МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

(ФГБОУ ВО «ВГТУ», ВГТУ)

Факультет информационных технологий и компьютерной безопасности

# Кафедра автоматизированных и вычислительных систем

ОТЧЁТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №6

По дисциплине: «Системы хранения и обработки данных»

Тема: «Развёртывание СУБД Postgres с использованием средств автоматизации развёртывания и управления приложениями»

Выполнил работу студент группы змИИВТ-241: Матыкина А.В.

подпись, дата

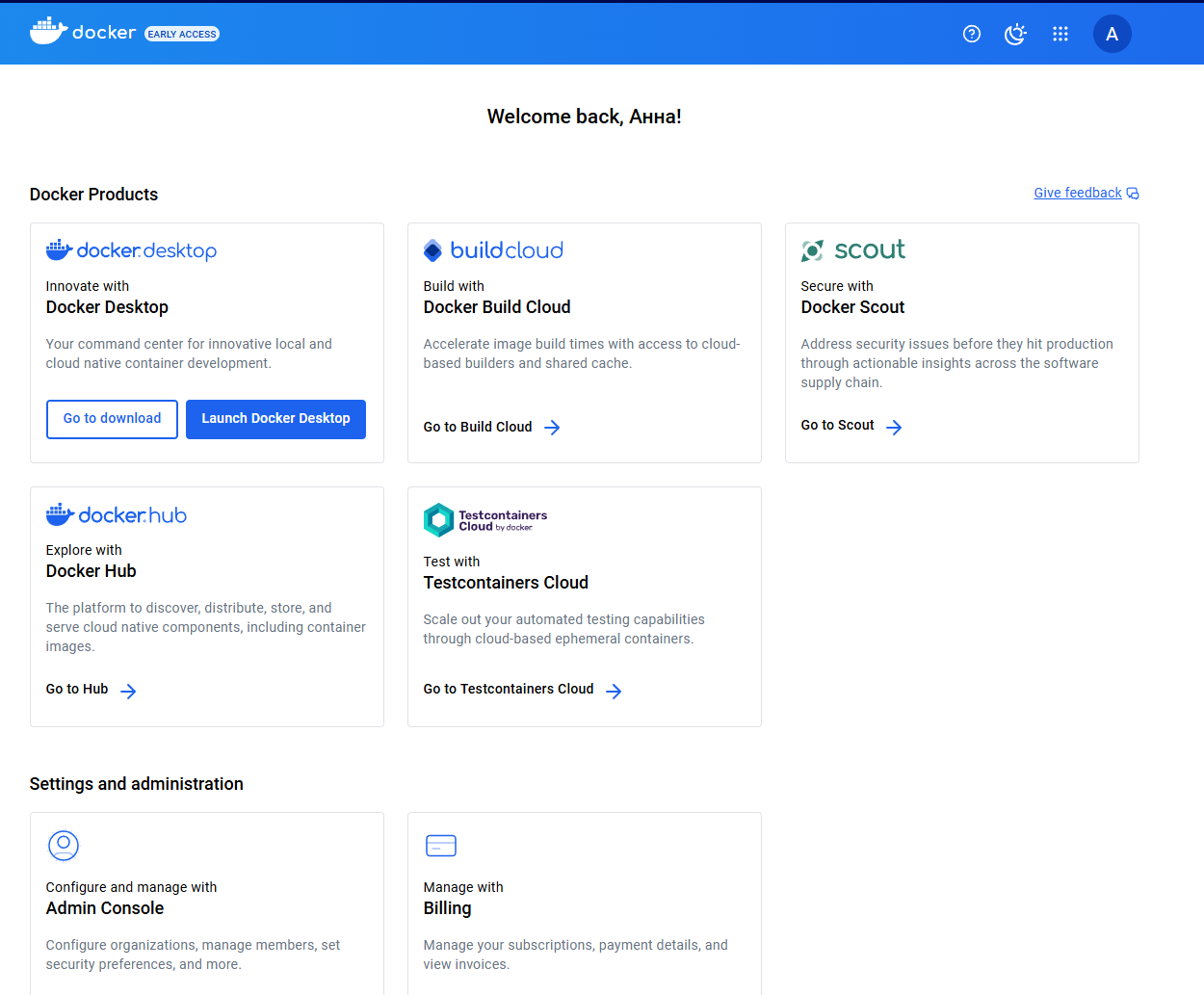
Принял: Короленко В.В.

