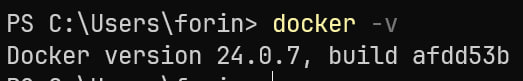
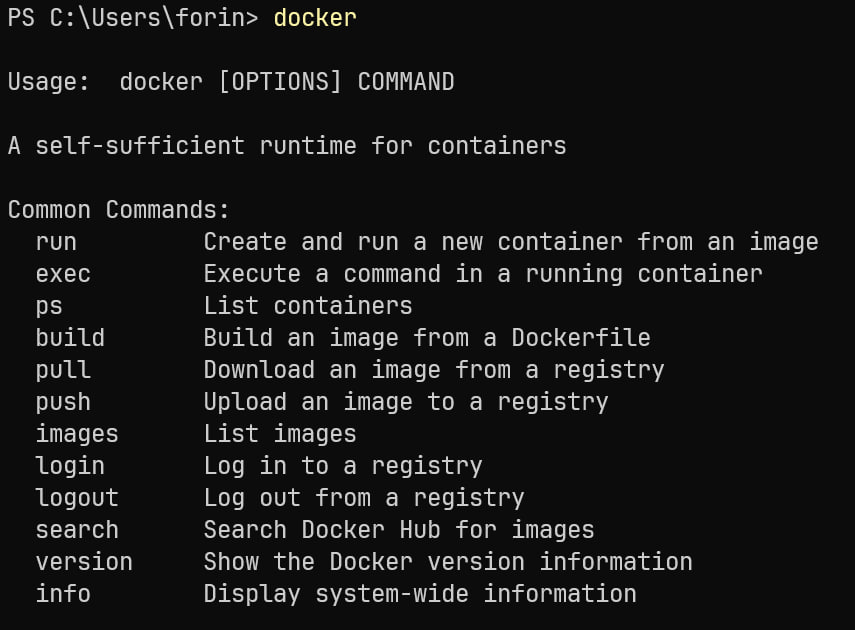
**Шишов Е. П. из 253505 с Лабораторная работа №2**

**и Вариант 27**

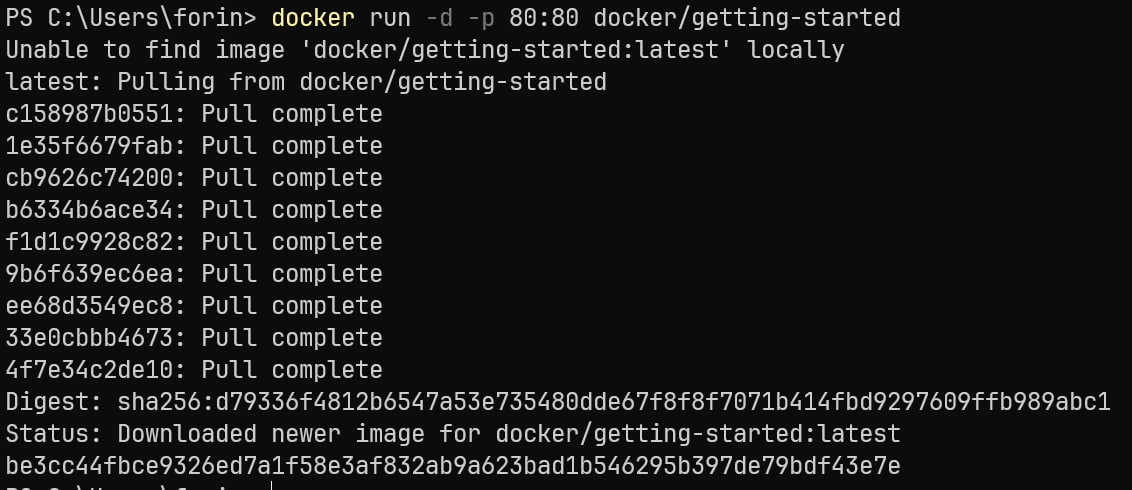
1.Подготовьте рабочее окружение в соответствии с типом вашей операционной системы

* + Установите Docker
  + 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

запуск контейнера



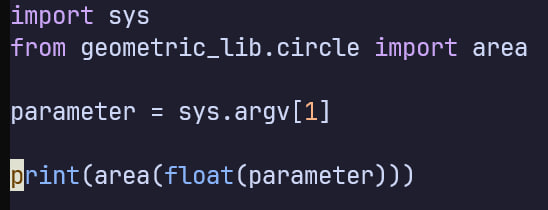
вывод запущенных контейнеров с указанием их размеров



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

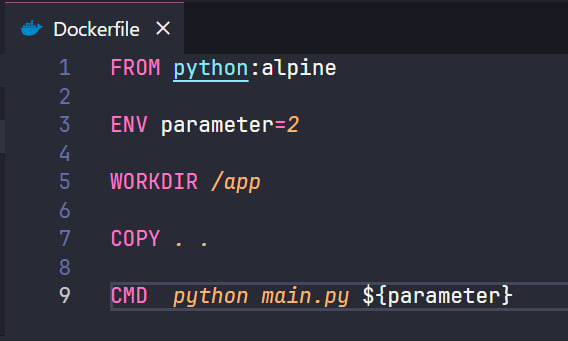
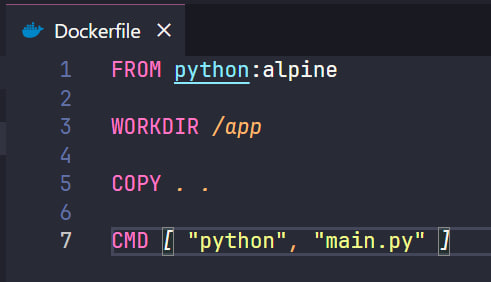
Код скрипта:



