Министерство образования Республики Беларусь

Учреждение образования «Белорусский государственный университет

информатики и радиоэлектроники»

Факультет компьютерных систем и сетей

Кафедра информатики

Дисциплина «Избранные главы информатики»

**ОТЧЕТ**

к лабораторной работе №2

БГУИР 1-40 04 01

|  |
| --- |
| Выполнил студент группы 253504  Лавренова Анастасия Сергеевна |
|  |
| (дата, подпись студента) |
| Проверил Жвакина Анна Васильевна |
|  |
| (дата, подпись преподавателя) |

Минск 2024

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

**Задание**:

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

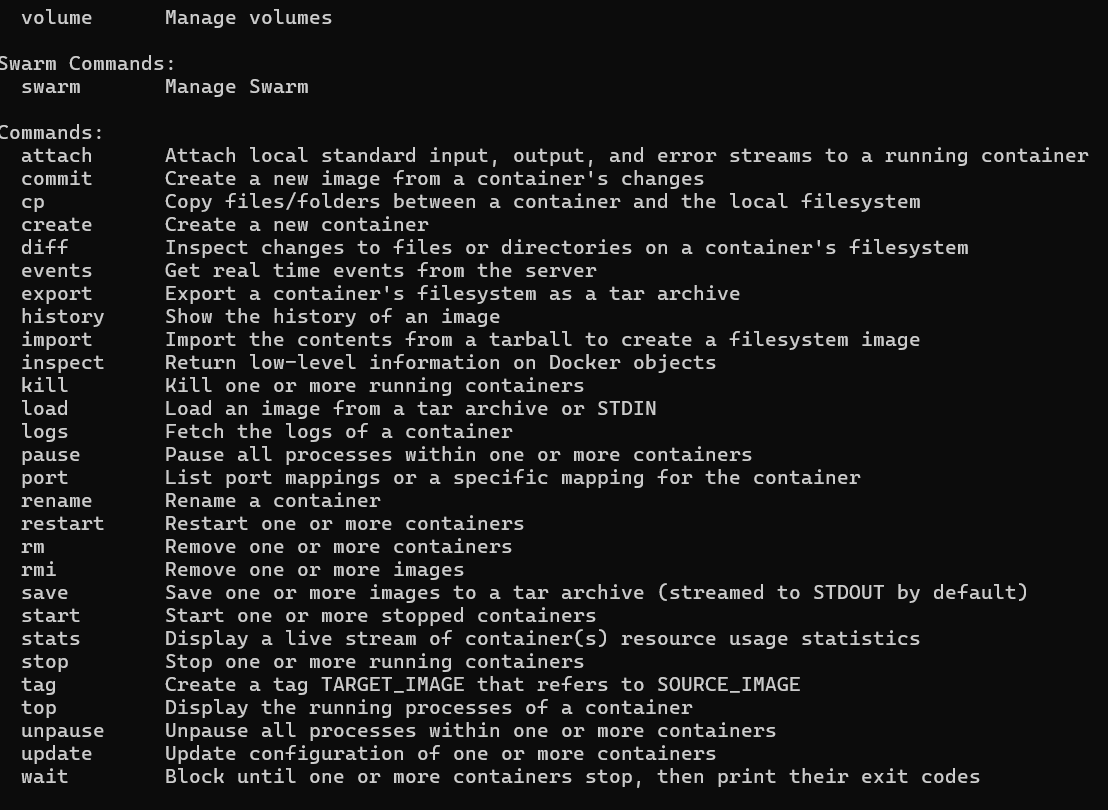
* Установите Docker
* Выполните базовую настройку

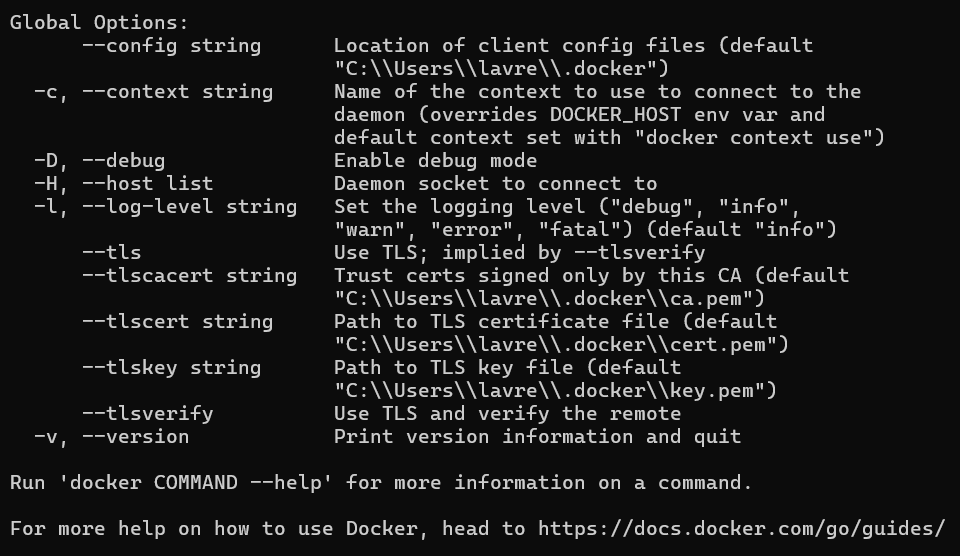


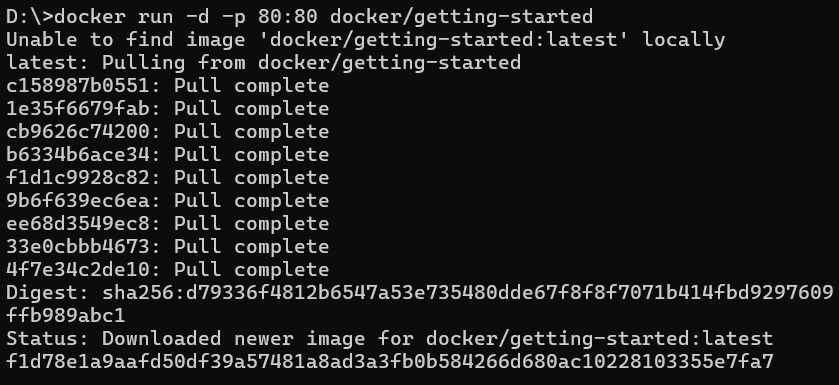
1. Изучите простейшие консольные команды и возможности Docker Desktop (см. лекцию), создать собственный контейнер docker/getting-started, открыть в браузере и изучить tutorial

Основные команды:

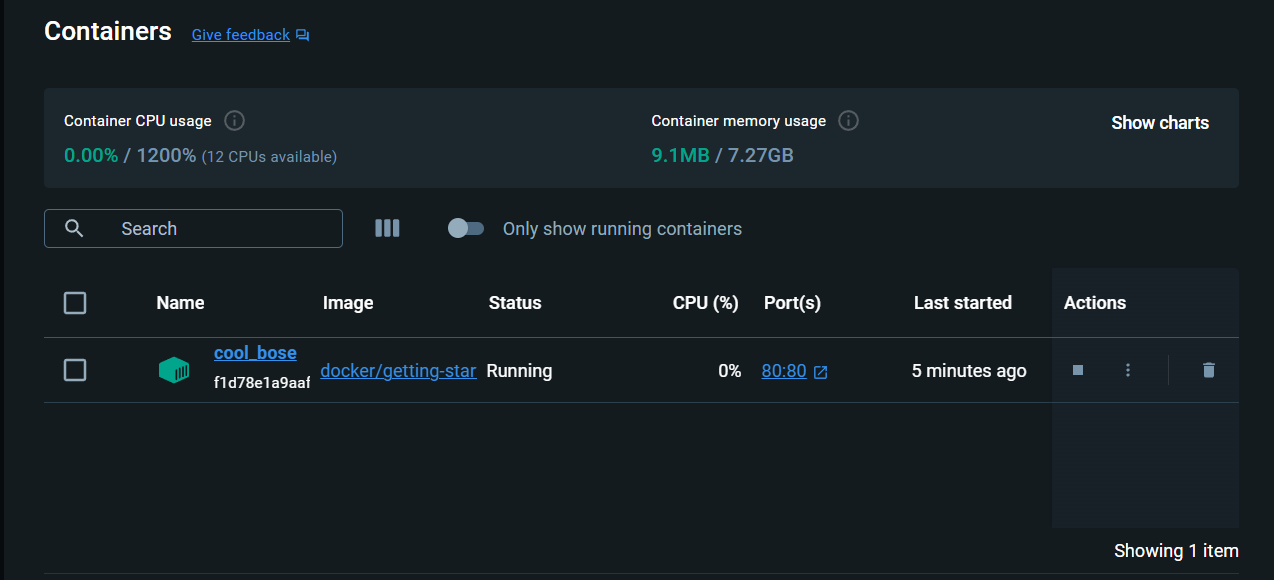
**

**

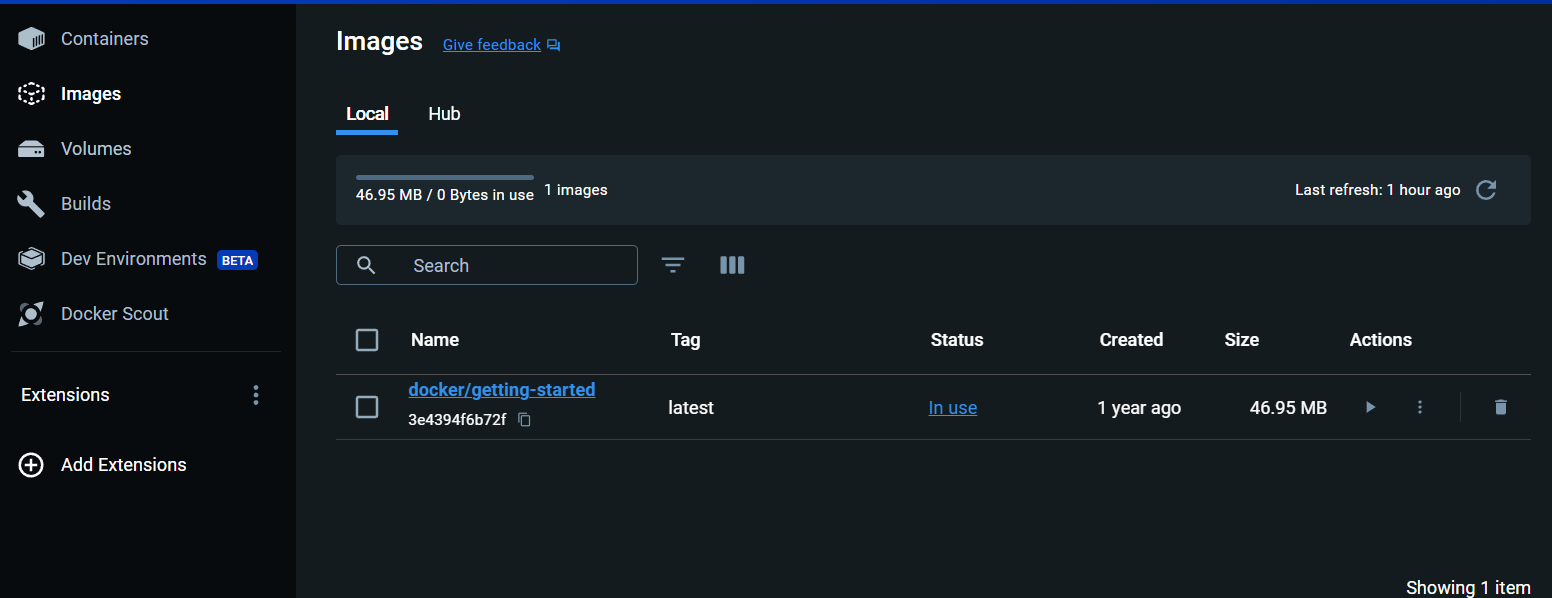
**

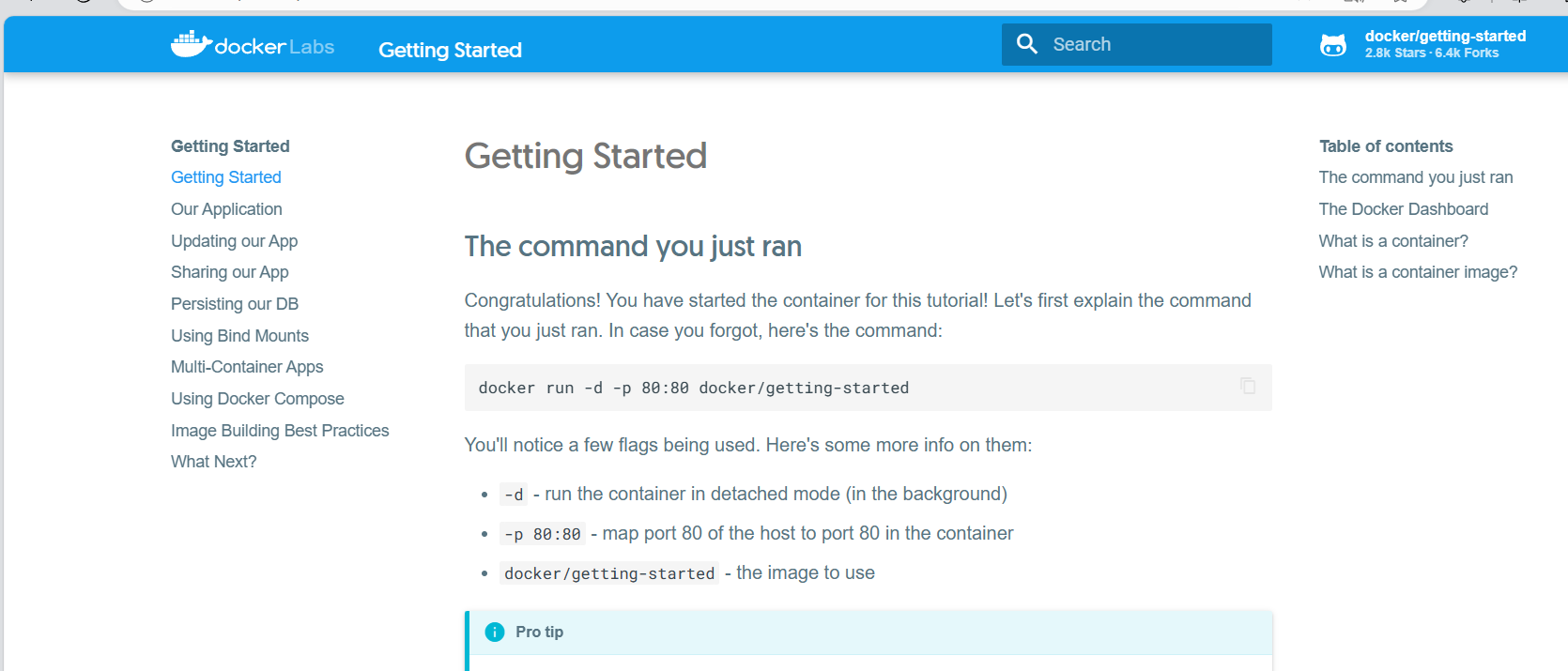
**

*Контейнер:*

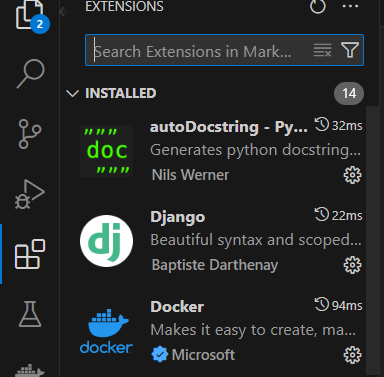
**

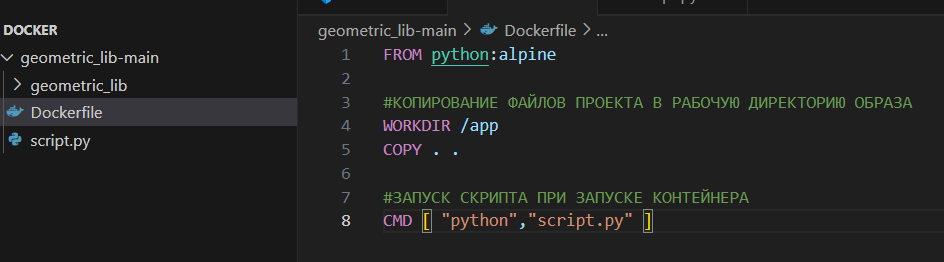
*Образ:*

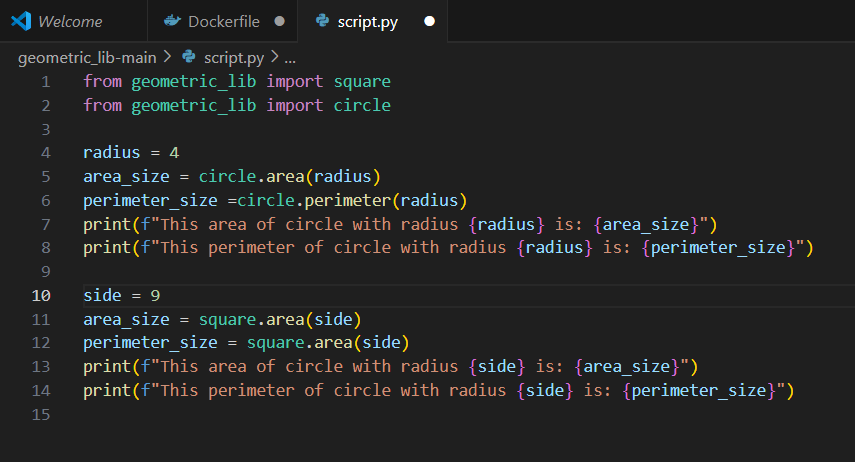
**

*localhost:80:* **

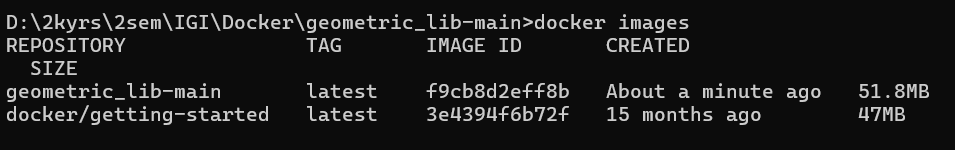
1. Создайте docker image, который запускает скрипт с использованием функций из https://github.com/smartiqaorg/geometric\_lib.
   1. Данные необходимые для работы скрипта передайте любым удобным способом (например: конфиг файл через docker volume, переменные окружения, перенаправление ввода). Изучите 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
   3. Использовать его для создания контейнера. Протестировать использование контейнера