подпись, дата

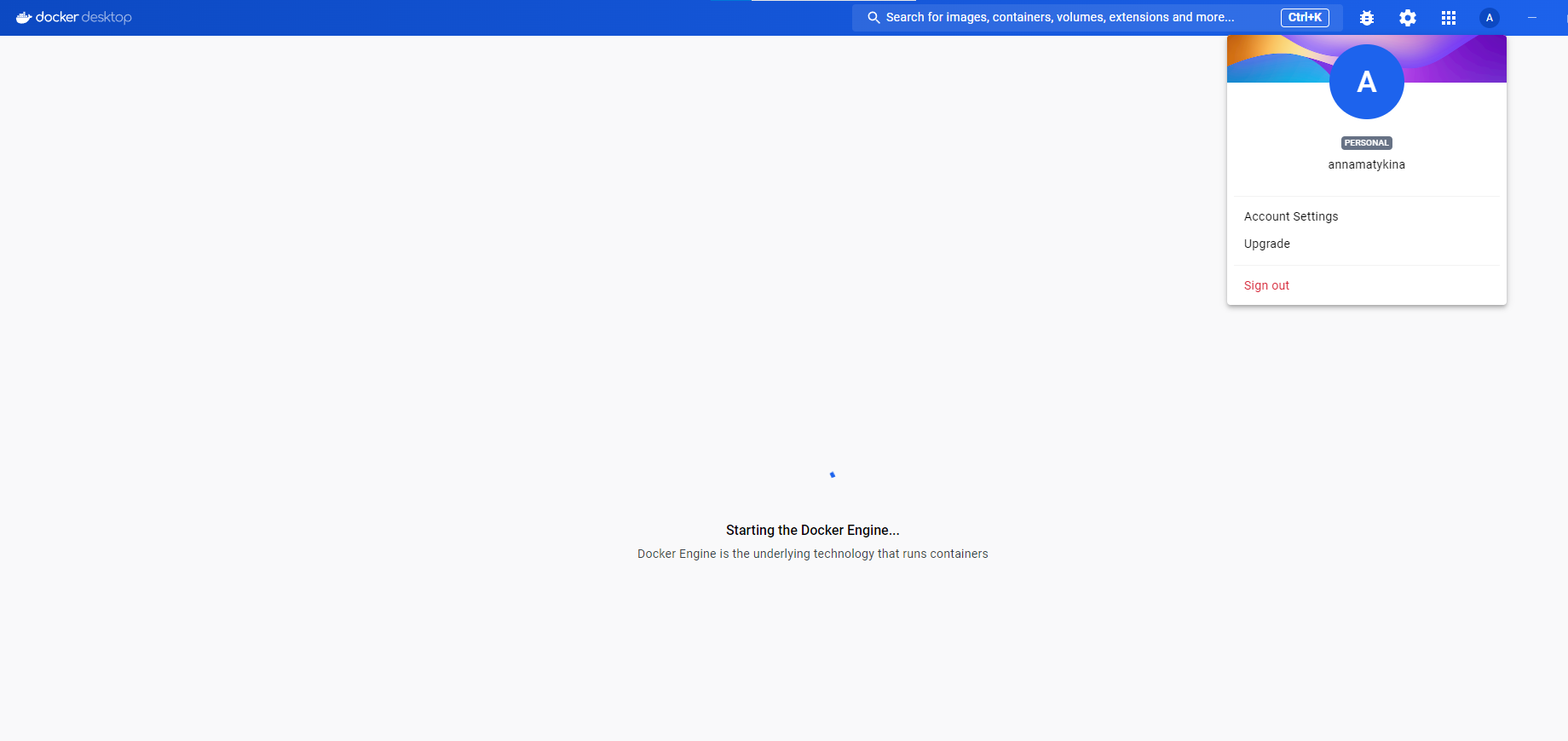
Воронеж 2024

