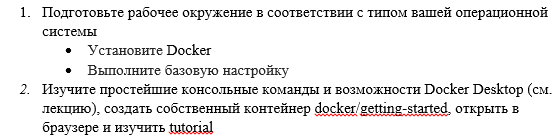
­­­ИГИ

Лабораторная работа №2 docker, вариант 13

Гр. 253505

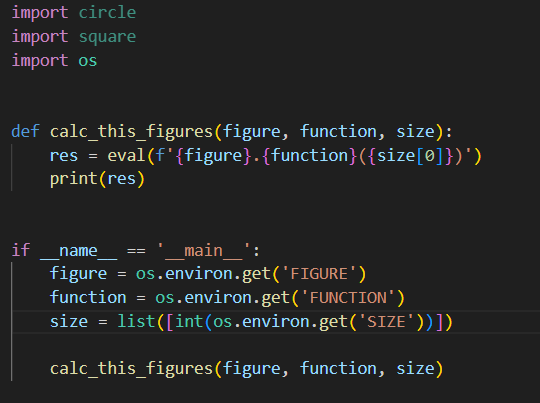
Малашкевич М.И

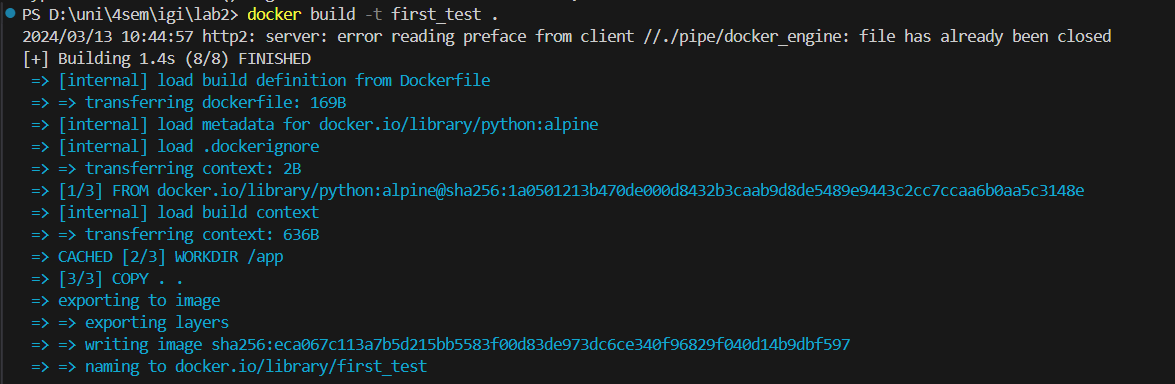


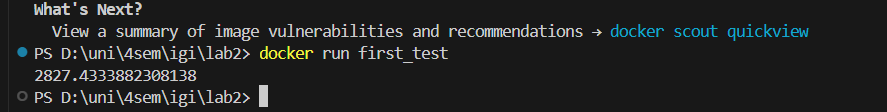


**3. Создайте docker image, который запускает скрипт с использованием функций из** [**https://github.com/smartiqaorg/geometric\_lib**](https://github.com/smartiqaorg/geometric_lib)**.­**

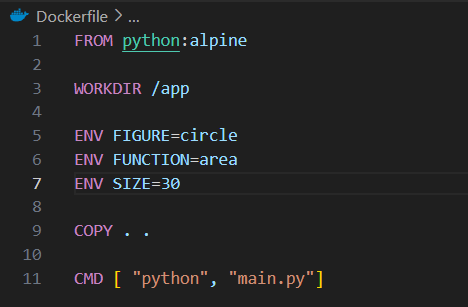
a. Данные необходимые для работы скрипта передайте любым удобным способом (например: конфиг файл через docker volume, переменные окружения, перенаправление ввода). Изучите простейшие консольные команды для работы с docker(см. лекцию). Зарегистрируйтесь на DockerHub и выберите необходимые для проекта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



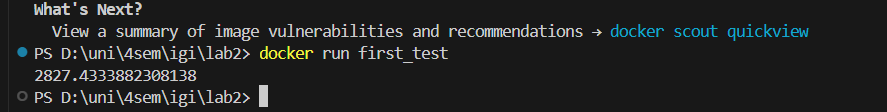


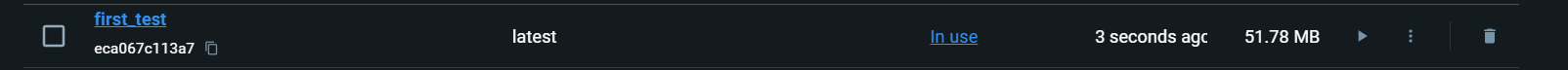


b: Создать Dockerfile для реализации сборки собственных Docker образов.



c: Использовать его для создания контейнера. Протестировать использование контейнера:





**4-5. 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. Настроить сети и тома для обеспечения связи между контейнерами и сохранения данных (исходные данные, логин, пароль и т.д.)**

**Backend docker file:**

FROM node:lts-buster-slim AS development

WORKDIR /usr/src/app

COPY package.json /usr/src/app/package.json

COPY package-lock.json /usr/src/app/package-lock.json

RUN npm ci

COPY . /usr/src/app

EXPOSE 3000

CMD [ "npm", "run", "dev" ]