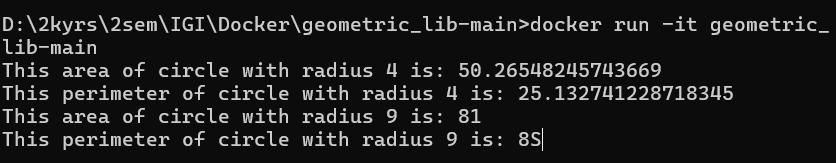


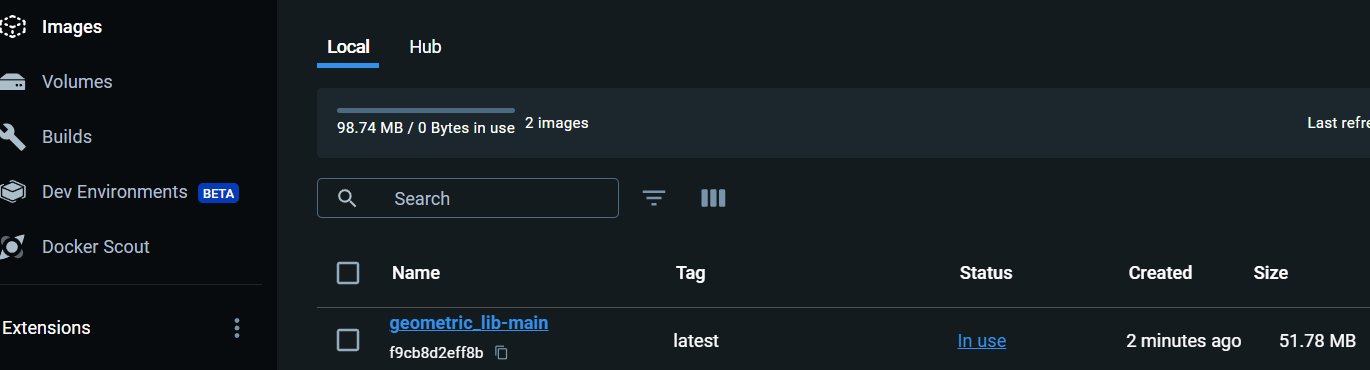




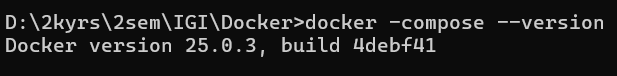


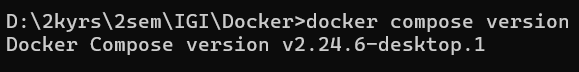




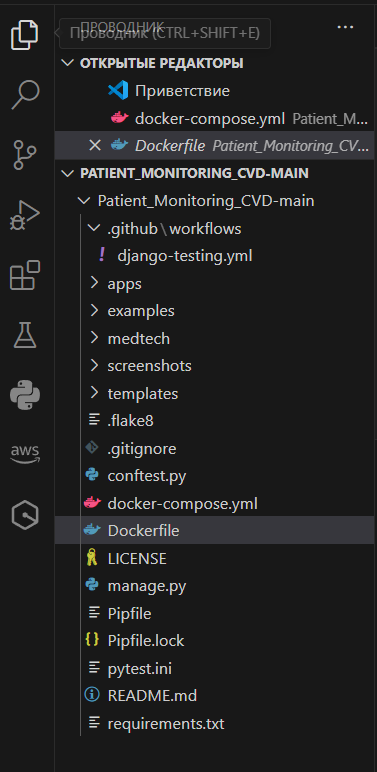


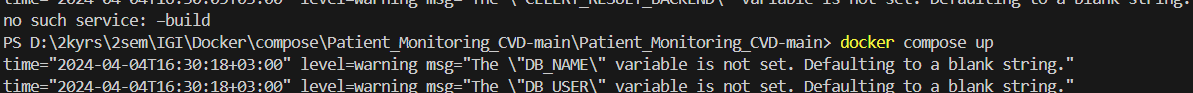
1. Скачать любой доступный проект с GitHub с произвольным стеком технологий или использовать свой, ранее разработанный. Создать для него необходимый контейнер, используя Docker Compose для управления многоконтейнерными приложениями. Запустить проект в контейнере.( Примеры Images: <https://hub.docker.com/_/phpmyadmin>,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

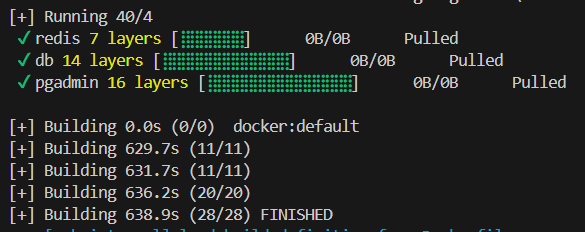


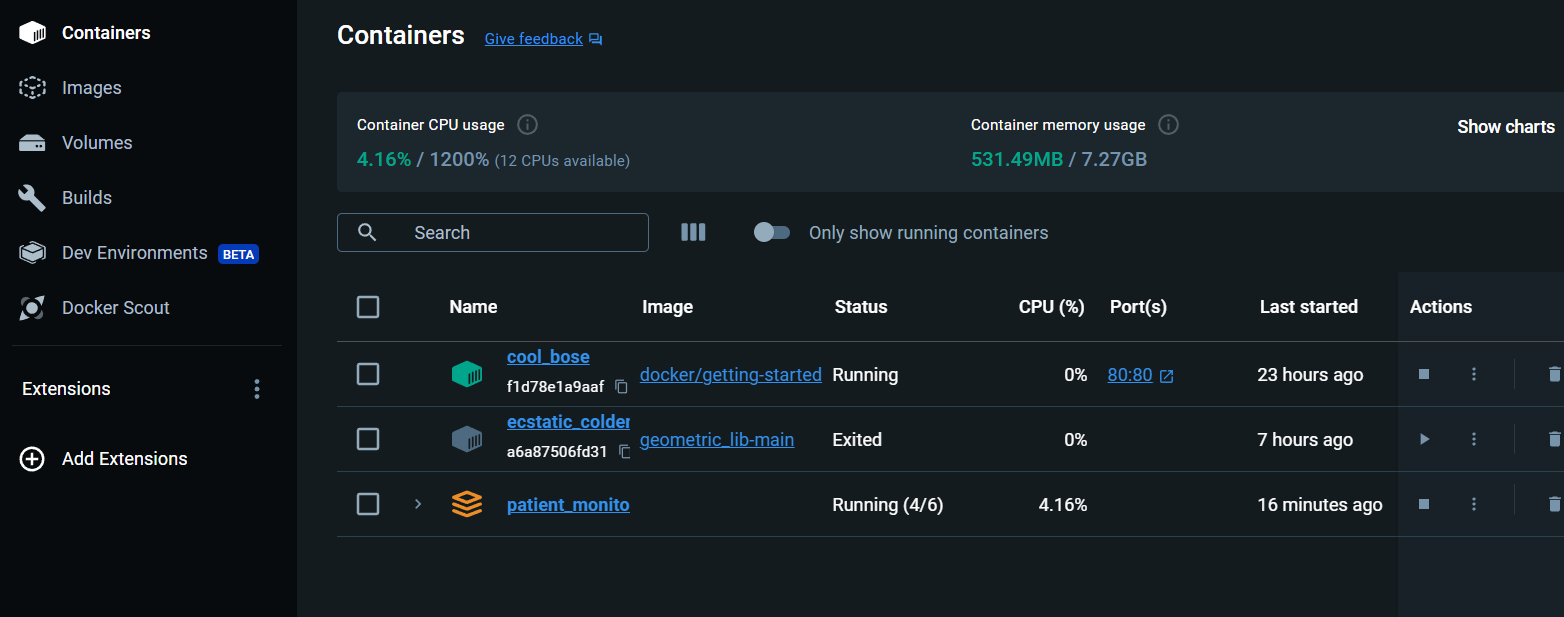


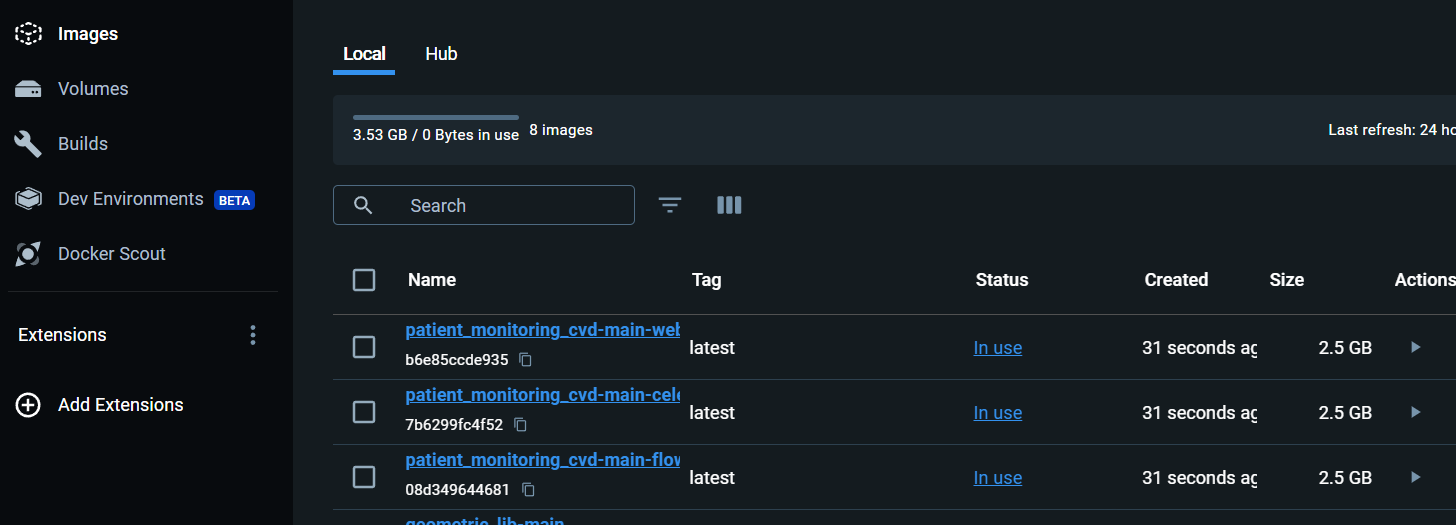
<https://github.com/Victorious-hub/Patient_Monitoring_CVD.git>