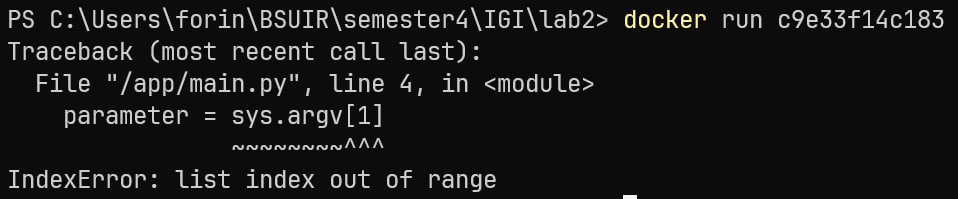
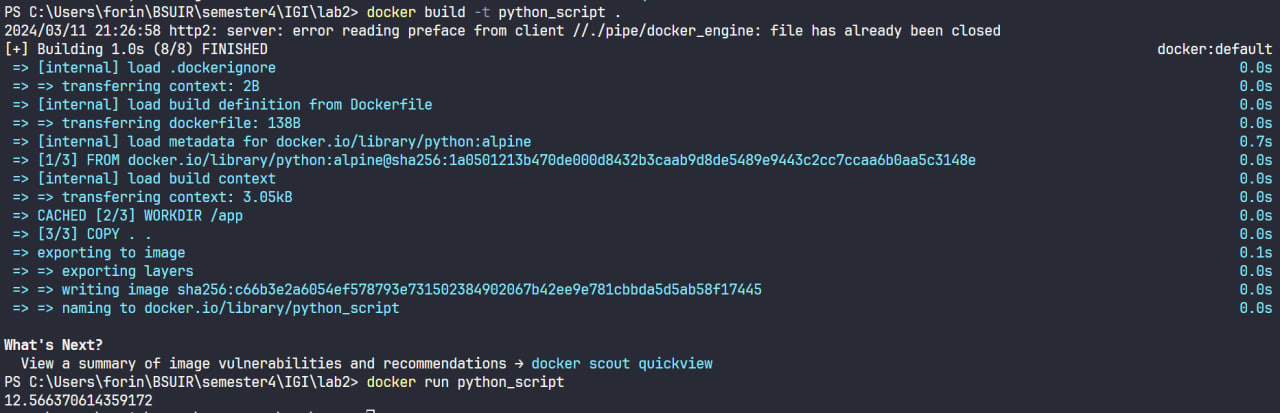
* 1. Создать Dockerfile для реализации сборки собственных Docker образов

Использование переменных среды Использование томов

Dockerfile:

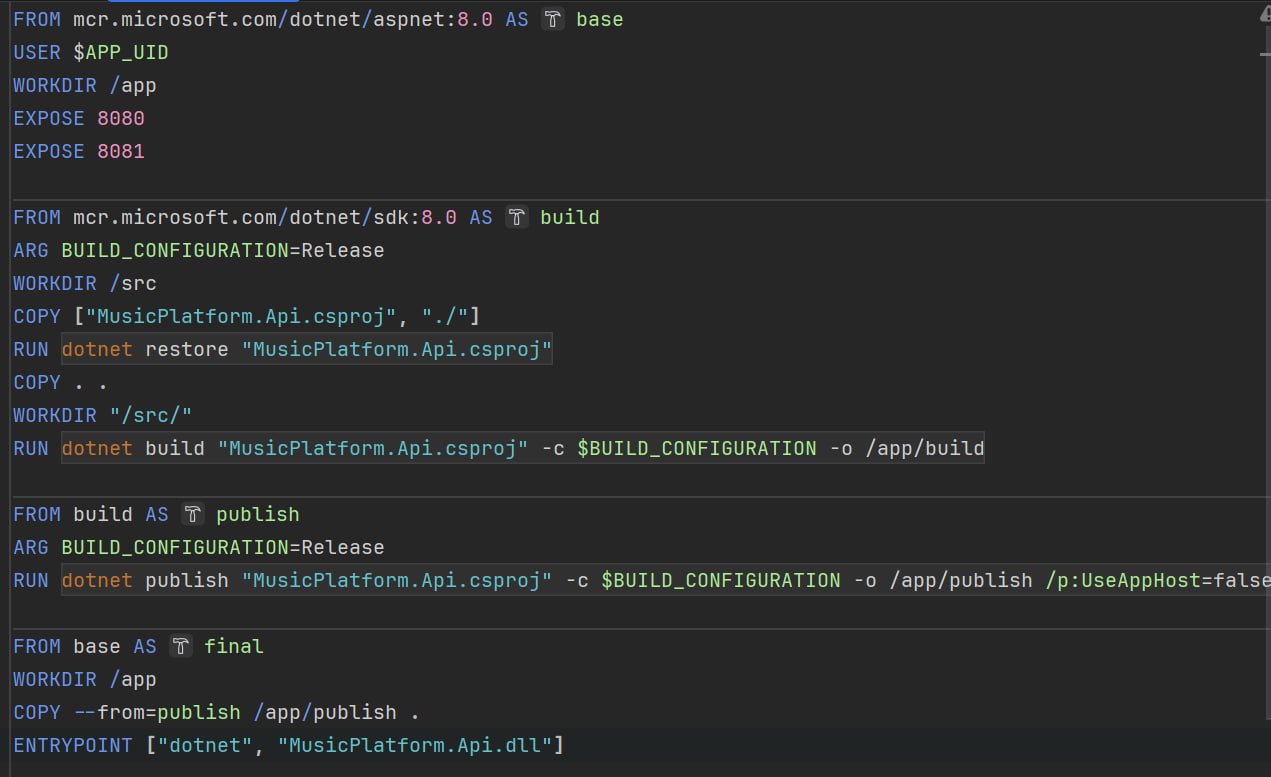


* 1. Использовать его для создания контейнера. Протестировать использование контейнера

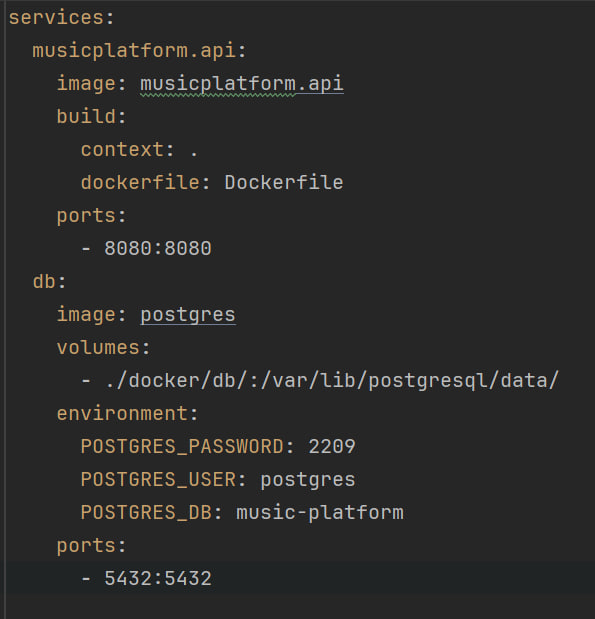


4. Скачать любой доступный проект с GitHub с произвольным стеком технологий или использовать свой, ранее разработанный. Создать для него необходимый контейнер, используя Docker Compose для управления многоконтейнерными приложениями. Запустить проект в контейнере.( Примеры Images: <https://hub.docker.com/_/phpmyadmin>, <https://hub.docker.com/_/mysql>, https://hub.docker.com/\_/postgres) Настроить сети и тома для обеспечения связи между контейнерами и сохранения данных (исходные данные, логин, пароль и т.д.)