FROM development as dev-envs

RUN <<EOF

apt-get update

apt-get install -y --no-install-recommends git

EOF

RUN <<EOF

useradd -s /bin/bash -m vscode

groupadd docker

usermod -aG docker vscode

EOF

COPY --from=gloursdocker/docker / /

CMD [ "npm", "run", "dev" ]

**Frontend docker file:**

FROM node:lts-buster­­

WORKDIR /usr/src/app

COPY package.json /usr/src/app

COPY package-lock.json /usr/src/app

RUN npm ci

COPY . /usr/src/app

EXPOSE 3000

CMD ["npm", "start"]

FROM development as dev-envs

RUN <<EOF

apt-get update

apt-get install -y --no-install-recommends git

EOF

RUN <<EOF

useradd -s /bin/bash -m vscode

groupadd docker

usermod -aG docker vscode

EOF

COPY --from=gloursdocker/docker / /

CMD [ "npm", "start" ]

**Docker-compose:**

services:

  frontend:

    image: miksailmalashkevich/react-no-app-web

    build:

      context: frontend

      target: development

    ports:

      - 3000:3000

    stdin\_open: true

    volumes:

      - ./frontend:/usr/src/app

      - /usr/src/app/node\_modules

    restart: always

    networks:

      - react-express

    depends\_on:

      - backend

  backend:

    image: miksailmalashkevich/back-for-no-app-wev

    restart: always

    build:

      context: backend

      target: development

    volumes:

      - ./backend:/usr/src/app

      - /usr/src/app/node\_modules

    depends\_on:

      - mongo

    networks:

      - express-mongo

      - react-express

    expose:

      - 3000

  mongo:

    restart: always

    image: mongo:4.2.0

    volumes:

      - mongo\_data:/data/db

    networks:

      - express-mongo

    expose:

      - 27017

networks:

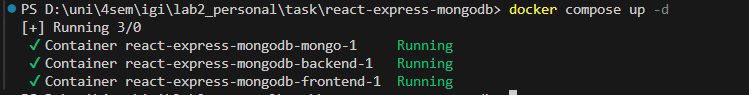
  react-express:

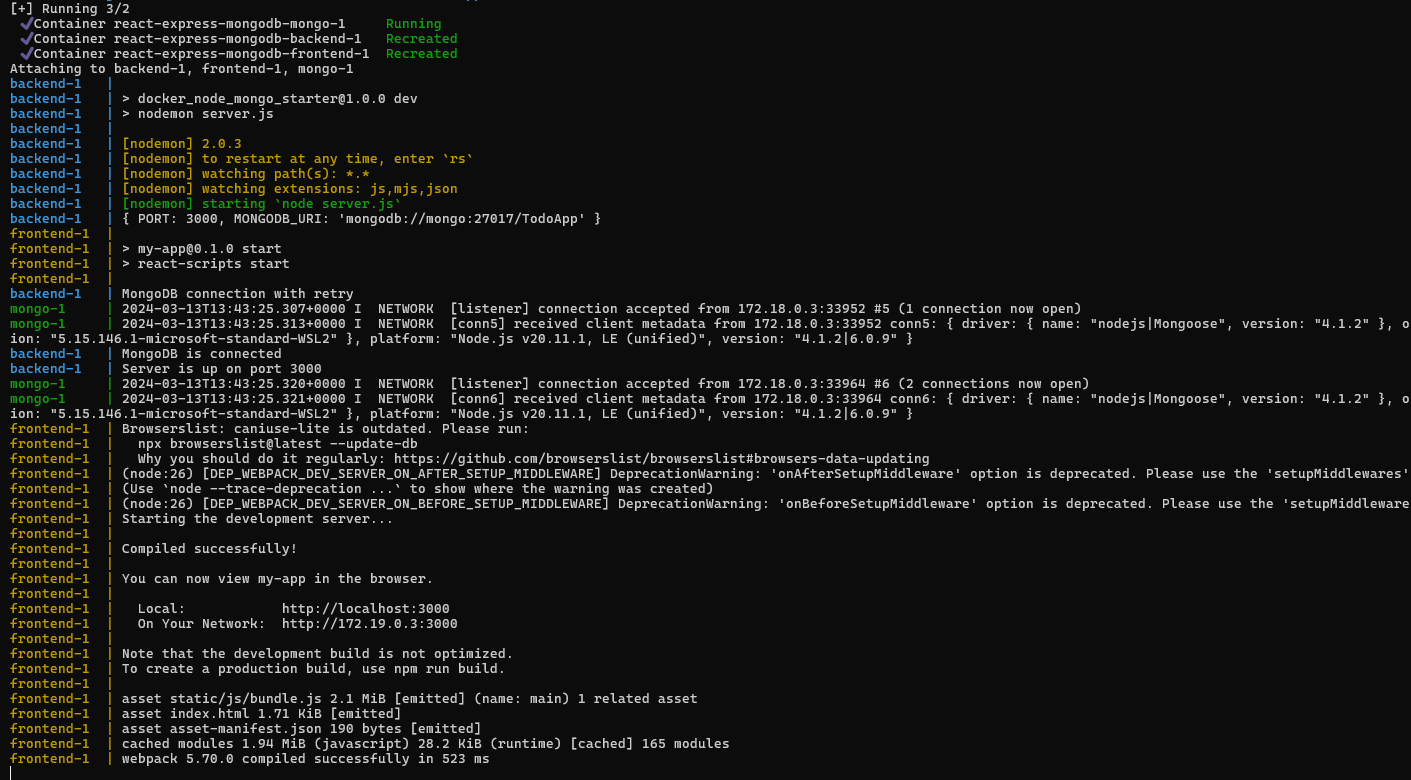
  express-mongo:

volumes:

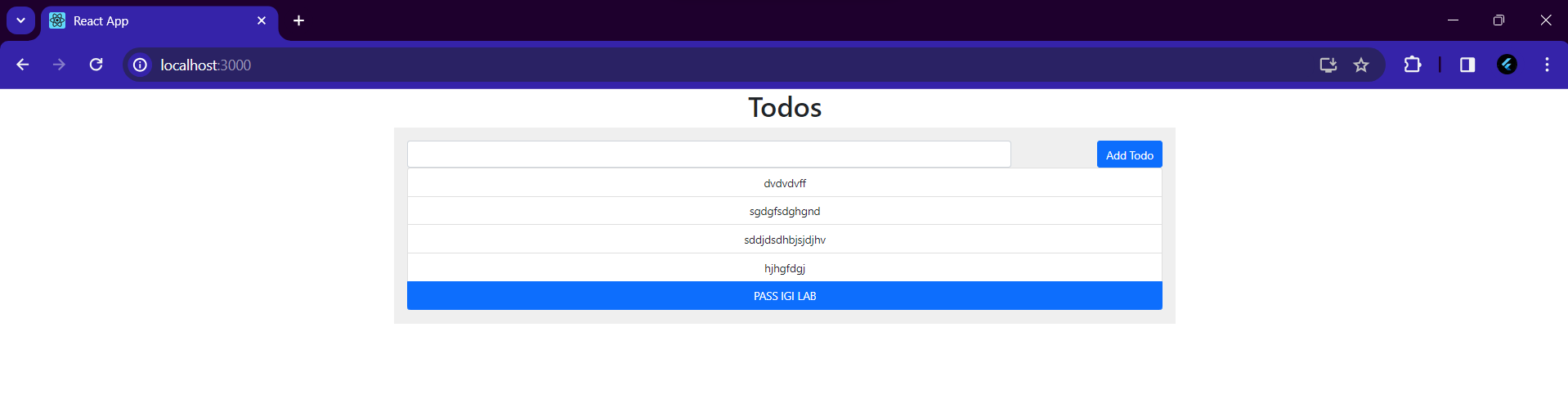
  mongo\_data

**Builduig(did it previous btw):**

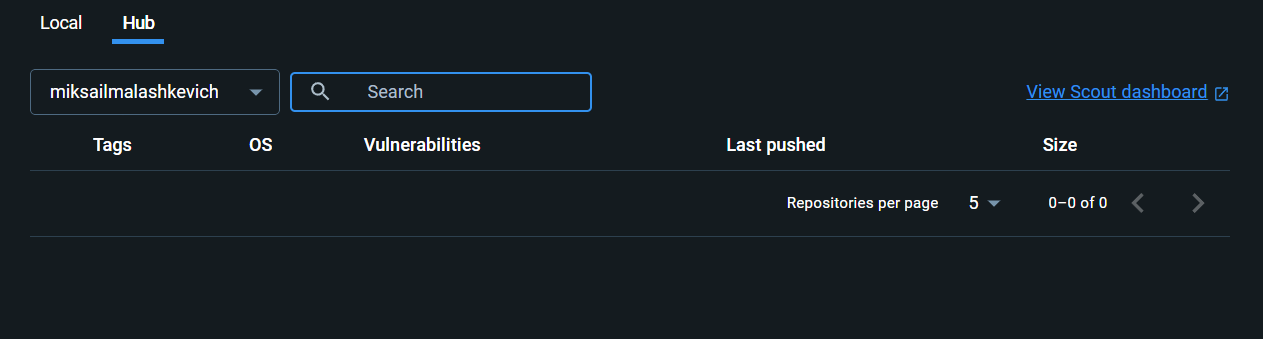
****

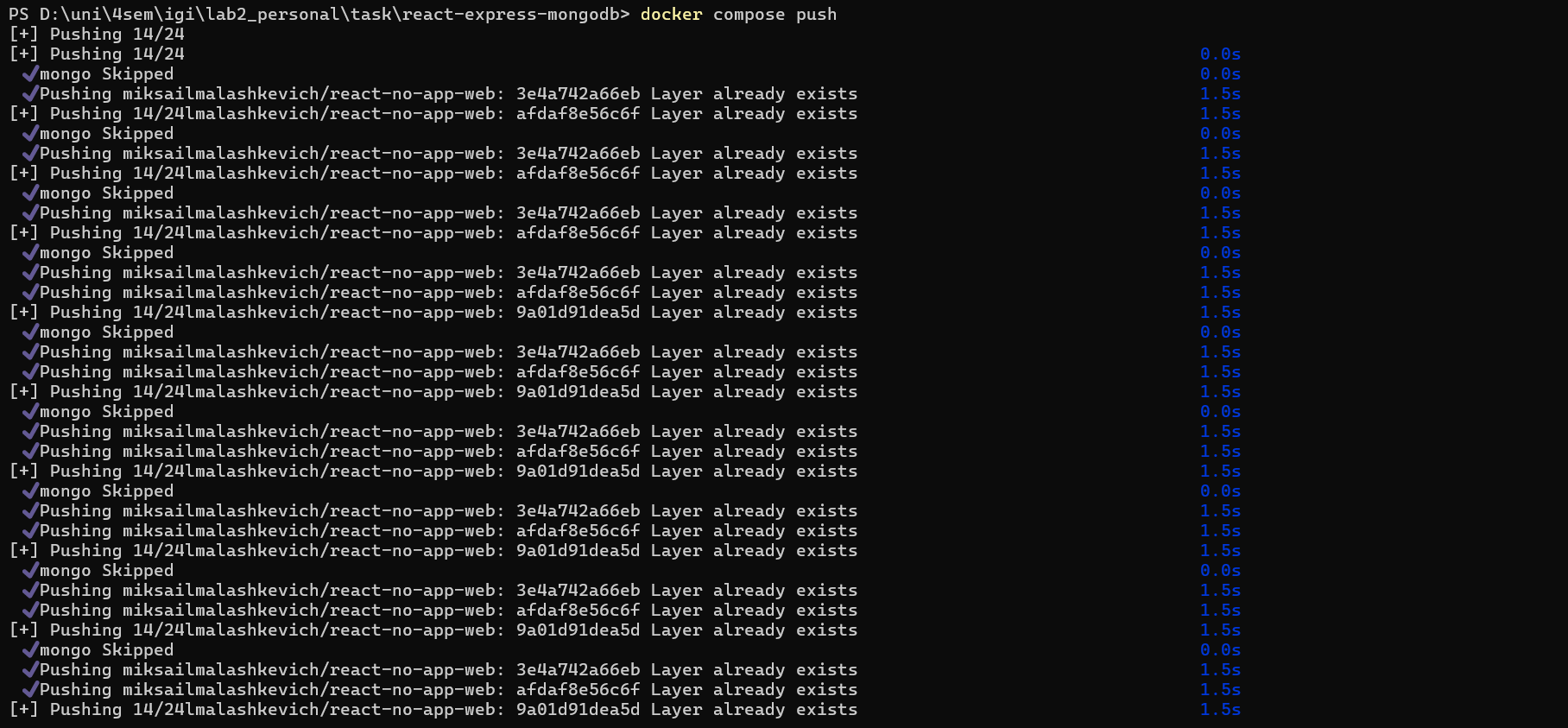
