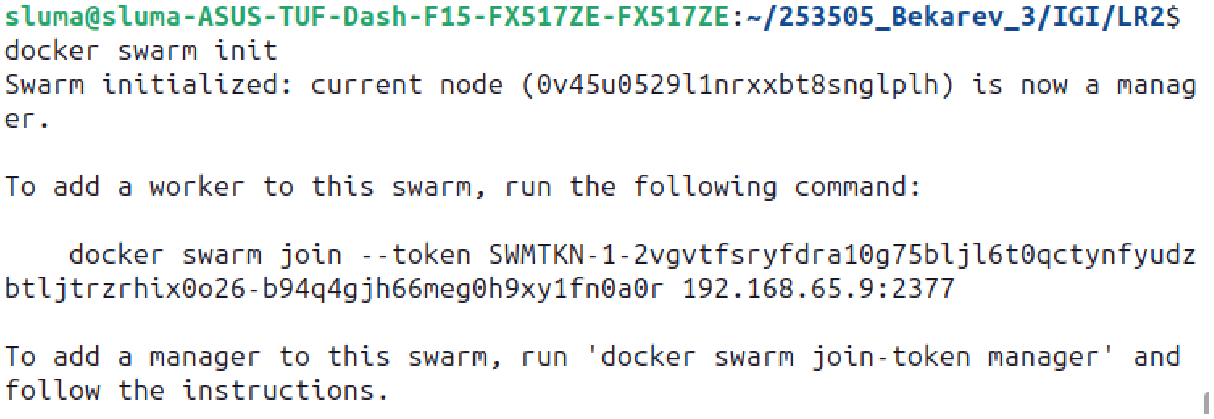


```
39: eth0@if40: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue s
tate UP group default
    link/ether 02:42:ac:12:00:04 brd ff:ff:ff:ff:ff:ff link-netnsid 0
    inet 172.18.0.4/16 brd 172.18.255.255 scope global eth0
       valid_lft forever preferred_lft forever
```

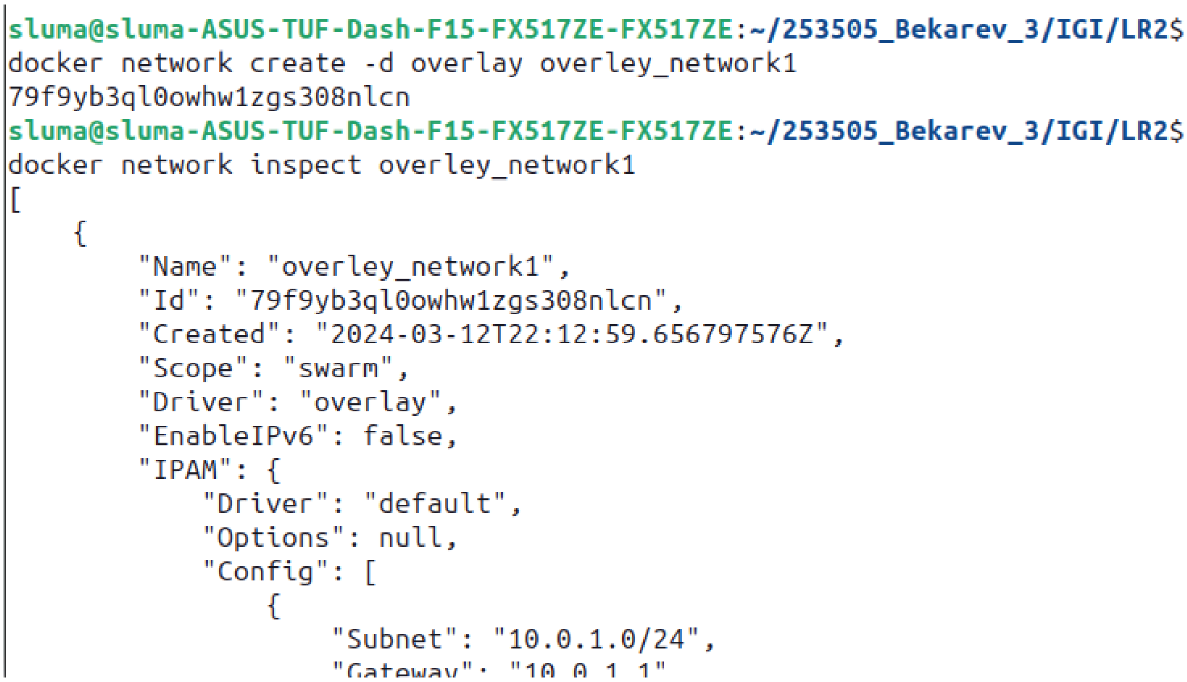


```
84543583e604:~# ping 172.18.0.4
PING 172.18.0.4 (172.18.0.4) 56(84) bytes of data.
64 bytes from 172.18.0.4: icmp_seq=1 ttl=64 time=0.106 ms
64 bytes from 172.18.0.4: icmp_seq=2 ttl=64 time=0.098 ms
64 bytes from 172.18.0.4: icmp_seq=3 ttl=64 time=0.085 ms
^C
--- 172.18.0.4 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2072ms
rtt min/avg/max/mdev = 0.085/0.096/0.106/0.008 ms
```