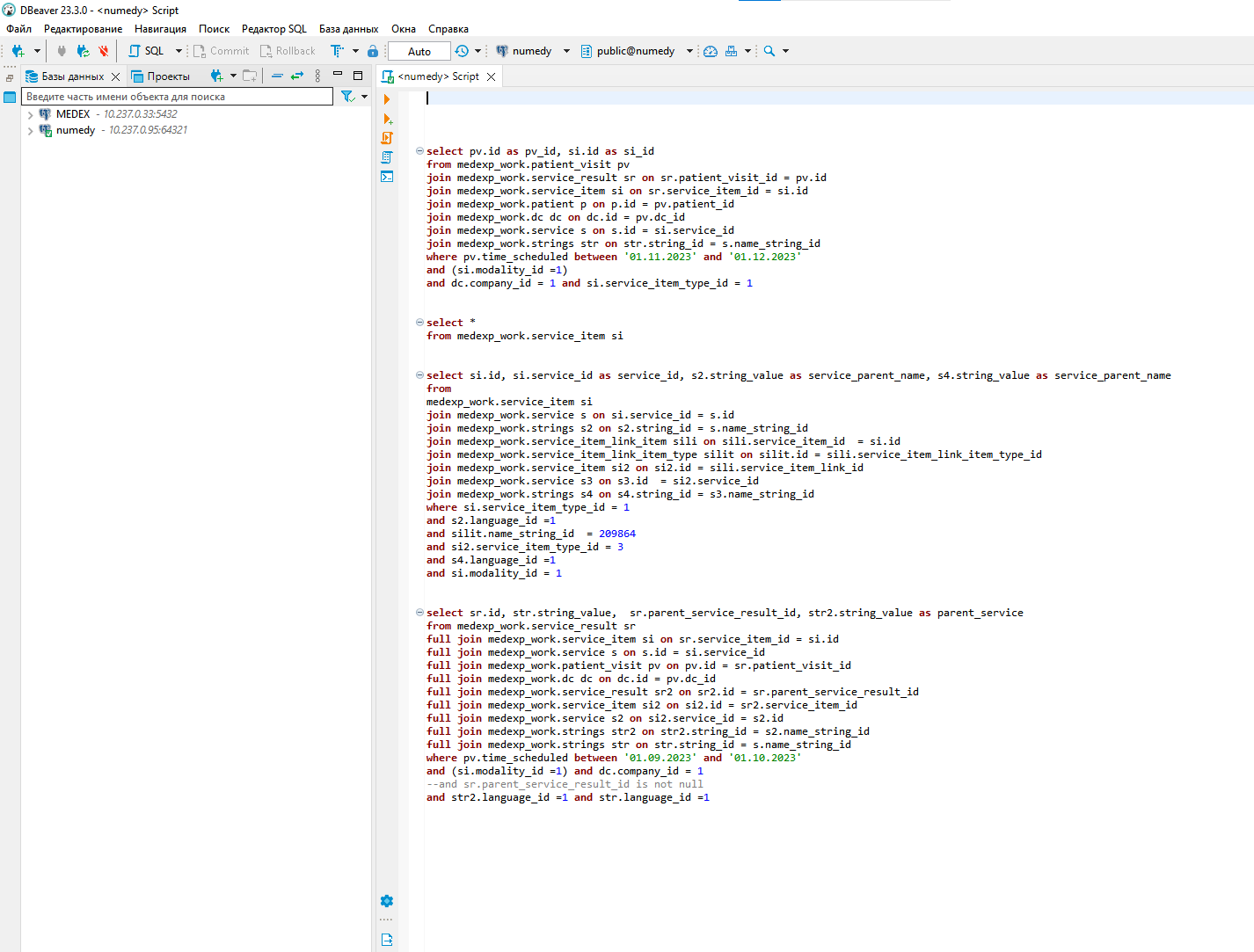
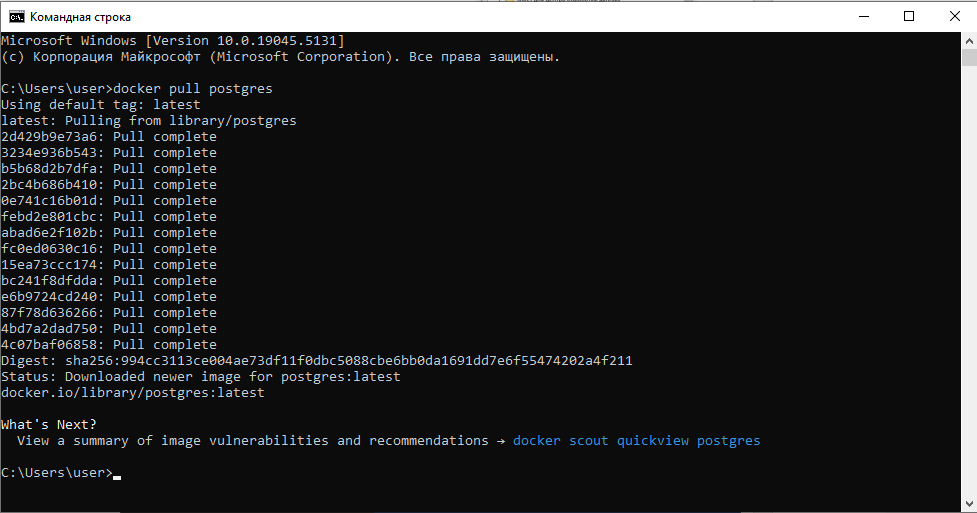
# **Учебная задача**