****

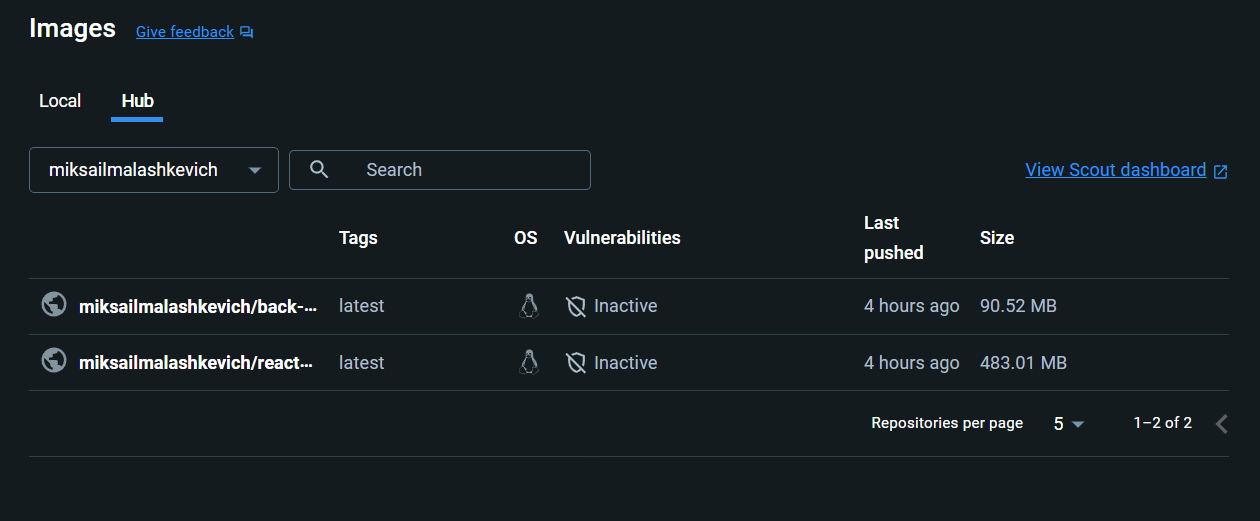
**App itself:**

****

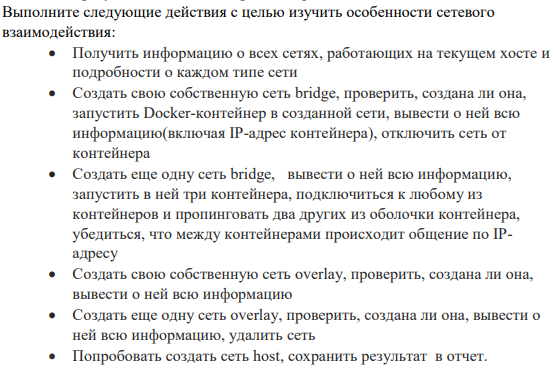
**6. Разместите результат в созданный репозиторий в DockerHub.**