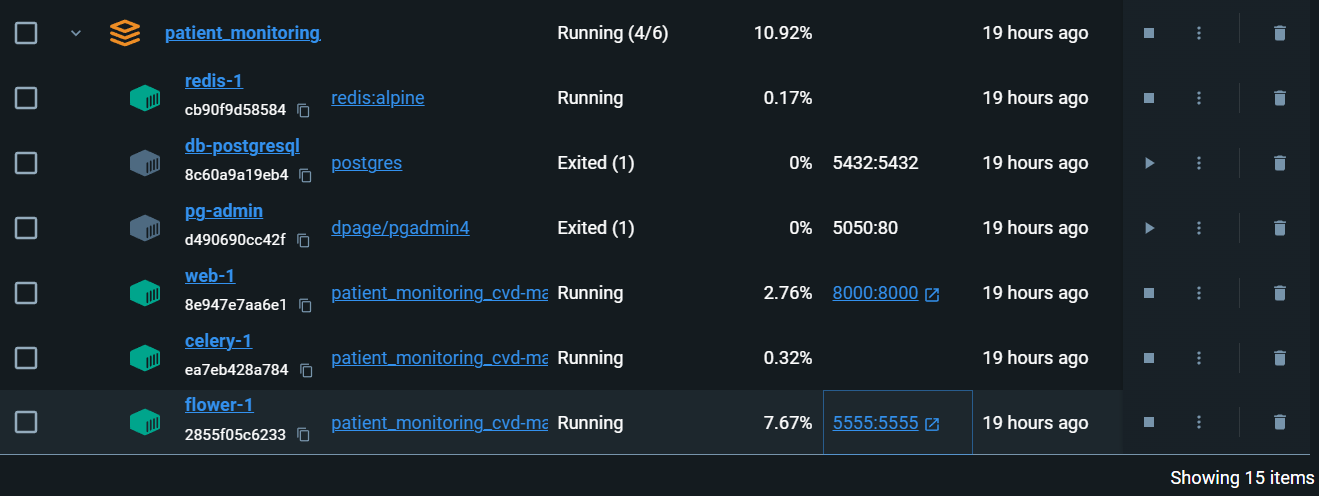


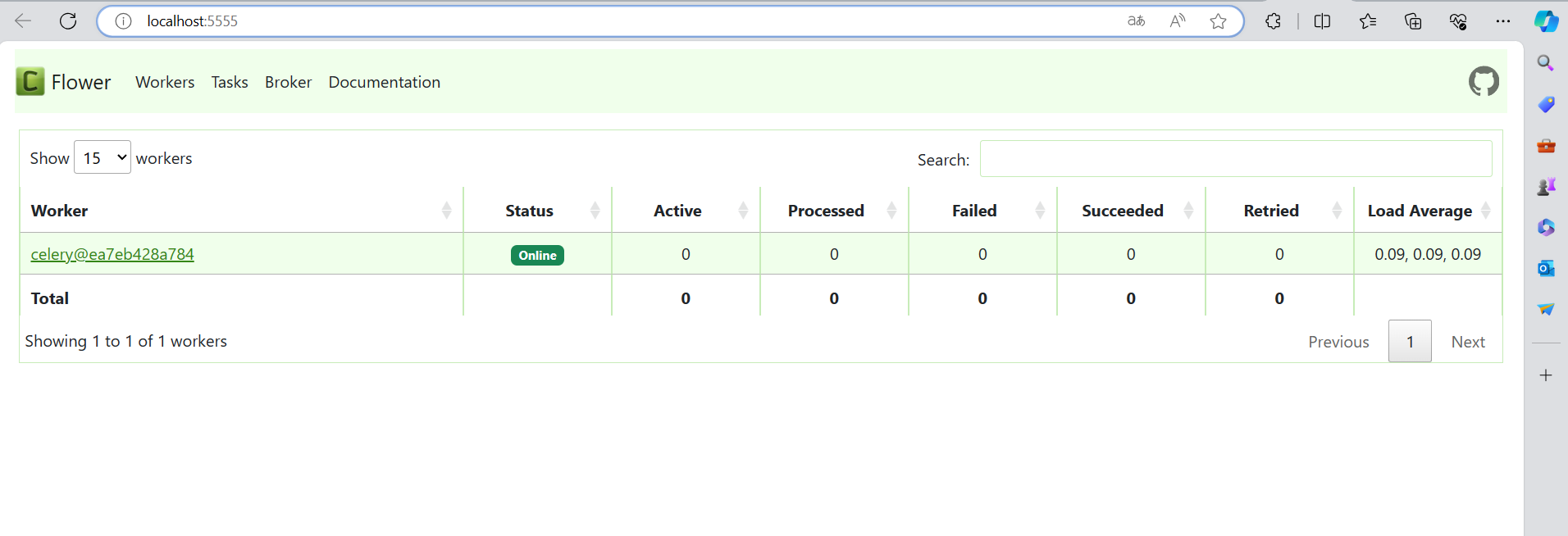






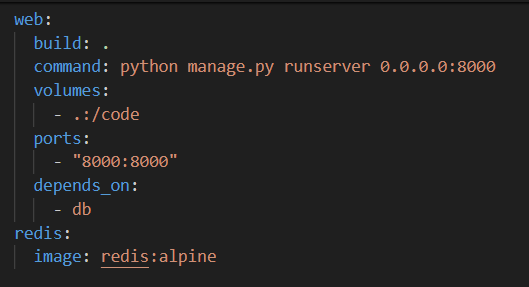




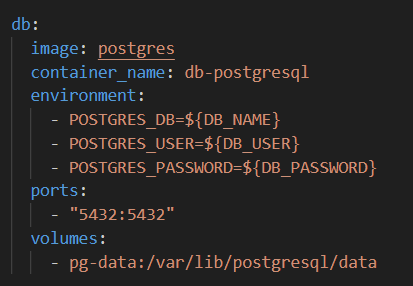


1. Настроить сети и тома для обеспечения связи между контейнерами и сохранения данных (исходные данные, логин, пароль и т.д.)

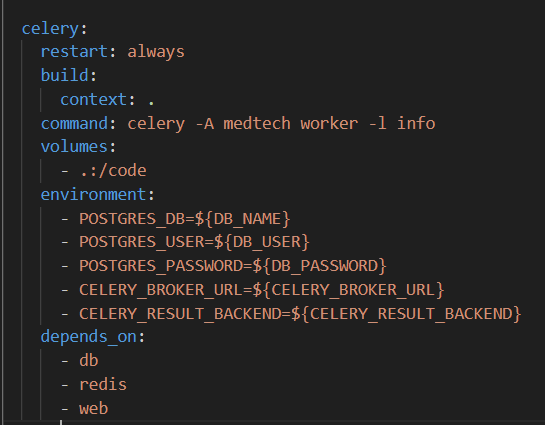
Контейнер web- том .../code



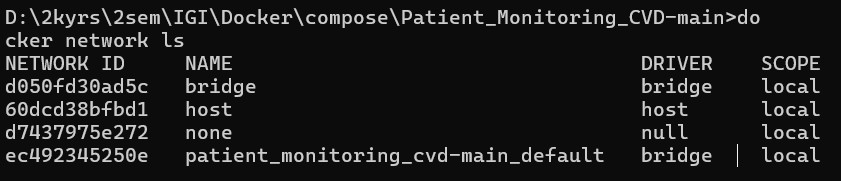
Контейнер db - volume pg-data (причём для базы данных мы указываем отдельный том в папке)



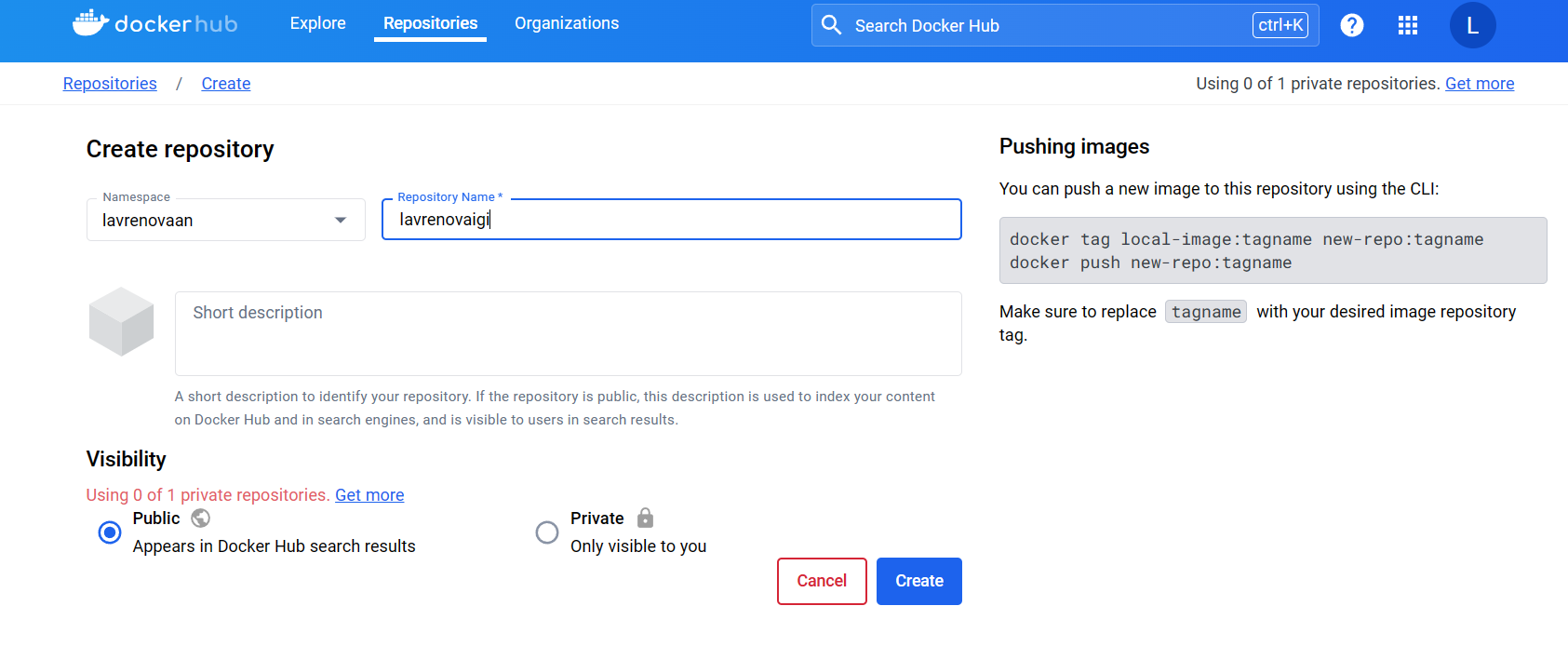
Контейнер celery – volume ../code

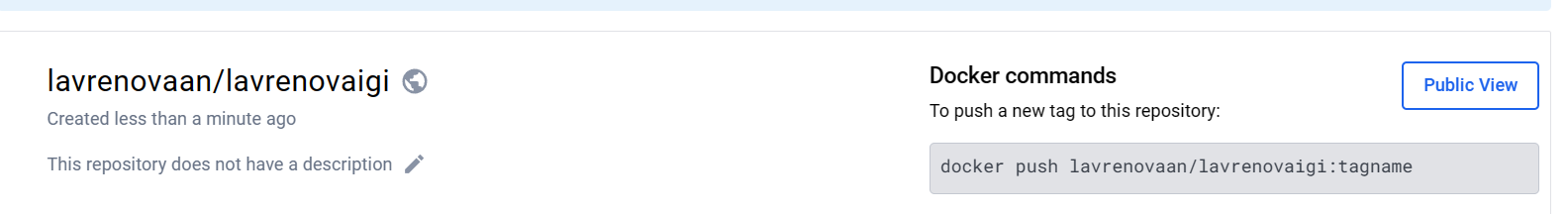


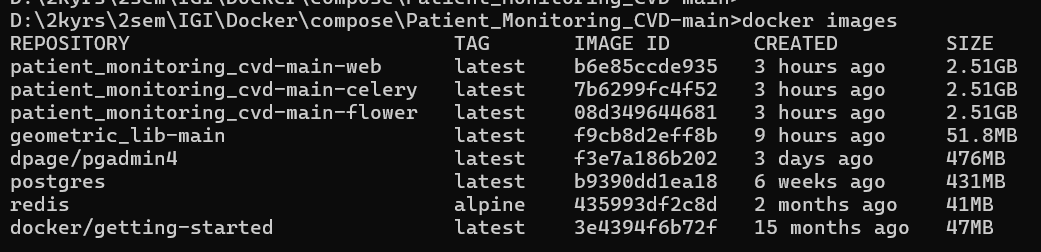
Создание Docker-compose.yml по умолчанию сеть для связи между контейнерами