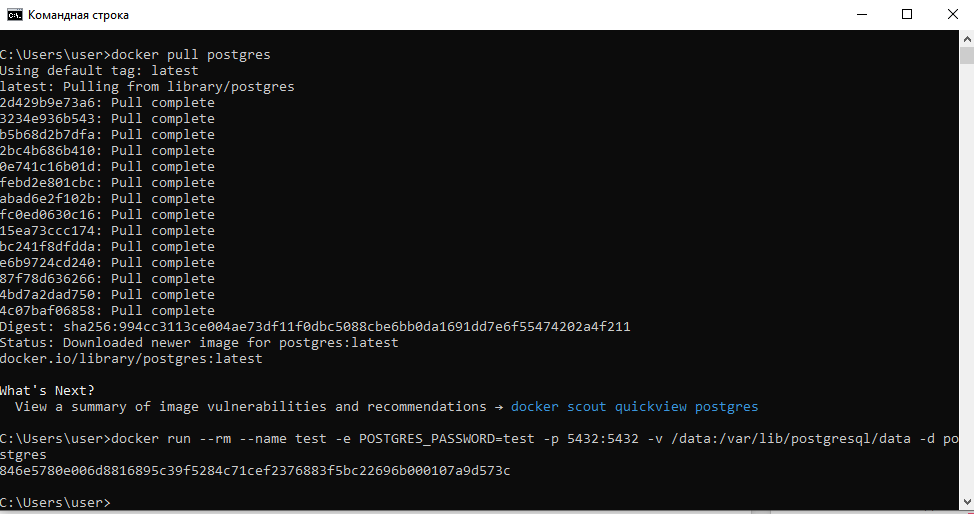
1. Изучить сайт [https://www.docker.com](https://www.docker.com/), документацию по docker (https://docs.docker.com), зарегистрироваться на Docker Hub. Изучить основные команды docker для консоли.