****

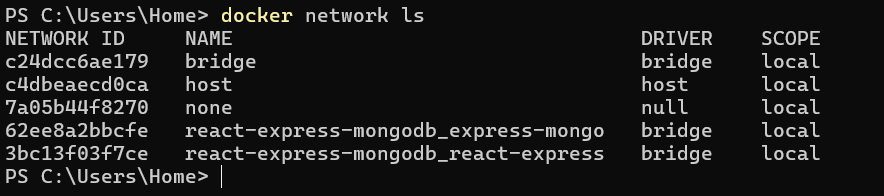
****

****

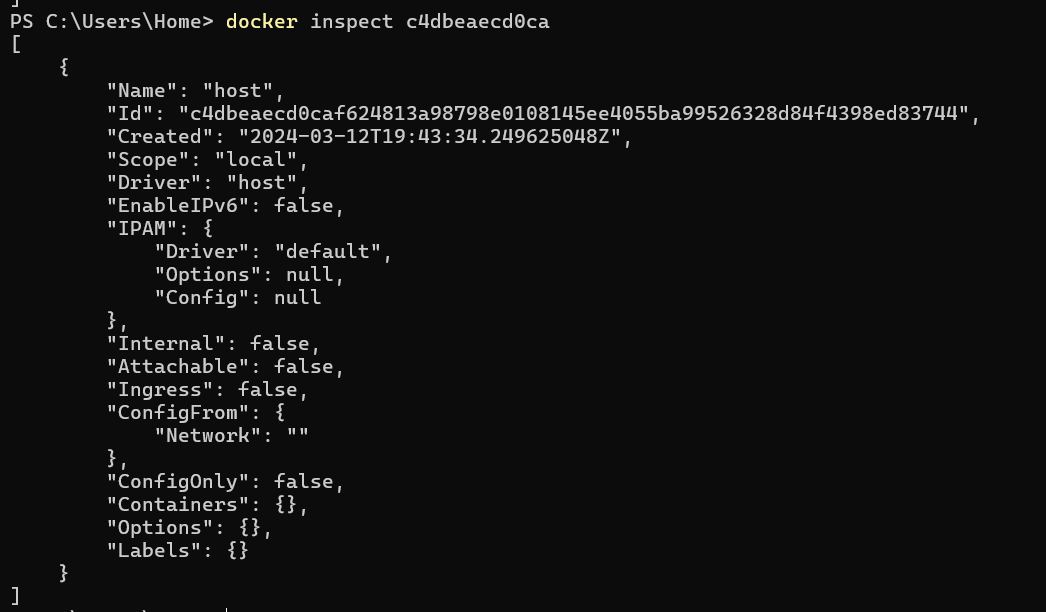
**7.**

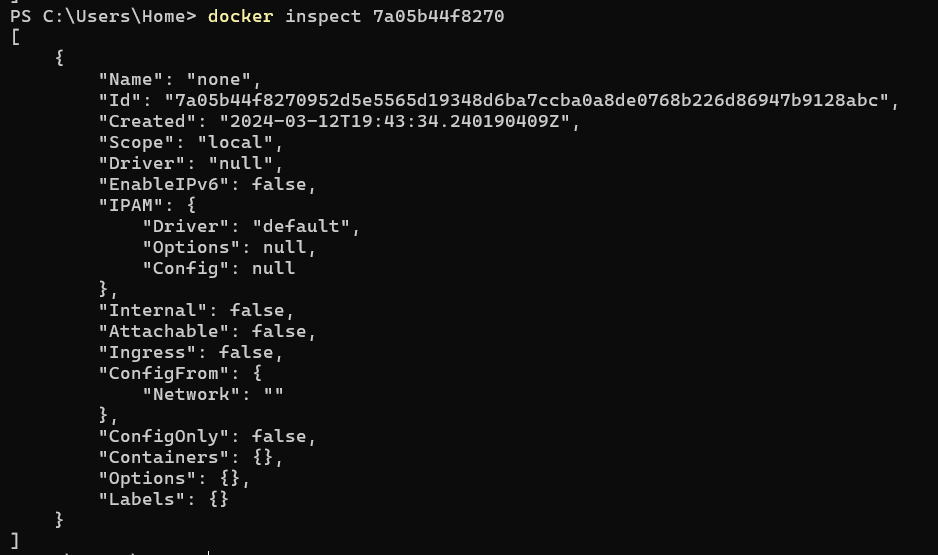


Получить информацию о всех сетях, работающих на текущем хосте и подробности о каждом типе сети:





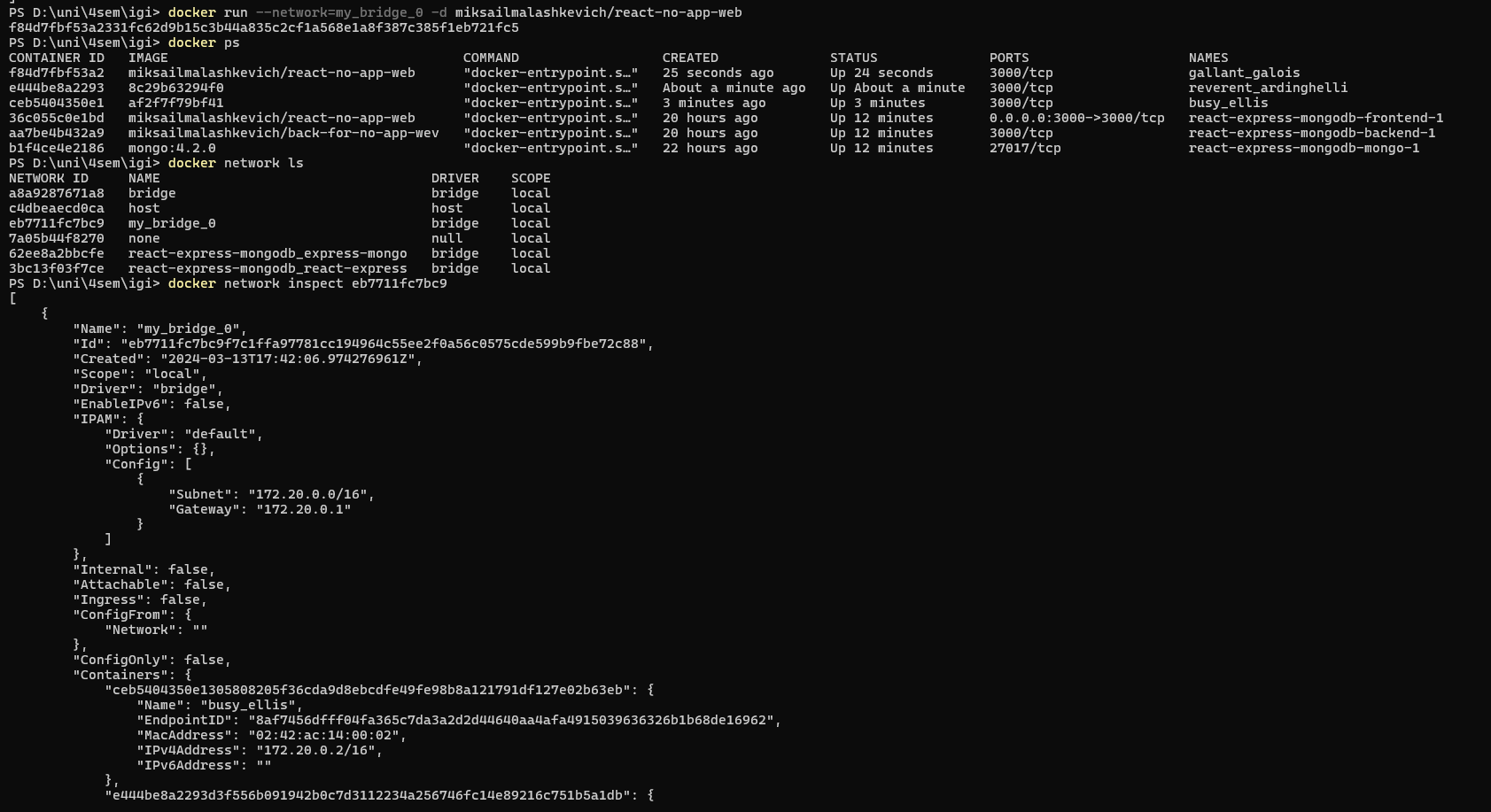


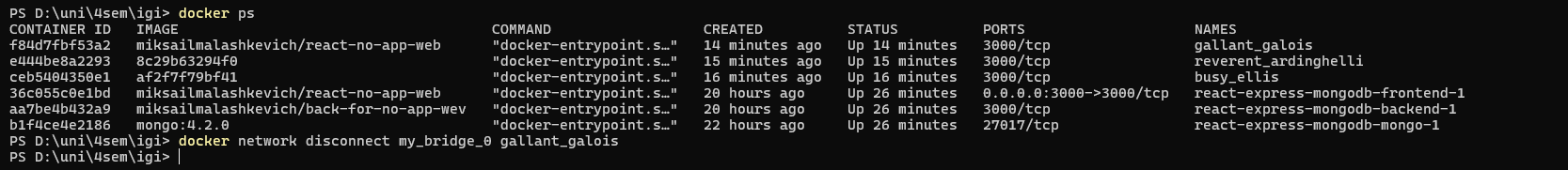




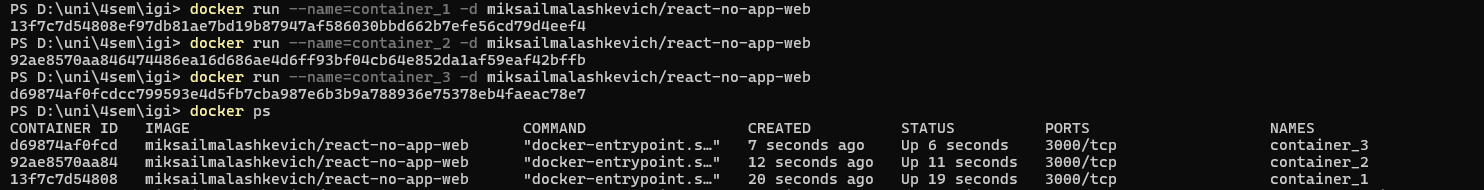


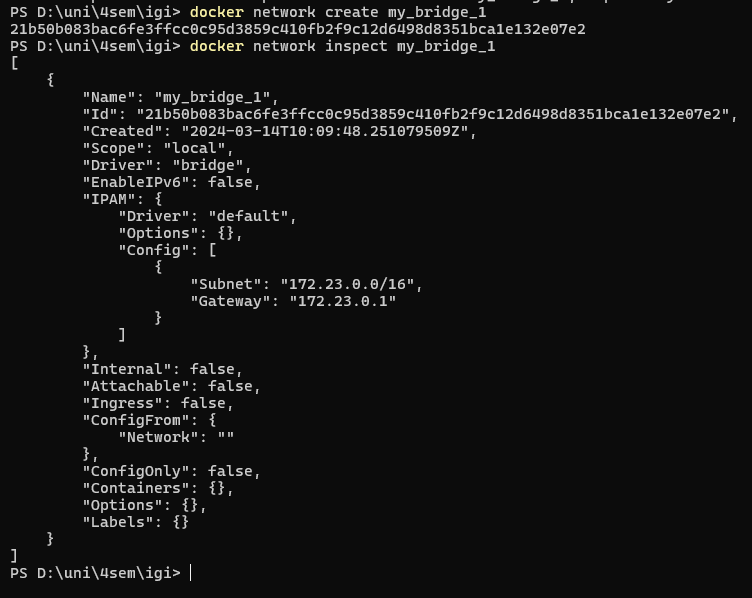
Создать свою собственную сеть bridge, проверить, создана ли она, запустить Docker-контейнер в созданной сети, вывести о ней всю информацию (включая IP-адрес контейнера), отключить сеть от контейнера