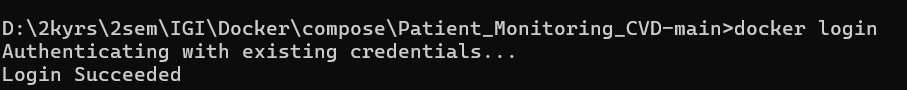


1. Разместите результат в созданный репозиторий в DockerHub

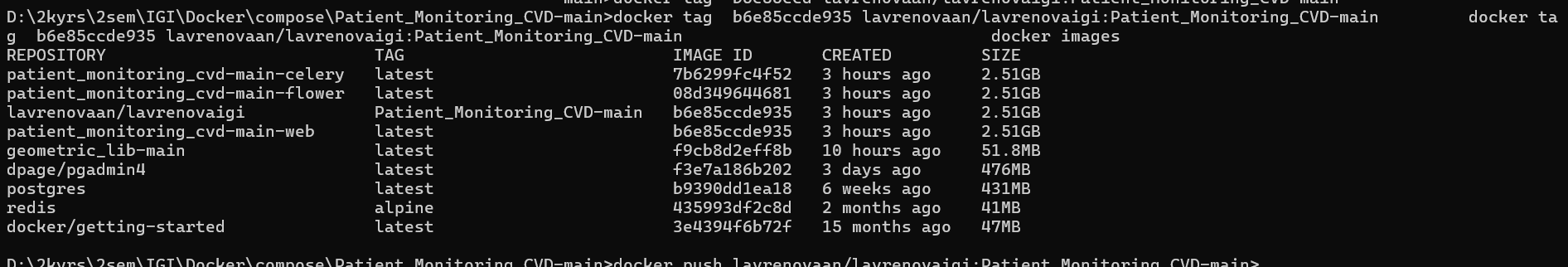


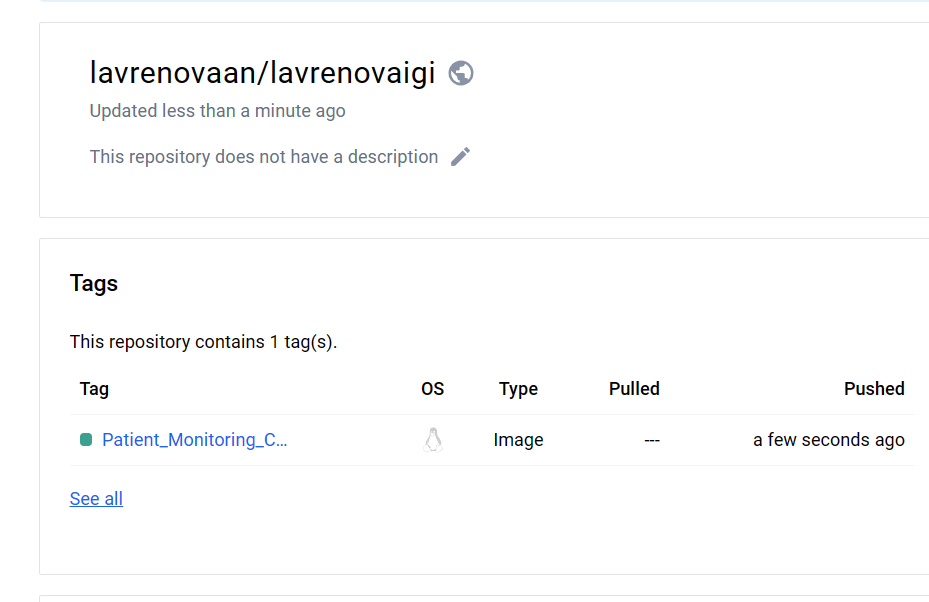






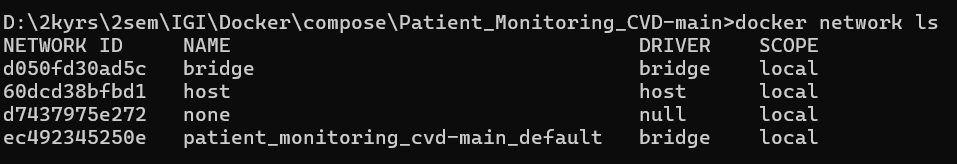


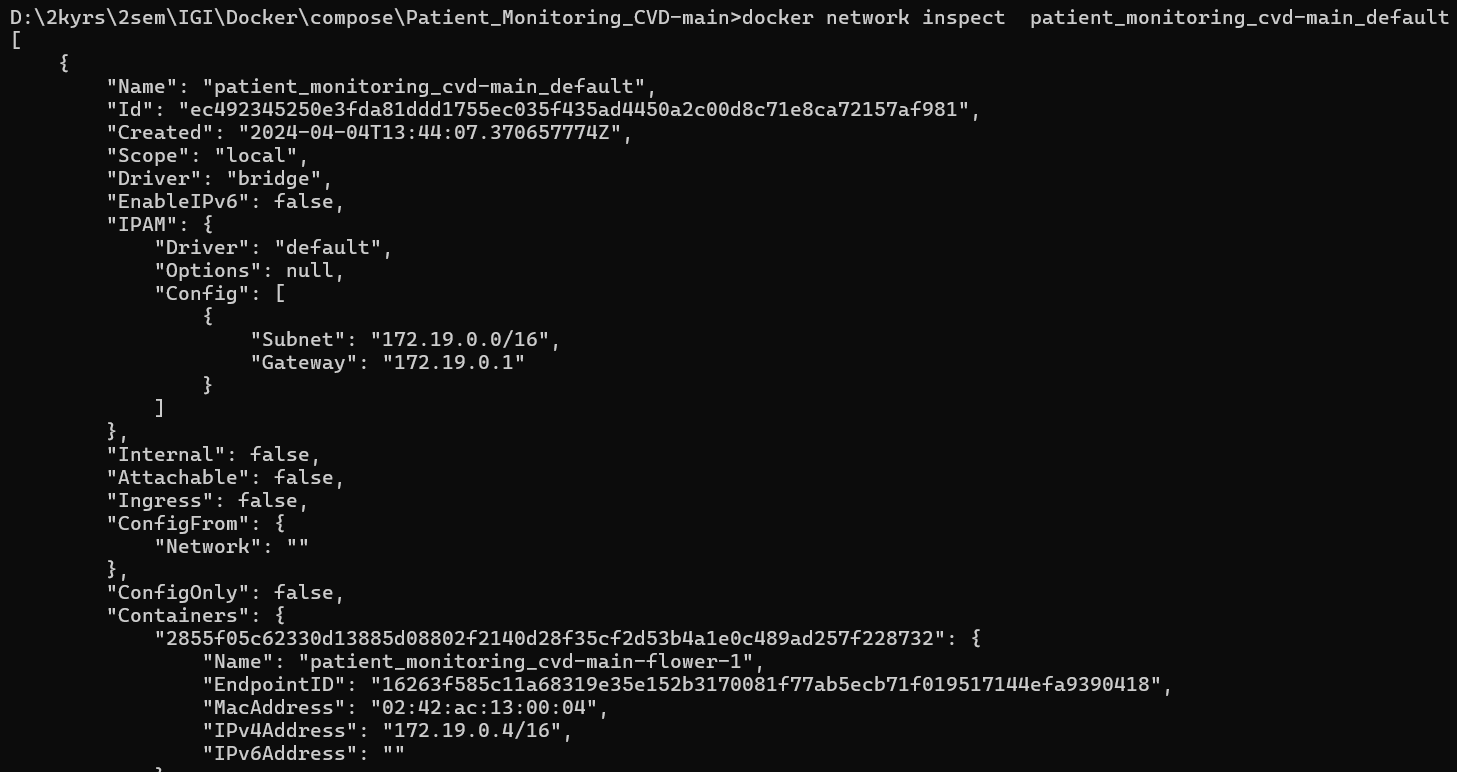




1. Выполните следующие действия с целью изучить особенности сетевого взаимодействия:

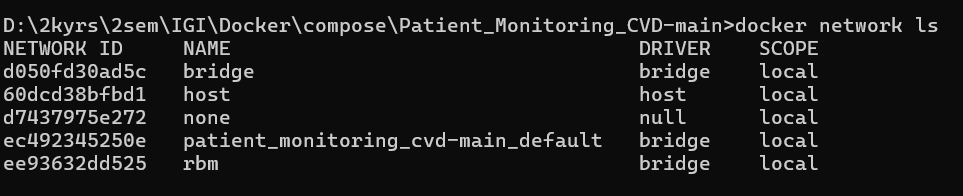
* Получить информацию о всех сетях, работающих на текущем хосте и подробности о каждом типе сети





* 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

Просмотр списка сетей:

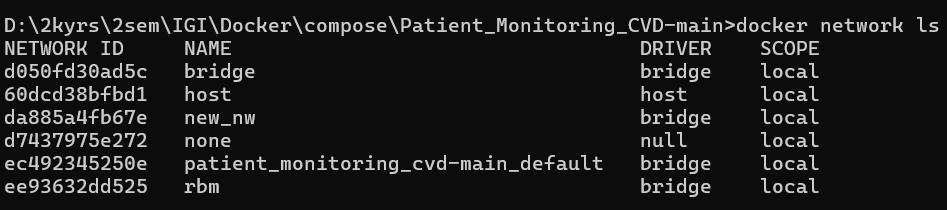


Сначала мы подключаем контейнер к мосту по умолчанию, который является драйвером, предоставляющим службу DNS по умолчанию для подключенных к нему контейнеров, выполнив команду. Выход : Ctrl + p, затем Ctrl + q

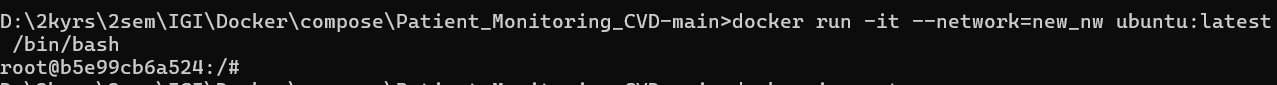


Теперь на этом шаге мы собираемся создать нашу собственную сеть, а затем на следующем шаге подключить контейнер к этой сети.