- Создать свою собственную сеть overlay, проверить, создана ли она, вывести о ней всю информацию

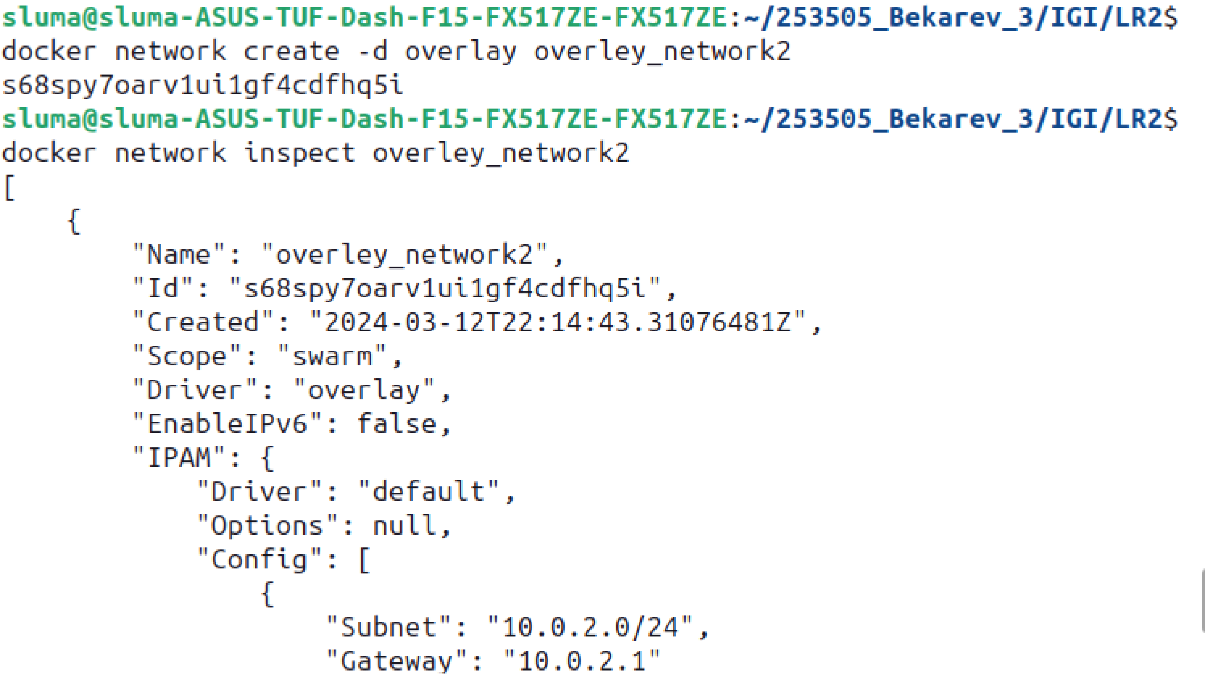


```
sluma@sluma-ASUS-TUF-Dash-F15-FX517ZE-FX517ZE:~/253505_Bekarev_3/IGI/LR2$
docker swarm init
Swarm initialized: current node (0v45u0529l1nrxxbt8snglplh) is now a manag
er.

To add a worker to this swarm, run the following command:

    docker swarm join --token SWMTKN-1-2vgvtfsryfdra10g75bljl6t0qctynfyudz
btljtrzrhix0o26-b94q4gjh66meg0h9xy1fn0a0r 192.168.65.9:2377

To add a manager to this swarm, run 'docker swarm join-token manager' and
follow the instructions.
```