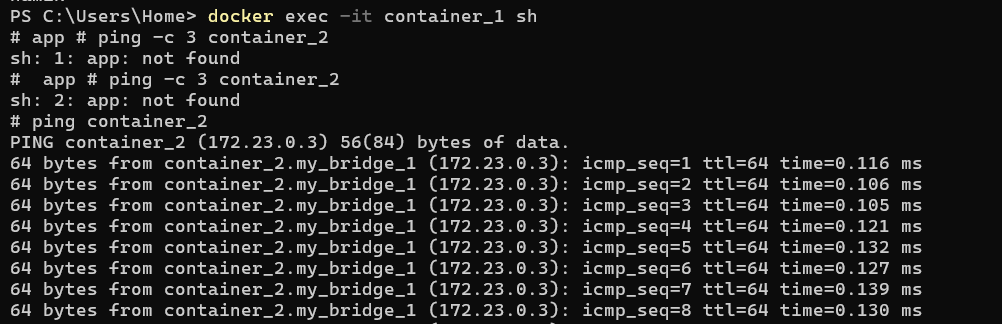


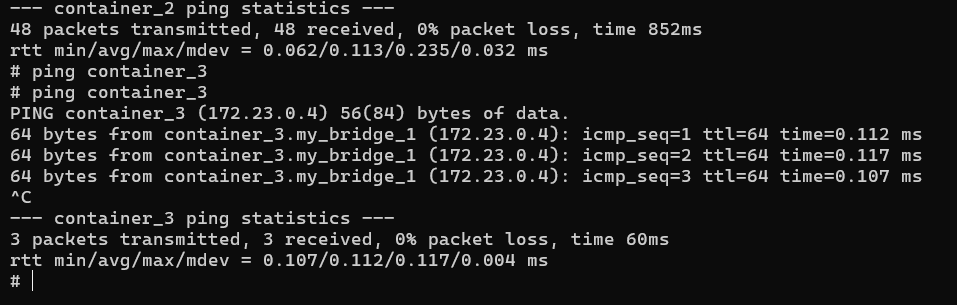
Создать еще одну сеть bridge, вывести о ней всю информацию, запустить в ней три контейнера, подключиться к любому из контейнеров и пропинговать два других из оболочки контейнера, убедиться, что между контейнерами происходит общение по IPадресу:



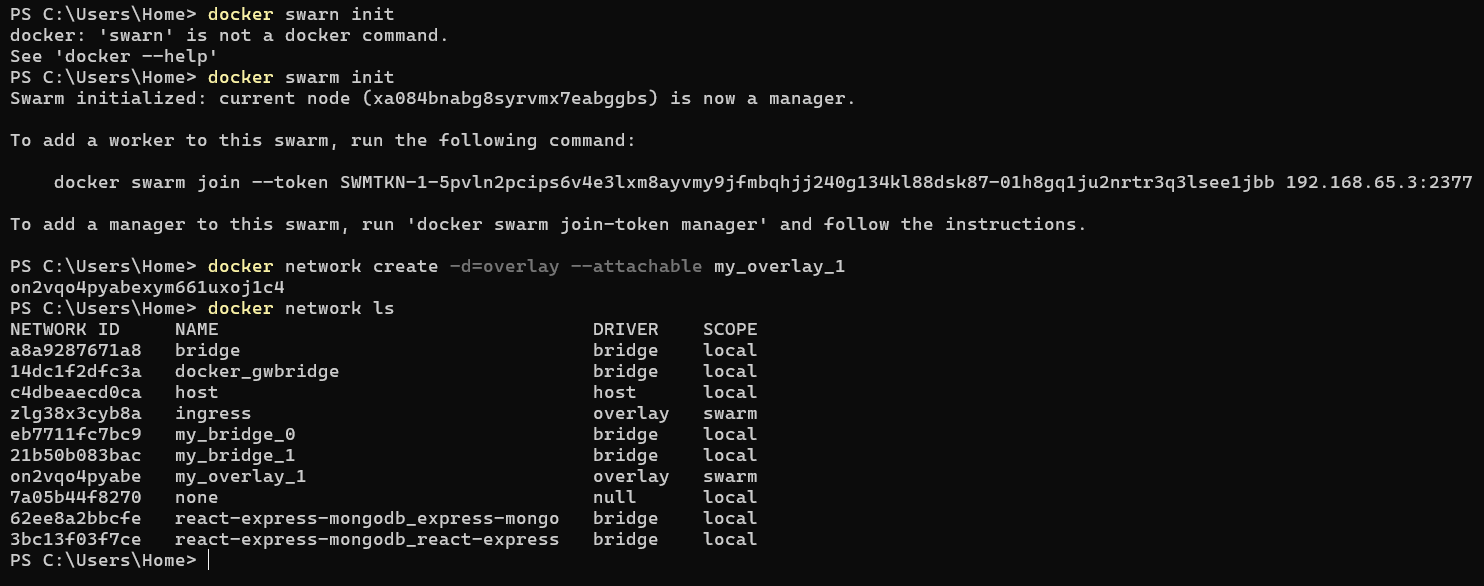




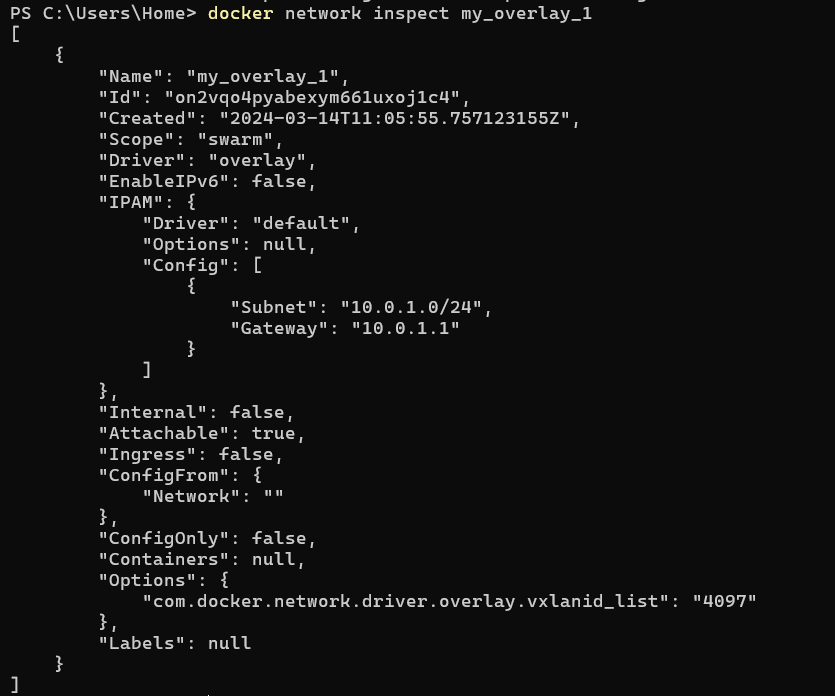




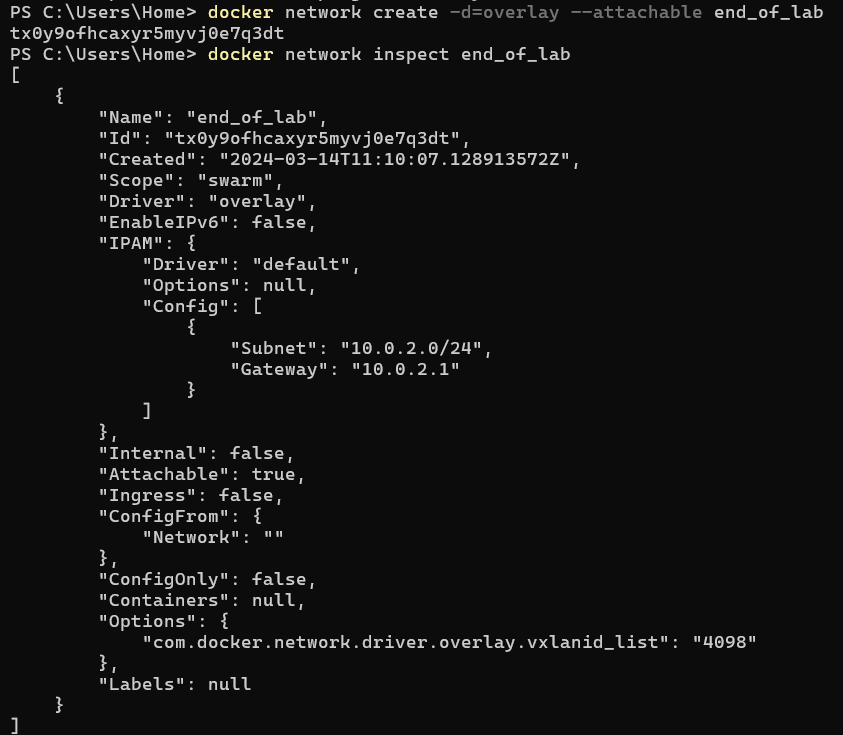
Создать свою собственную сеть overlay, проверить, создана ли она, вывести о ней всю информацию:

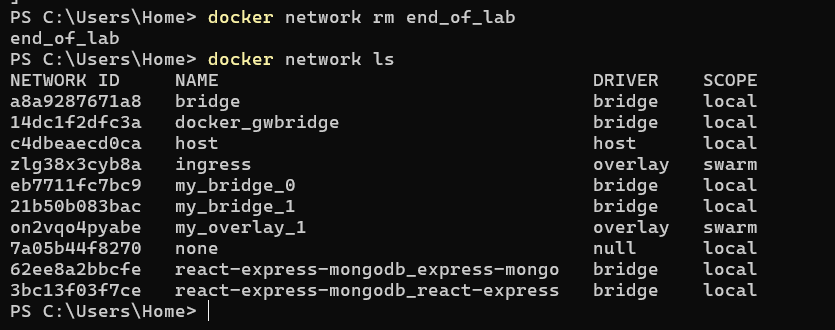


Создали оверлейную сеть а также сам оверлей.



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

****

****