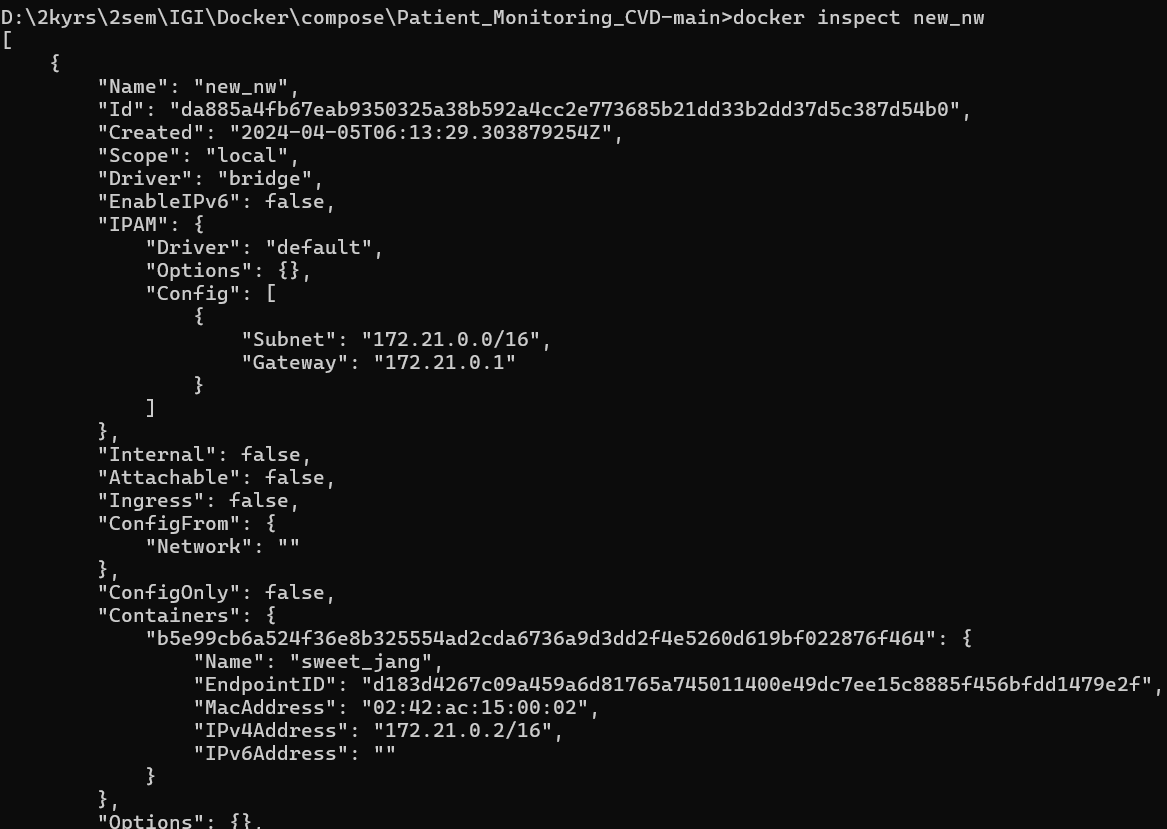




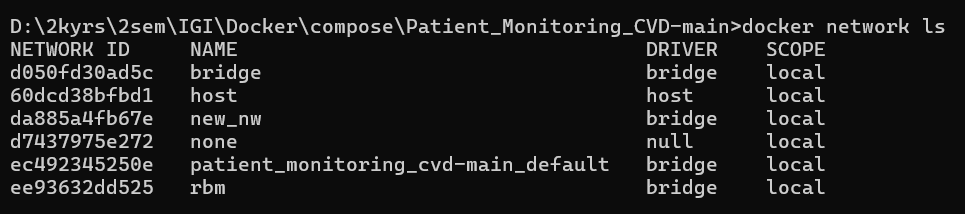
На этом шаге мы подключим контейнер к нашей сети, который мы создали на предыдущем шаге.

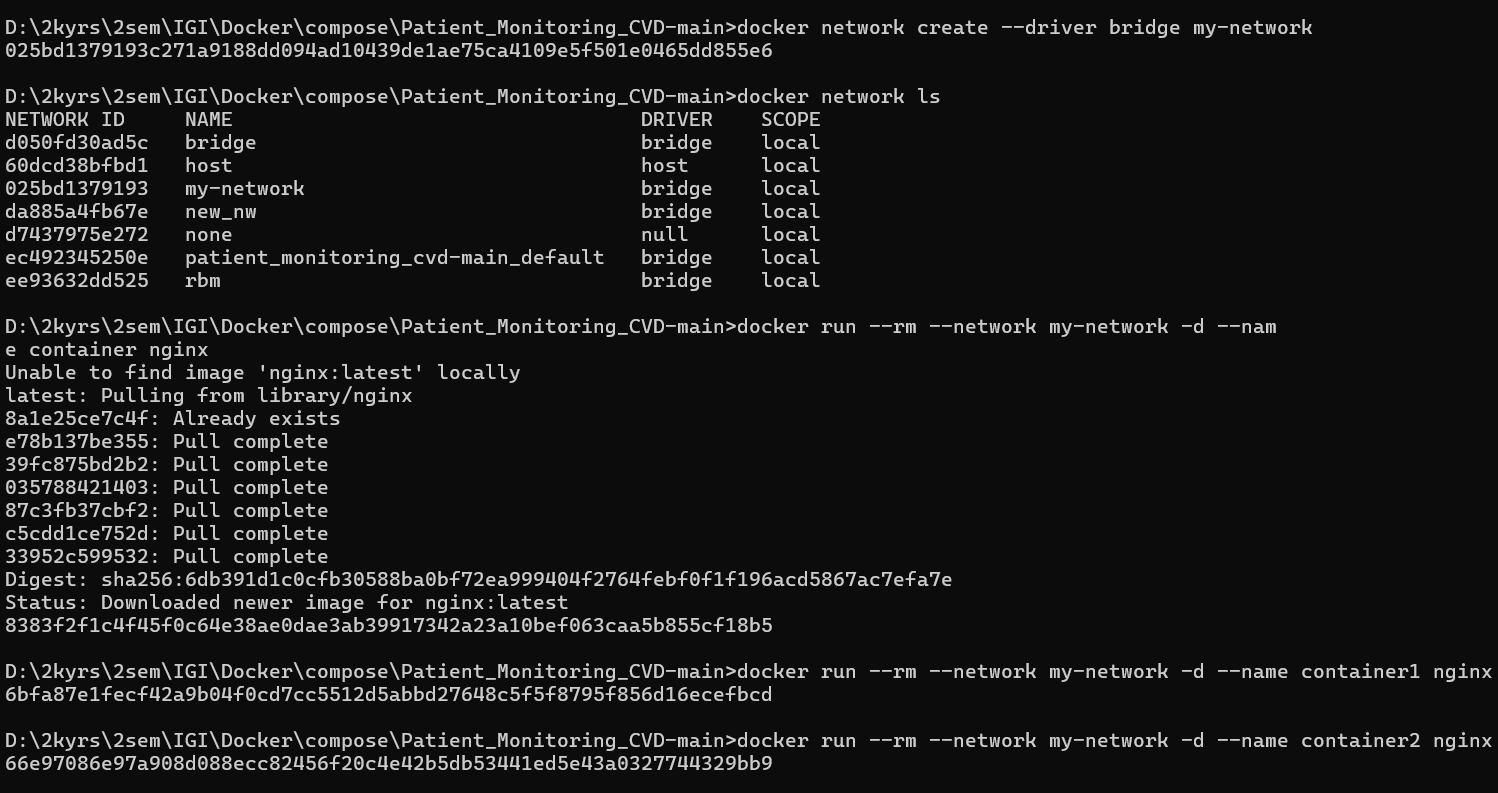


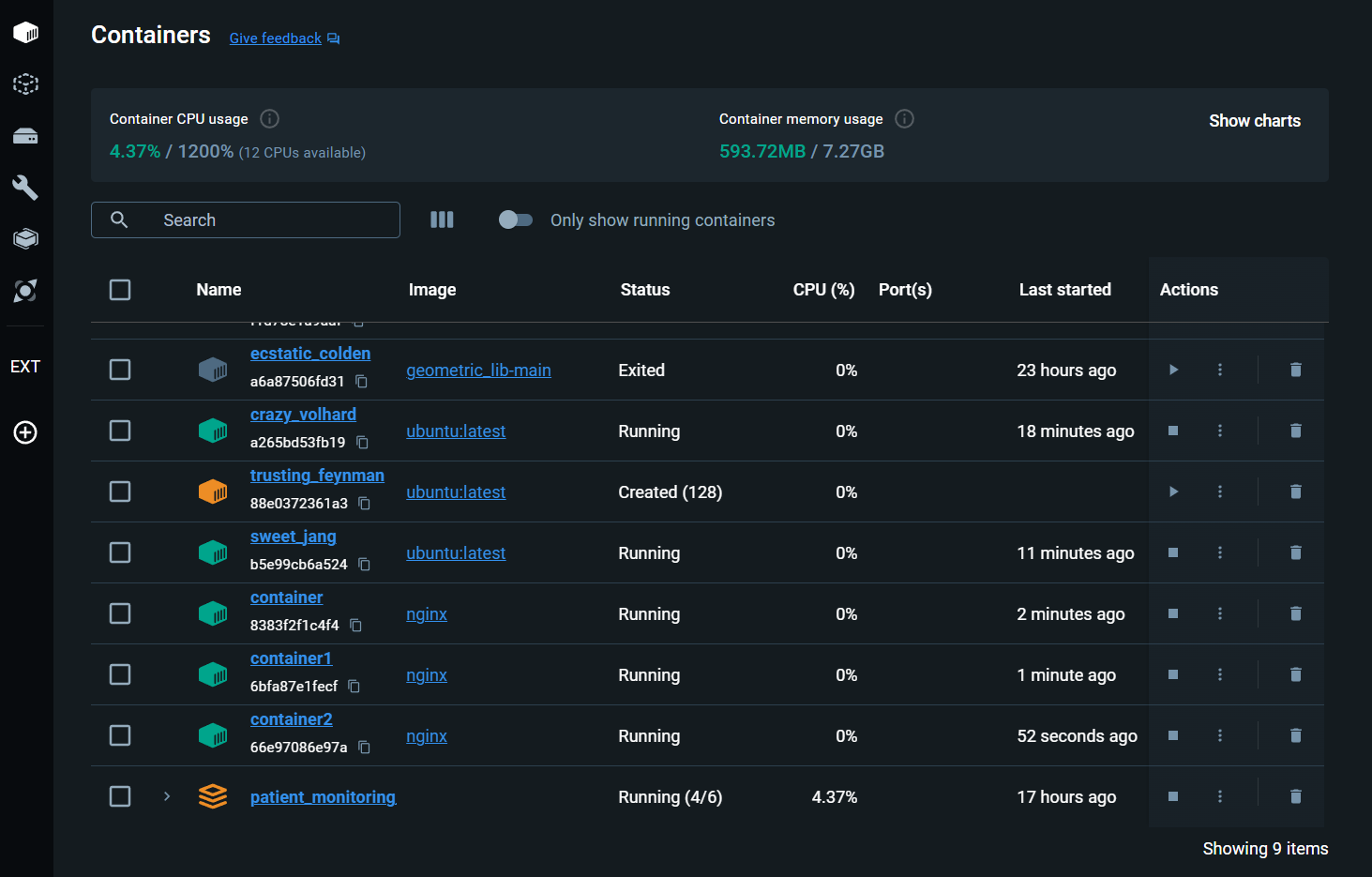
Теперь проверьте созданную нами сеть. Там упоминается контейнер, который мы подключаем к уважаемой сети.