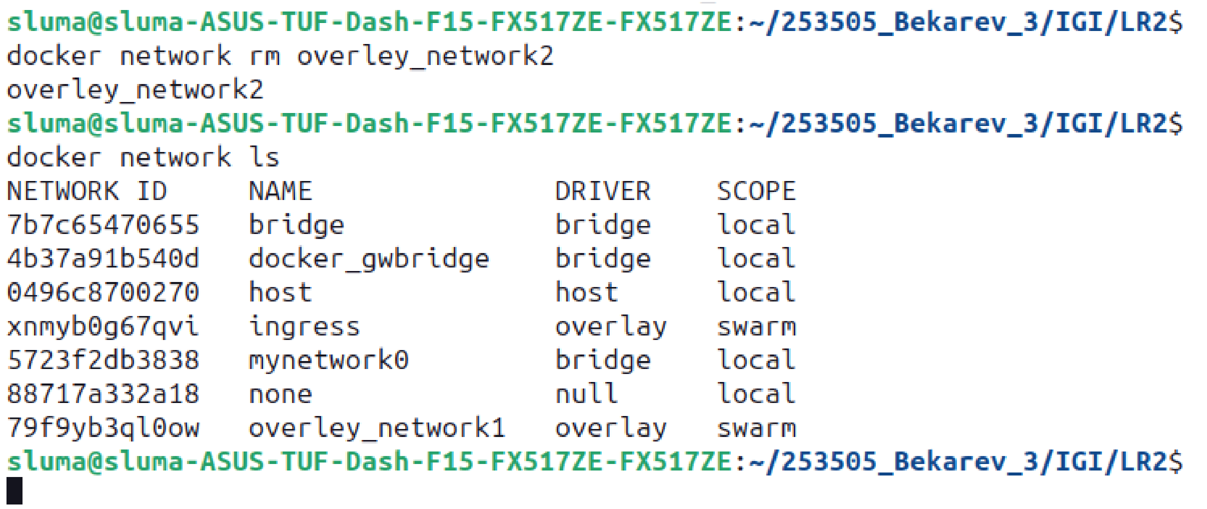


```
sluma@sluma-ASUS-TUF-Dash-F15-FX517ZE-FX517ZE:~/253505_Bekarev_3/IGI/LR2$
docker network create -d overlay overley_network1
79f9yb3ql0owhw1zgs308nlcn
sluma@sluma-ASUS-TUF-Dash-F15-FX517ZE-FX517ZE:~/253505_Bekarev_3/IGI/LR2$
docker network inspect overley_network1
[
    {
        "Name": "overley_network1",
        "Id": "79f9yb3ql0owhw1zgs308nlcn",
        "Created": "2024-03-12T22:12:59.656797576Z",
        "Scope": "swarm",
        "Driver": "overlay",
        "EnableIPv6": false,
        "IPAM": {
            "Driver": "default",
            "Options": null,
            "Config": [
                {
                    "Subnet": "10.0.1.0/24",
                    "Gateway": "10.0.1.1"
```

- Создать еще одну сеть overlay, проверить, создана ли она, вывести о ней всю информацию, удалить сеть

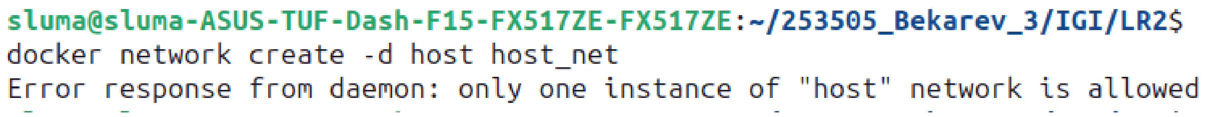


```
sluma@sluma-ASUS-TUF-Dash-F15-FX517ZE-FX517ZE:~/253505_Bekarev_3/IGI/LR2$
docker network create -d overlay overley_network2
s68spy7oarv1ui1gf4cdfhq5i
sluma@sluma-ASUS-TUF-Dash-F15-FX517ZE-FX517ZE:~/253505_Bekarev_3/IGI/LR2$
docker network inspect overley_network2
[
    {
        "Name": "overley_network2",
        "Id": "s68spy7oarv1ui1gf4cdfhq5i",
        "Created": "2024-03-12T22:14:43.31076481Z",
        "Scope": "swarm",
        "Driver": "overlay",
        "EnableIPv6": false,
        "IPAM": {
            "Driver": "default",
            "Options": null,
            "Config": [
                {
                    "Subnet": "10.0.2.0/24",
                    "Gateway": "10.0.2.1"
```



```
sluma@sluma-ASUS-TUF-Dash-F15-FX517ZE-FX517ZE:~/253505_Bekarev_3/IGI/LR2$
docker network rm overley_network2
overley_network2
sluma@sluma-ASUS-TUF-Dash-F15-FX517ZE-FX517ZE:~/253505_Bekarev_3/IGI/LR2$
docker network ls
NETWORK ID     NAME               DRIVER    SCOPE
7b7c65470655   bridge             bridge    local
4b37a91b540d   docker_gwbridge    bridge    local
0496c8700270   host               host      local
xnmyb0g67qvi   ingress            overlay   swarm
5723f2db3838   mynetwork0         bridge    local
88717a332a18   none               null      local
79f9yb3ql0ow   overley_network1   overlay   swarm
sluma@sluma-ASUS-TUF-Dash-F15-FX517ZE-FX517ZE:~/253505_Bekarev_3/IGI/LR2$
```

- Попробовать создать сеть host, сохранить результат в отчет.



```
sluma@sluma-ASUS-TUF-Dash-F15-FX517ZE-FX517ZE:~/253505_Bekarev_3/IGI/LR2$
docker network create -d host host_net
Error response from daemon: only one instance of "host" network is allowed
```