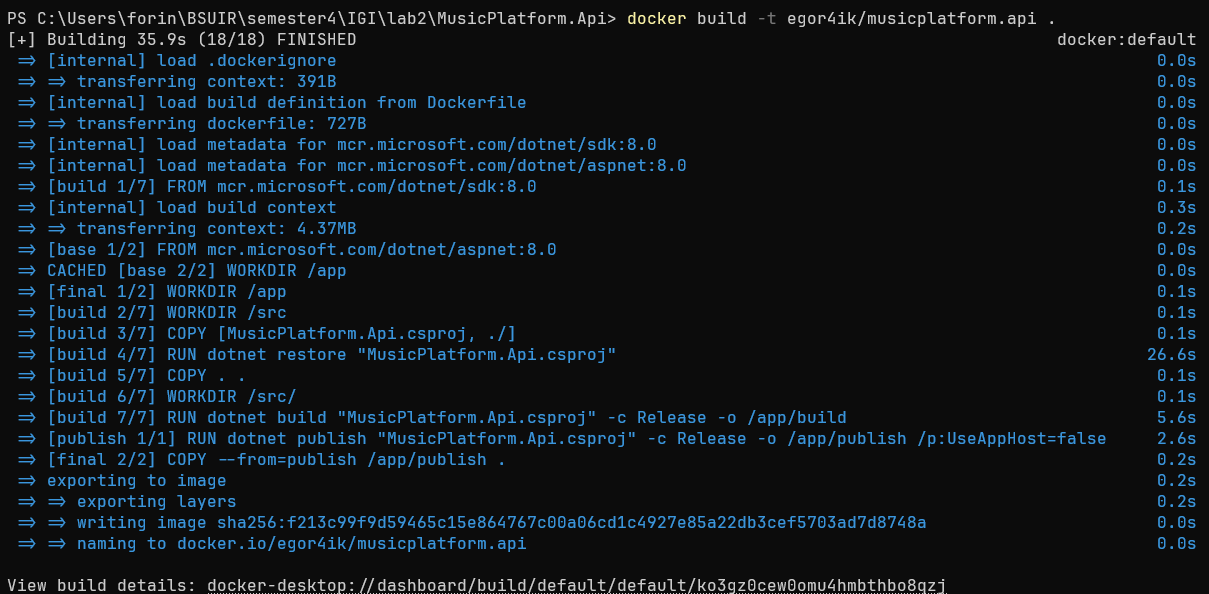
Dockerfile:

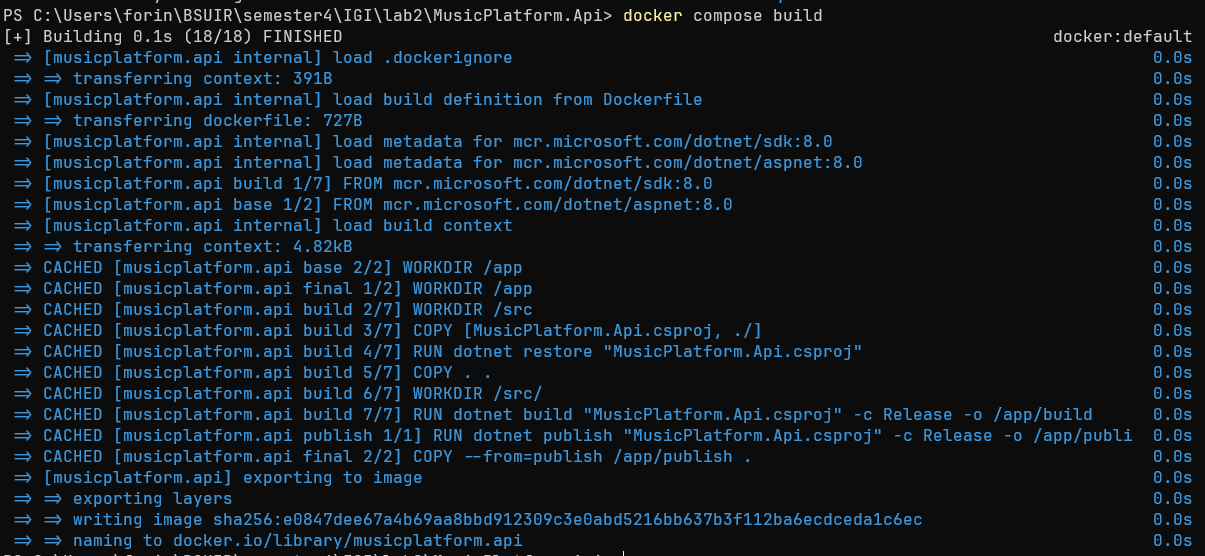


Docker Compose:

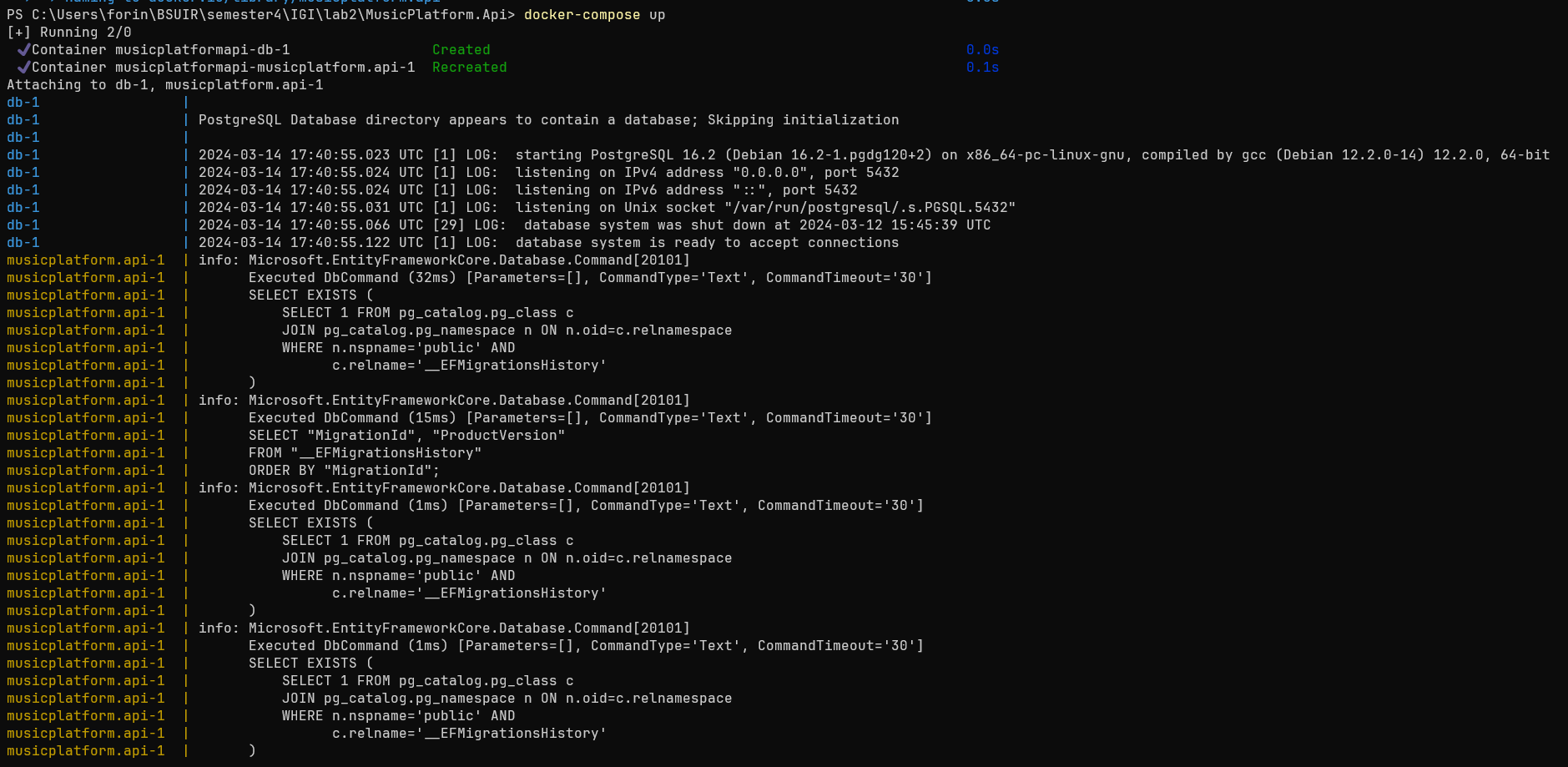


Построение:

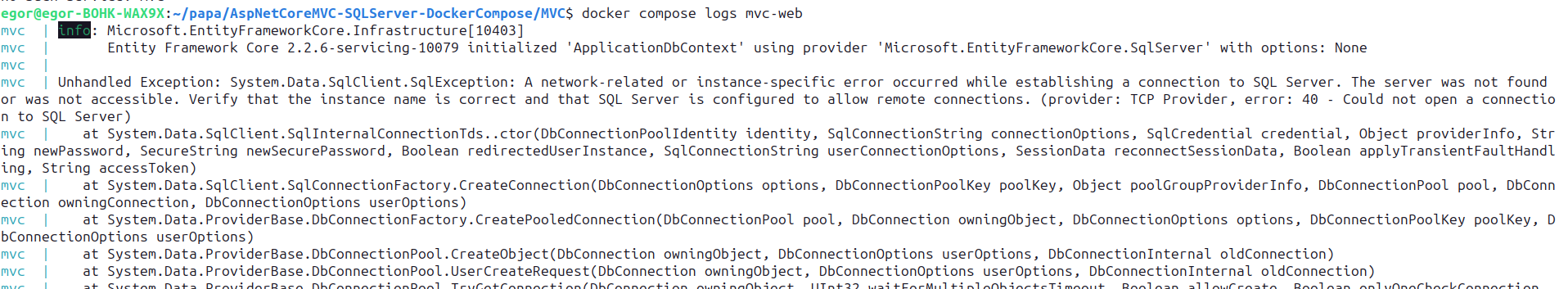




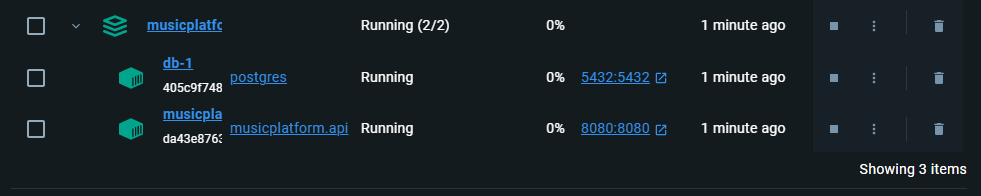
Запуск:

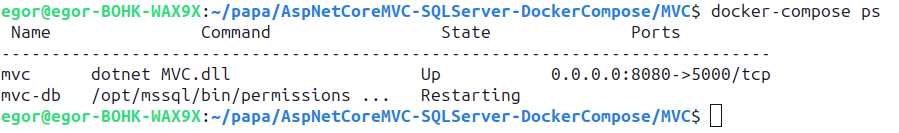


Просмотр логов отдельного сервиса



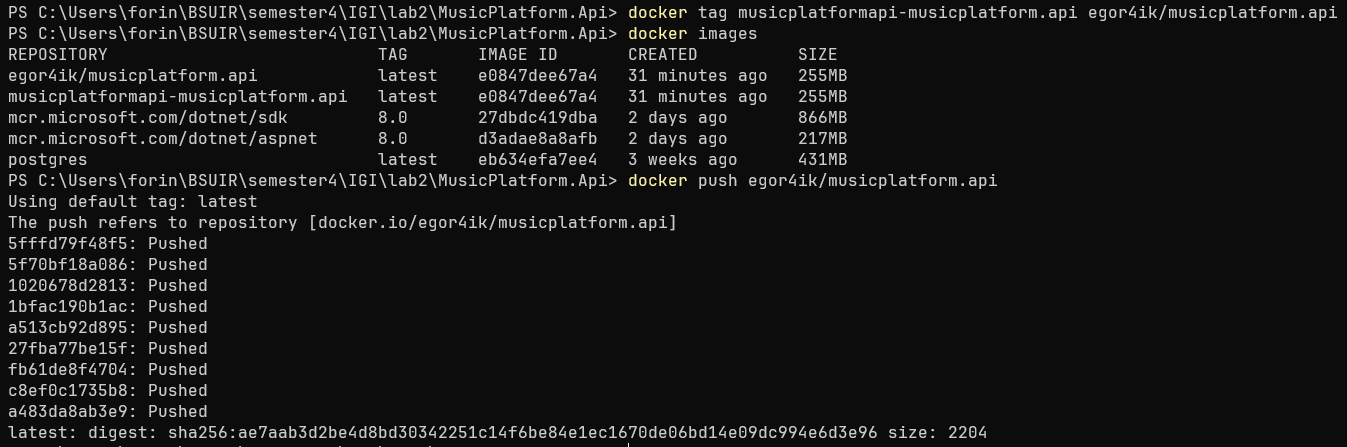
Либо через Docker Desktop

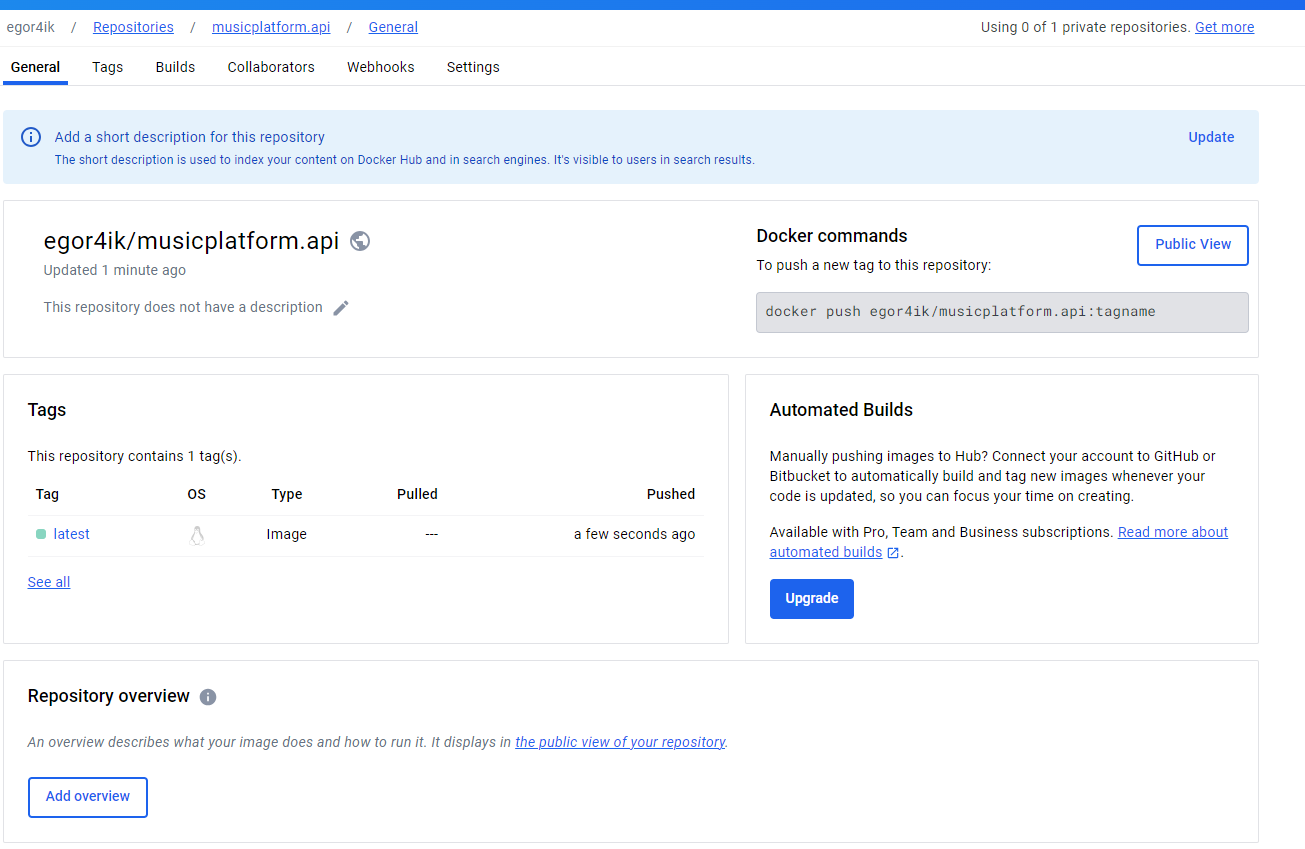


Просмотр всех сервисов  


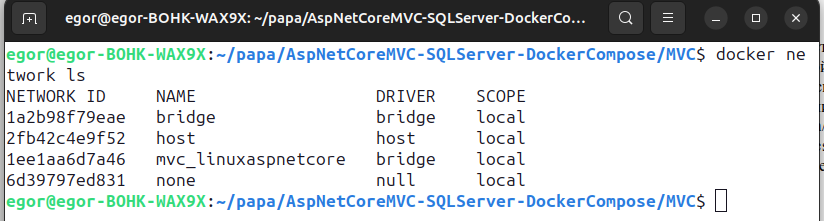
6. Разместите результат в созданный репозиторий в DockerHub

C помощью команды push название\_образа можно разместить его в DockerHub



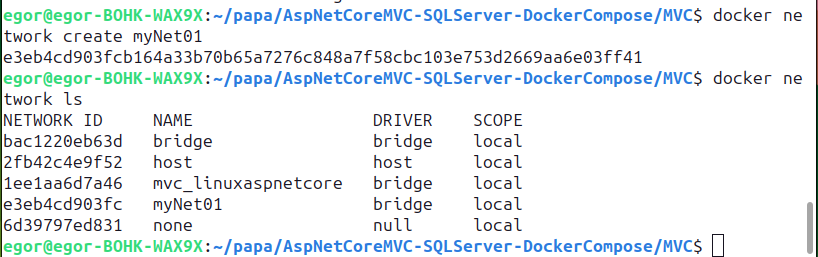


7. Выполните следующие действия с целью изучить особенности сетевого взаимодействия:

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



* 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



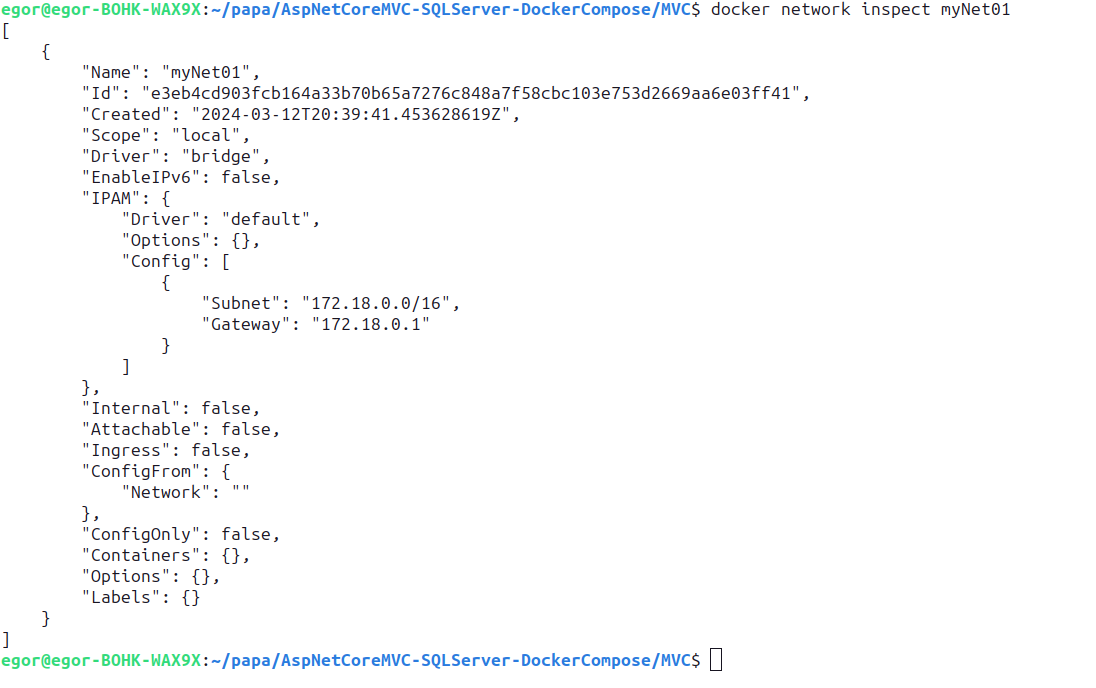






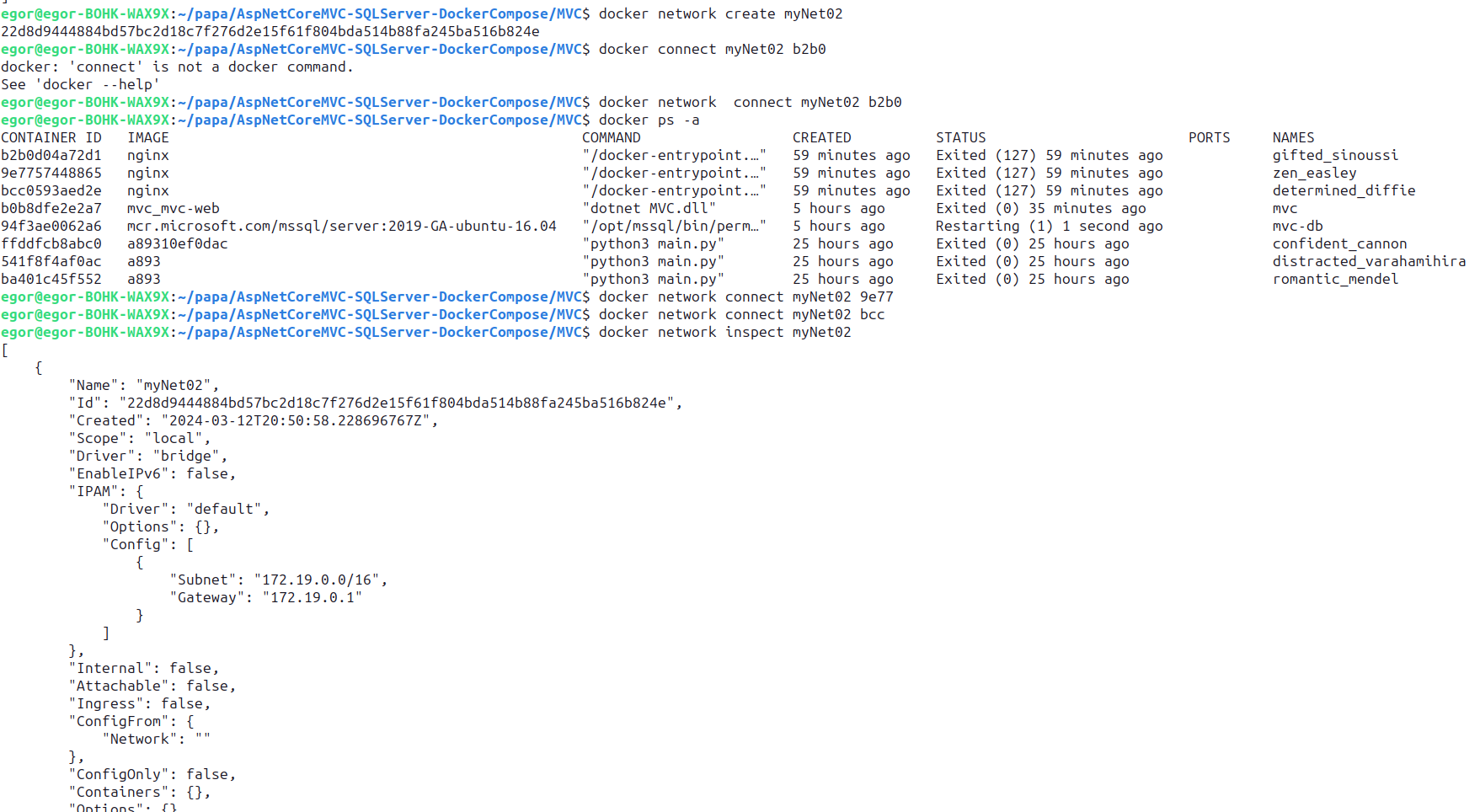


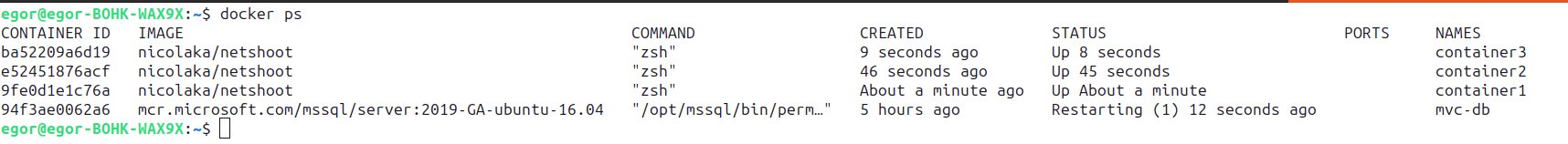




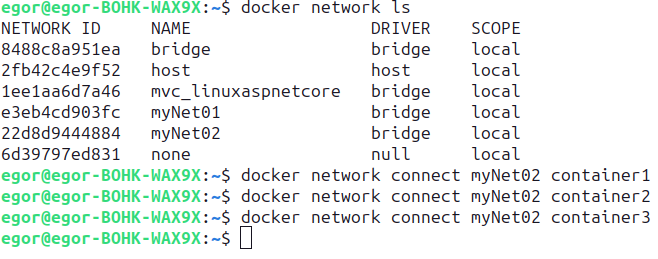
* Создать еще одну сеть bridge, вывести о ней всю информацию, запустить в ней три контейнера, подключиться к любому из контейнеров и пропинговать два других из оболочки контейнера, убедиться, что между контейнерами происходит общение по IP-адреса