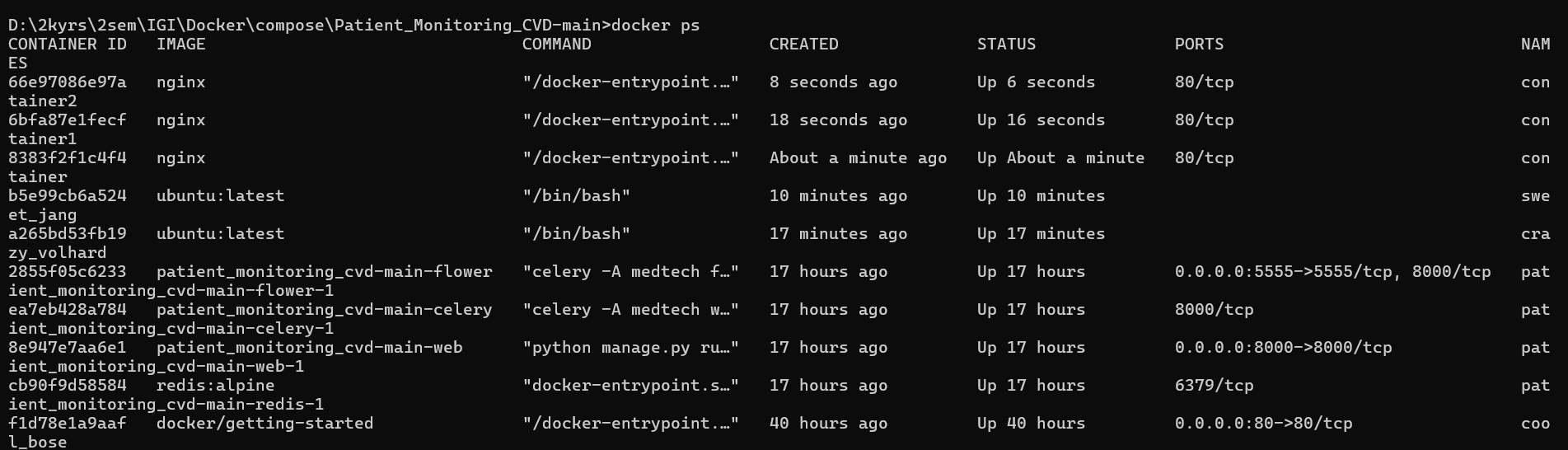


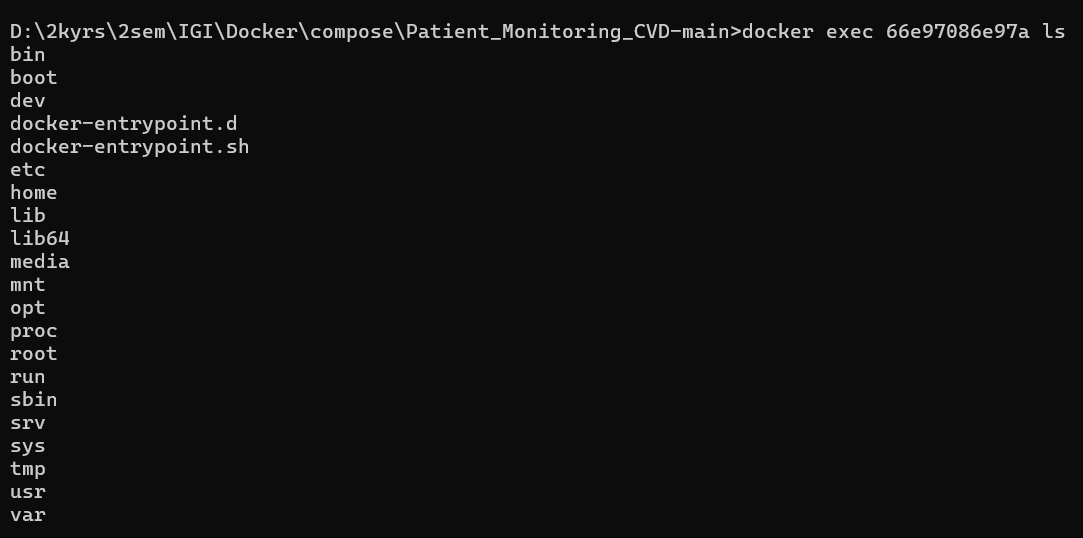
* Создать еще одну сеть bridge,   вывести о ней всю информацию, запустить в ней три контейнера, подключиться к любому из контейнеров и пропинговать два других из оболочки контейнера, убедиться, что между контейнерами происходит общение по IP-адресу

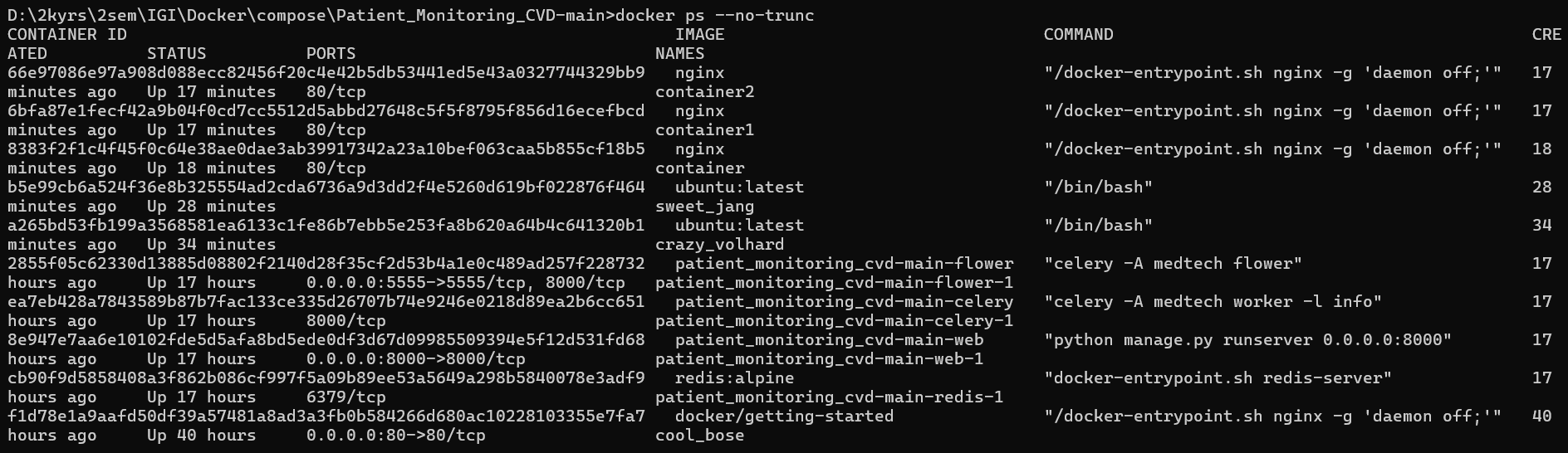






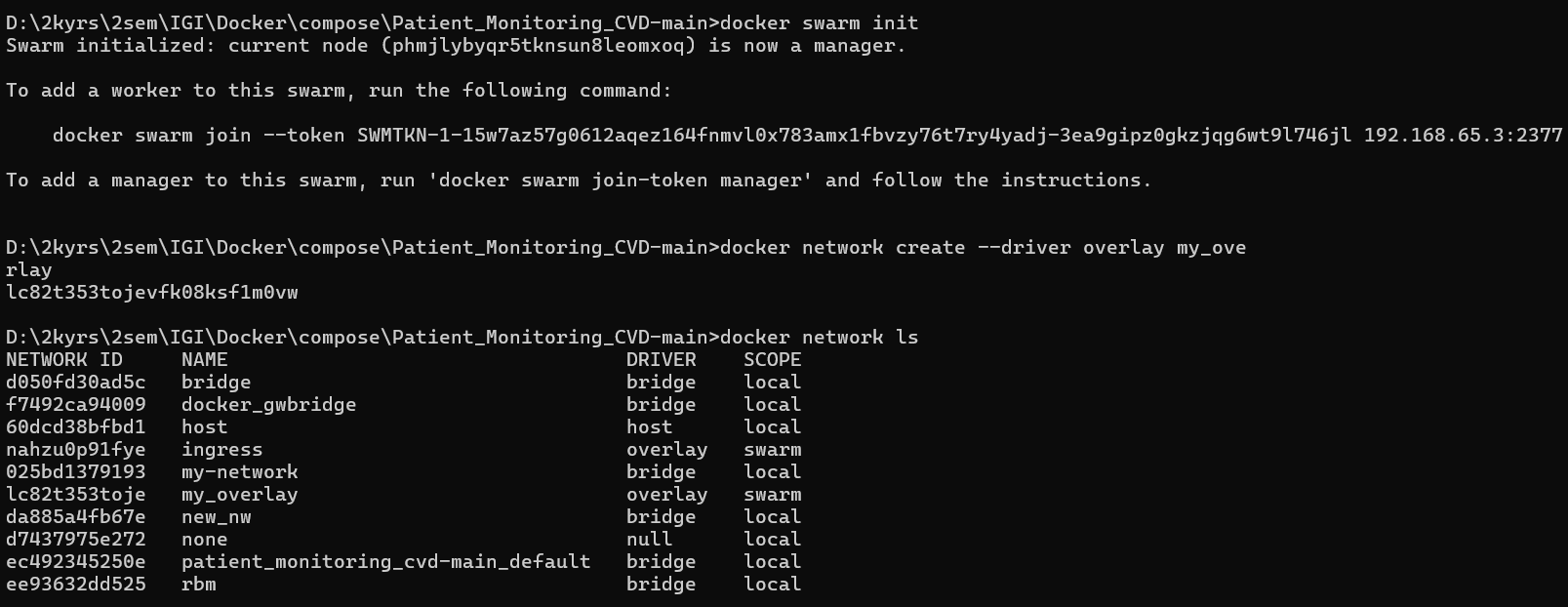


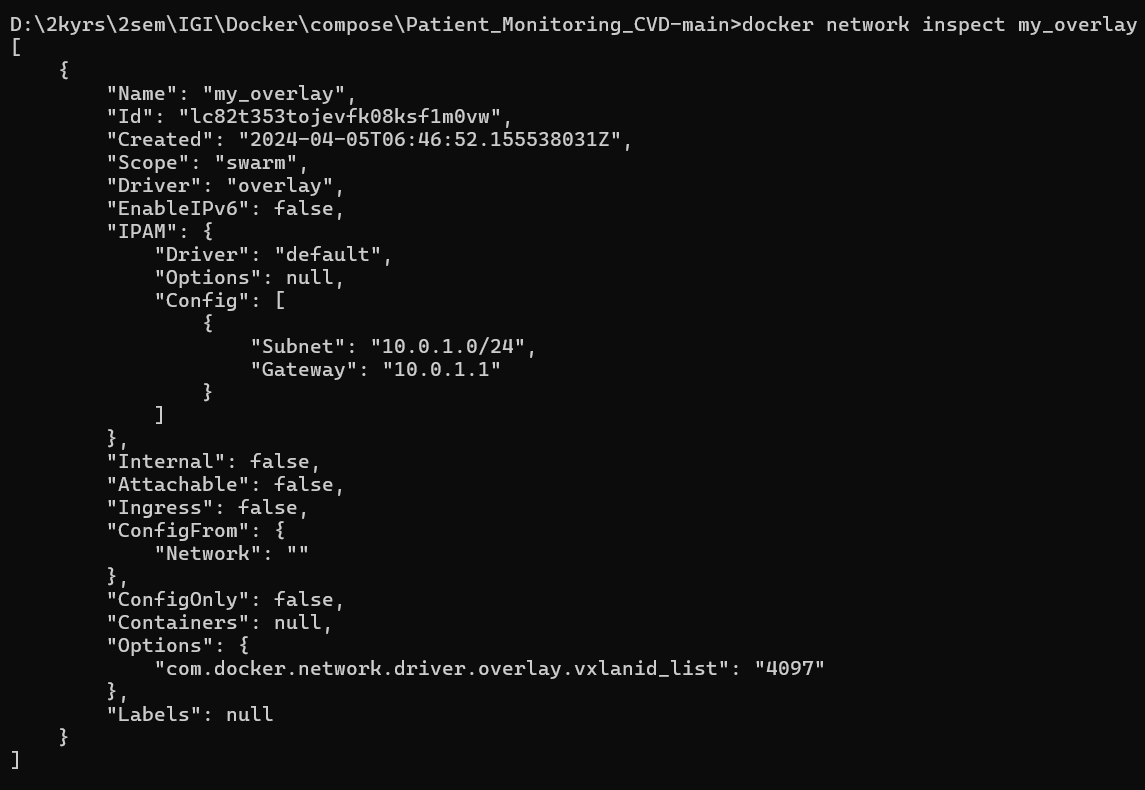




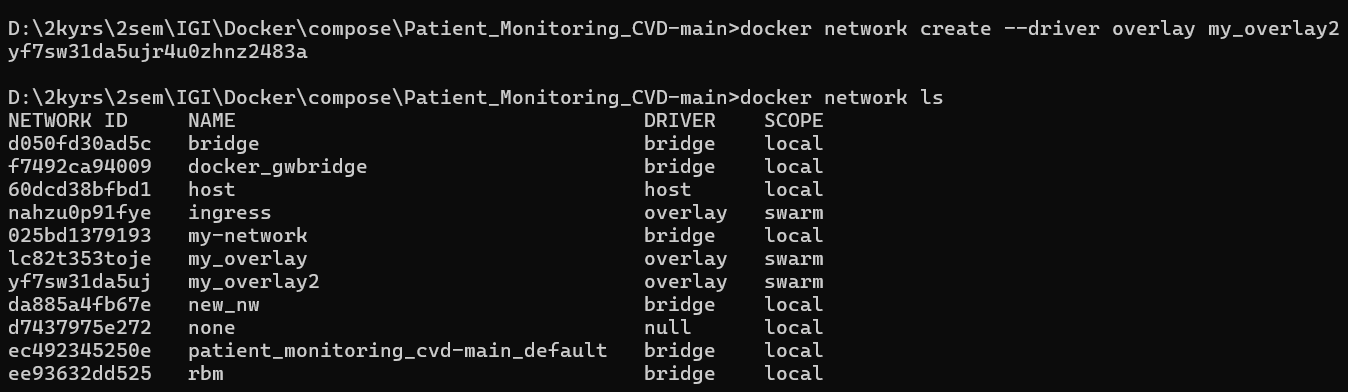


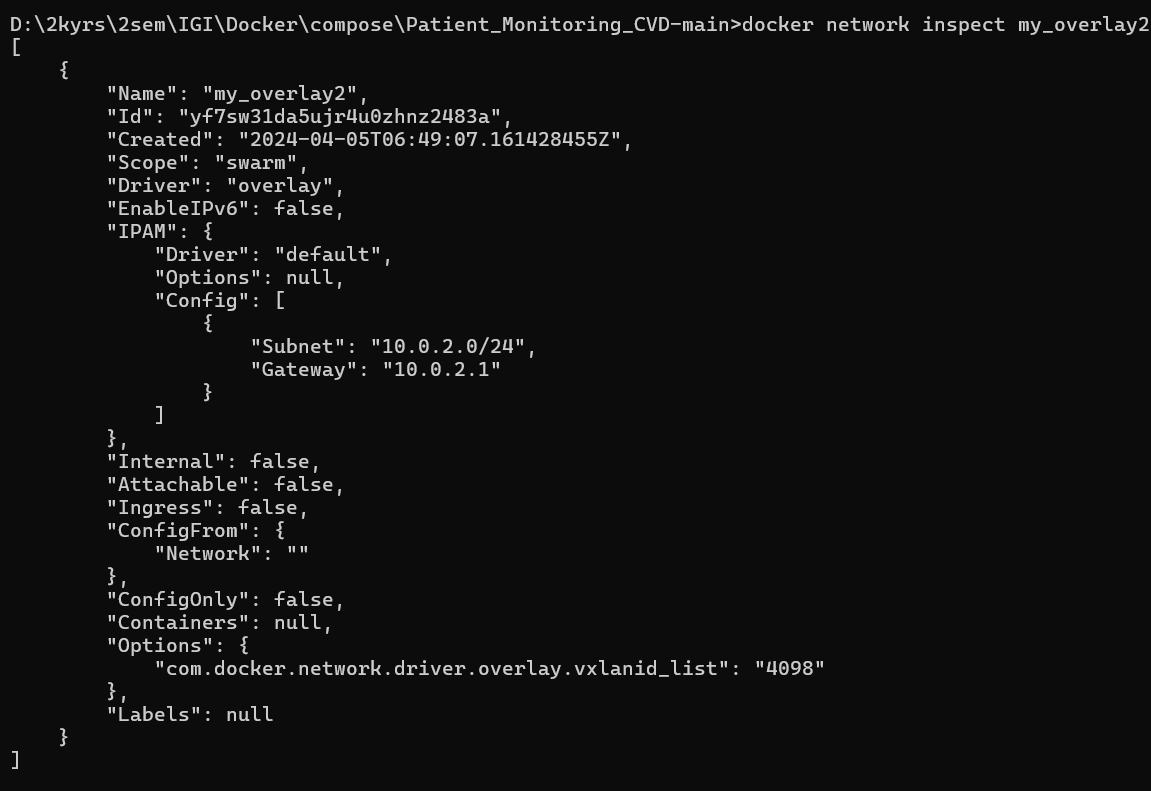
* Создать свою собственную сеть overlay, проверить, создана ли она, вывести о ней всю информацию

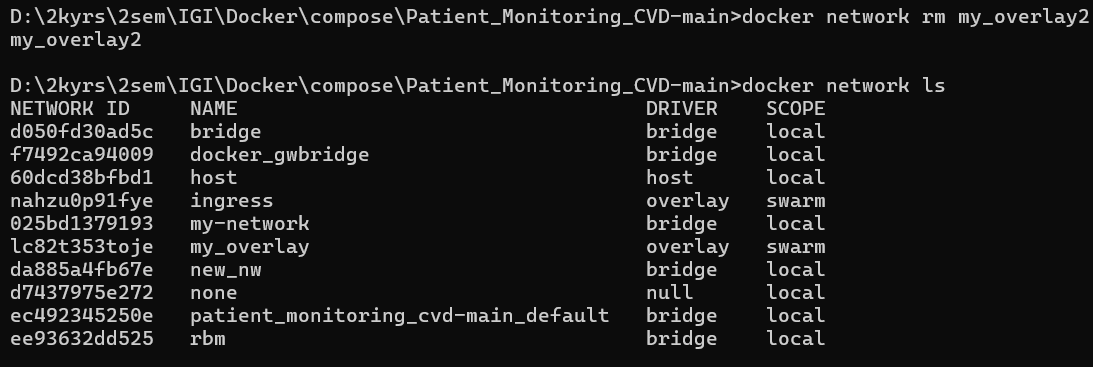




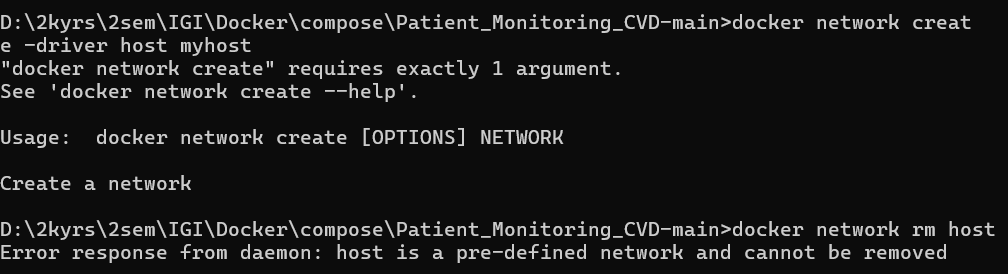
* Создать еще одну сеть overlay, проверить, создана ли она, вывести о ней всю информацию, удалить сеть







* Попробовать создать сеть host, сохранить результат  в отчет.



Какие сети доступны по умолчанию в Docker?

По умолчанию есть:

* bridge, сеть, к которой подключаются контейнеры, если не указано иначе.
* none, сетевой стек без наличия сетевого интерфейса в контейнере.
* host, подключение к сетевому стеку сервера.

Что такое образ Docker, что делает команда Docker run?

Образ Docker — это набор файлов, соединенный с настройками, с помощью которого можно создать экземпляры, которые запускаются в отдельных контейнерах в виде изолированных процессов. Образ строится с использованием инструкций для получения полной исполняемой версии приложения, зависящей от версии ядра сервера. Команда Docker run используется для создания таких экземпляров, называемых контейнерами, запускаемыми с использованием образов Docker. При запуске одного образа пользователь может создать несколько контейнеров.

Поясните разницу между docker run и docker create

Главная разница между этими командами заключается в том, что вторая создаст контейнер в остановленном состоянии. Также docker create может быть использована для хранения и вывода идентификатора контейнера для будущего использования. Лучше всего это делать с помощью docker run, добавляя --cidfile FILENAME, поскольку повторный запуска не перезаписывает файл.