Создадим новую сеть

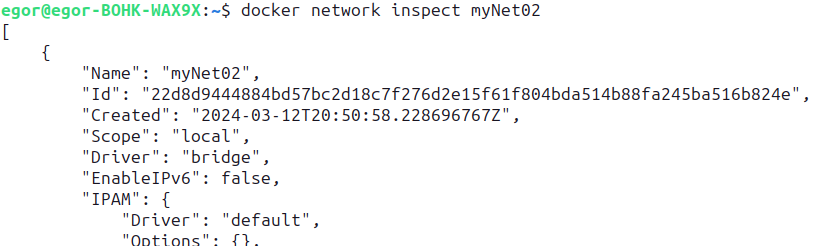




Подключим к сети контейнера



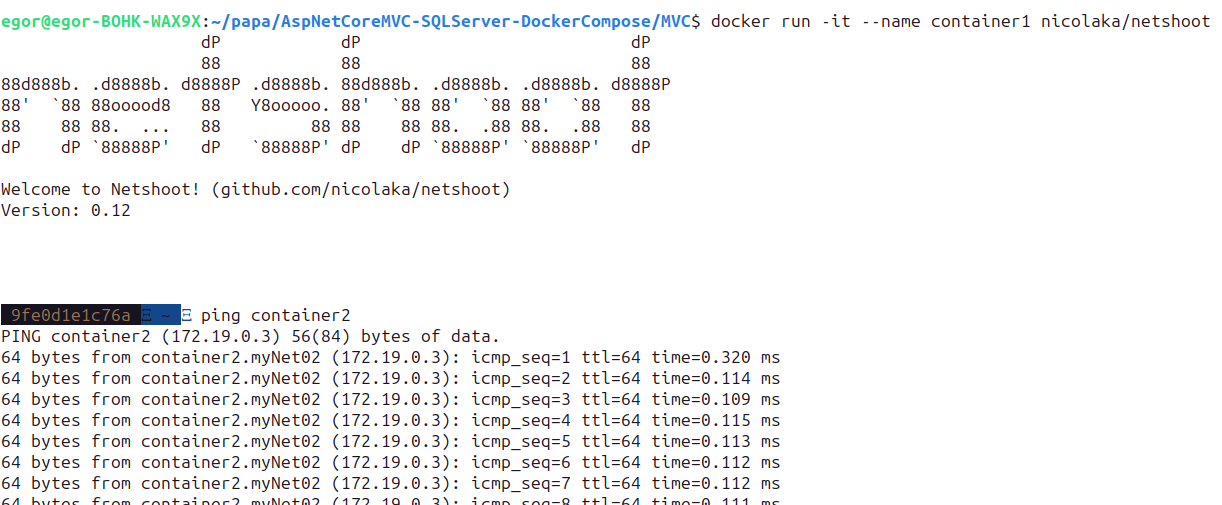
Проинспектируем сеть

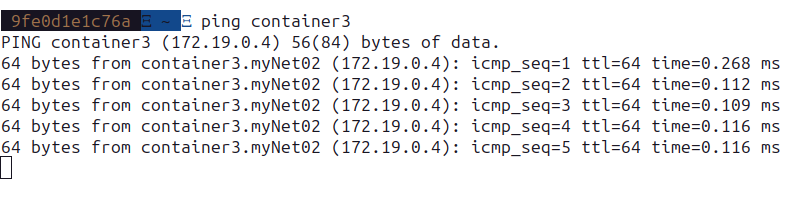


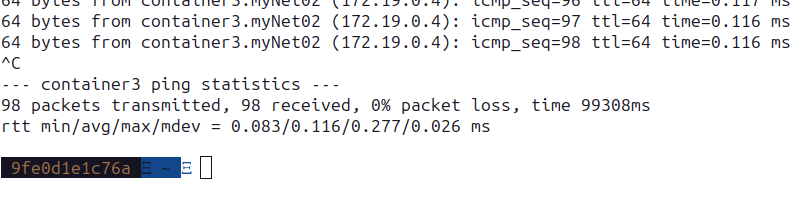
Подключенные к сети контейнеры:



Попробуем пингануть другие контейнеры:

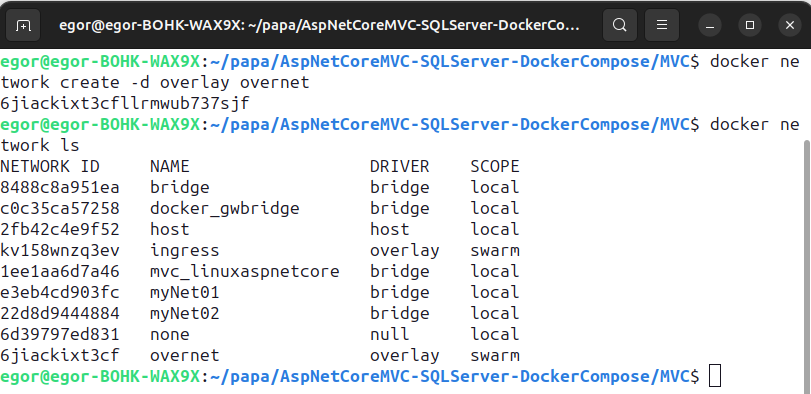


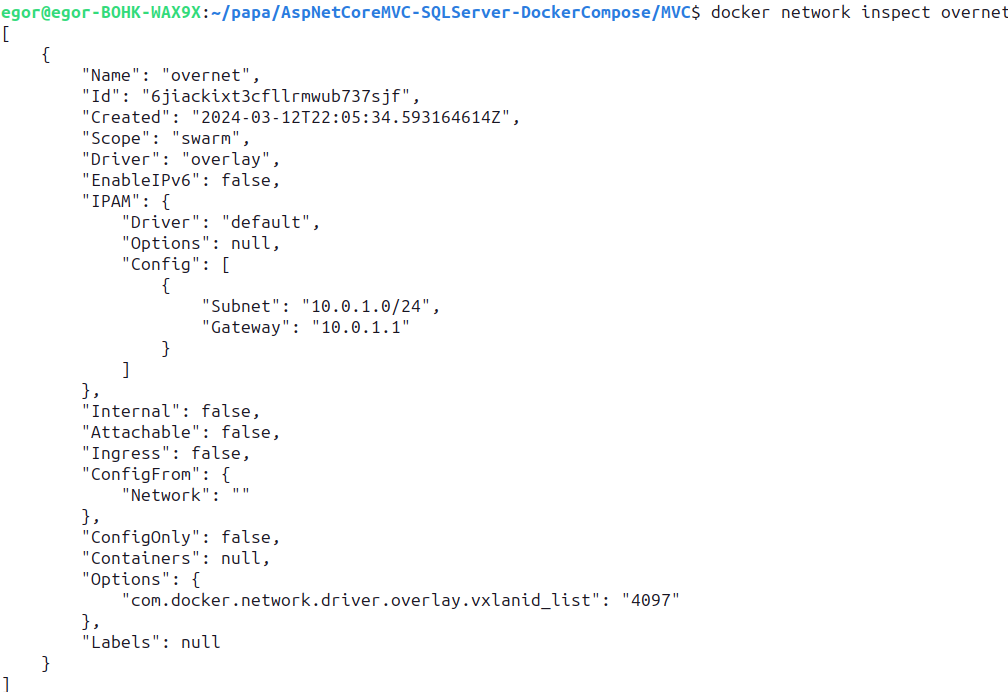


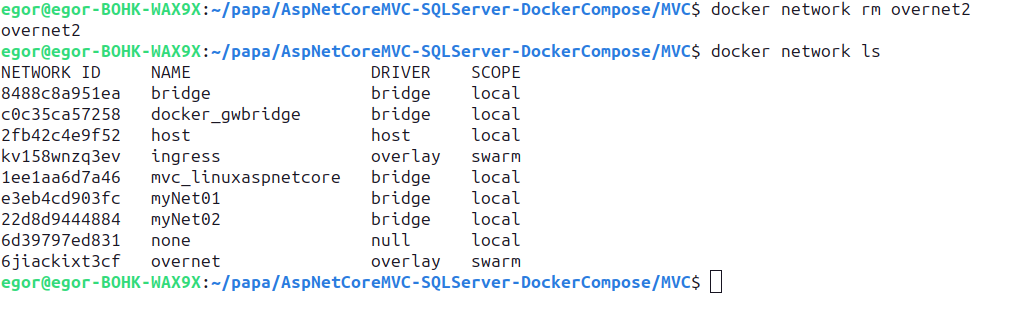
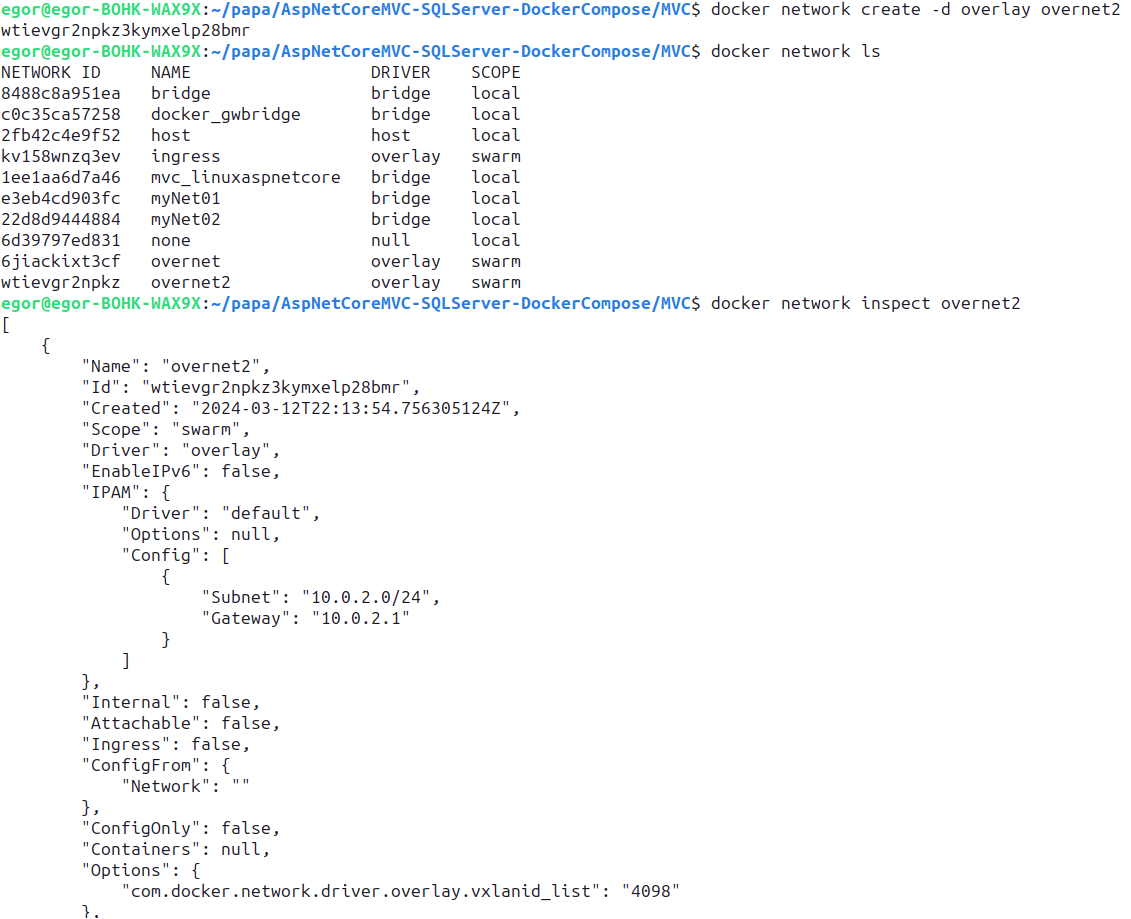


* Создать свою собственную сеть overlay, проверить, создана ли она, вывести о ней всю информацию

**docker swarm init** if not exist





* Создать еще одну сеть overlay, проверить, создана ли она, вывести о ней всю информацию, удалить сеть
* Попробовать создать сеть host, сохранить результат в отчет.

Не создаться, так как host сеть у данного сервера может быть только одна(она так устроена, общается через сервер, у сервера один физический айпи адресс, зато другие могут общаться через порты)