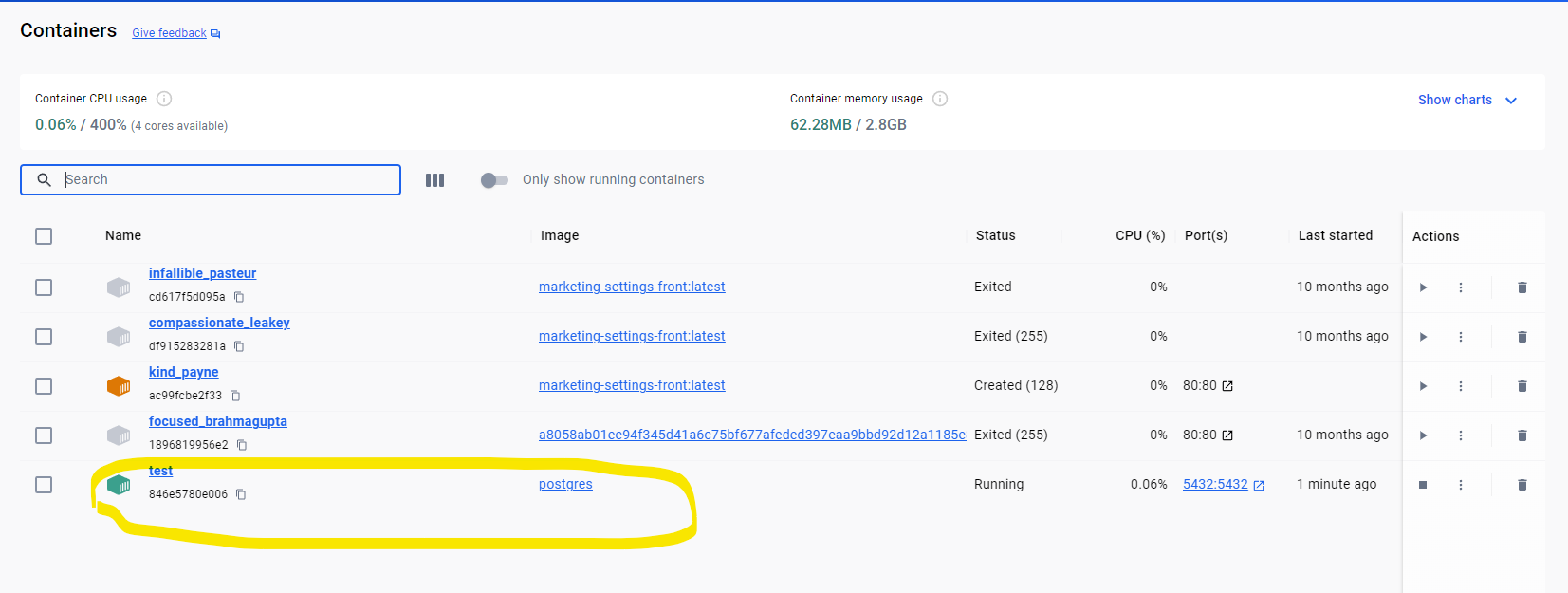


1. Скачать и установить Docker Desktop

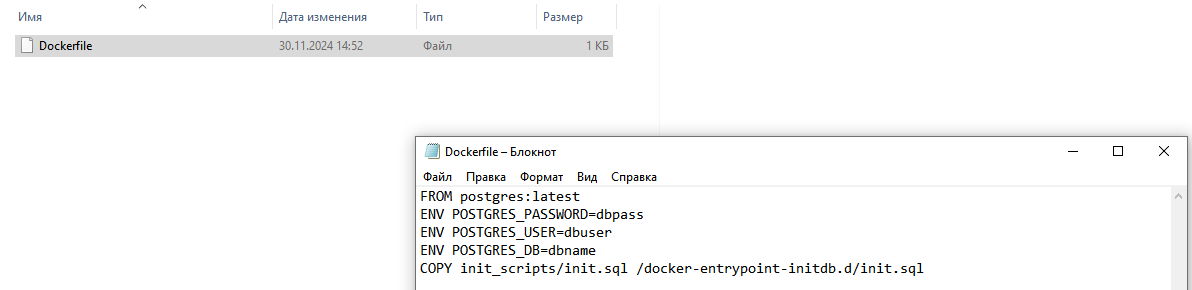


1. Скачать и установить средство для работы с СУБД DBeaver
2. Скачать docker-образ с СУБД postgres с помощью команды для консоли.
3. .





Создадим Dockerfile:



Команды:

FROM postgres:latest – указывает, откуда брать родительский образ контейнера

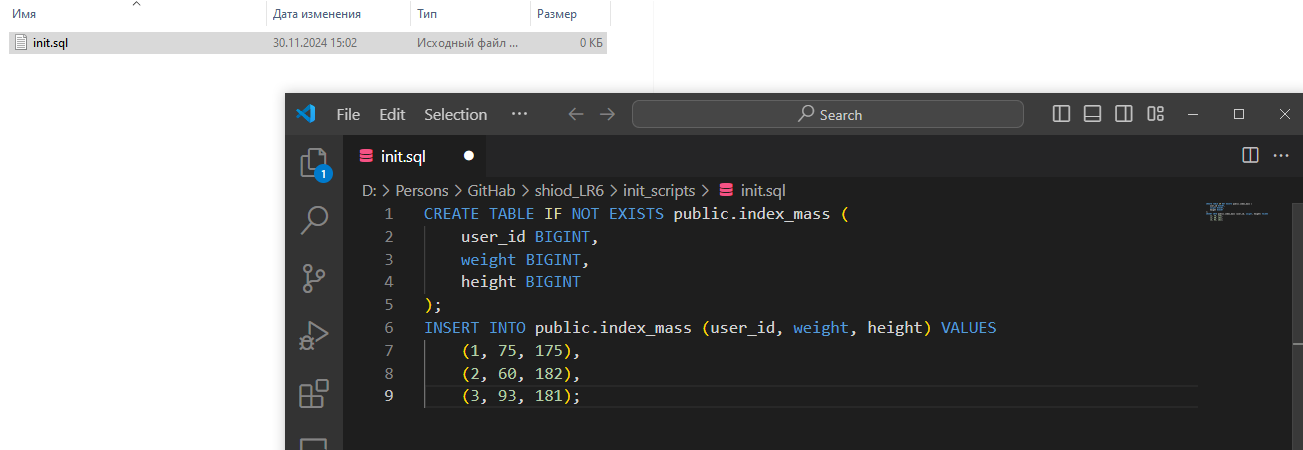
ENV POSTGRES\_PASSWORD=dbpass – устанавливает постоянные переменные среды, в данном случае пароль к БД

ENV POSTGRES\_USER=dbuser – устанавливает постоянные переменные среды, в данном случае логин к БД

ENV POSTGRES\_DB=dbname устанавливает постоянные переменные среды, в данном наименование БД

COPY init\_scripts/init.sql /docker-entrypoint-initdb.d/init.sql – копирует в контейнер папки и файлы

Создала sql файл в рабочей директории:



CREATE TABLE IF NOT EXISTS public.index\_mass ( // создать таблицу, если она не создана

user\_id BIGINT, // таблицу создавать с колонками user\_id, тип данных в колонке BIGINT

weight BIGINT, // таблицу создавать с колонками weight , тип данных в колонке BIGINT

height BIGINT // таблицу создавать с колонками height, тип данных в колонке BIGINT

);

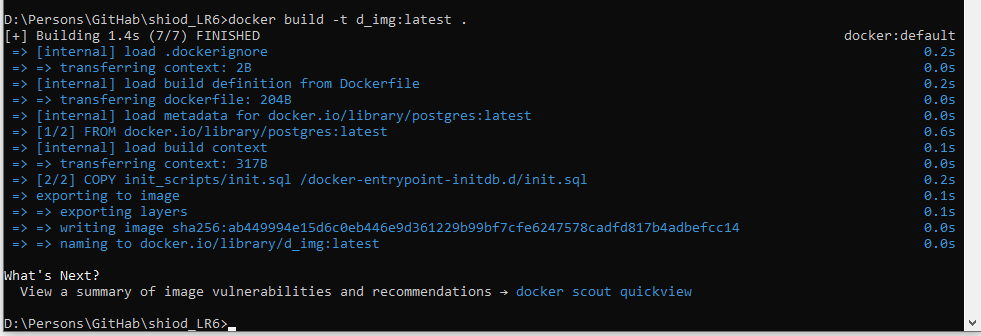
INSERT INTO public.index\_mass (user\_id, weight, height) VALUES // заполнить таблицу значениями

(1, 75, 175),

(2, 60, 182),

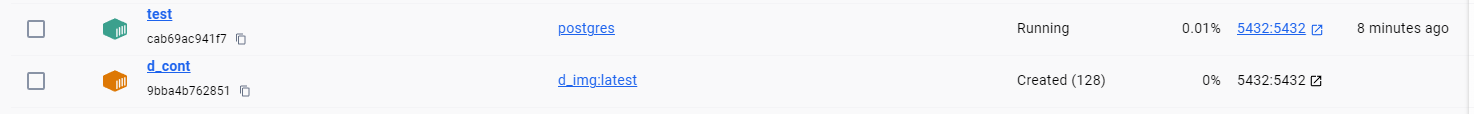
(3, 93, 181);

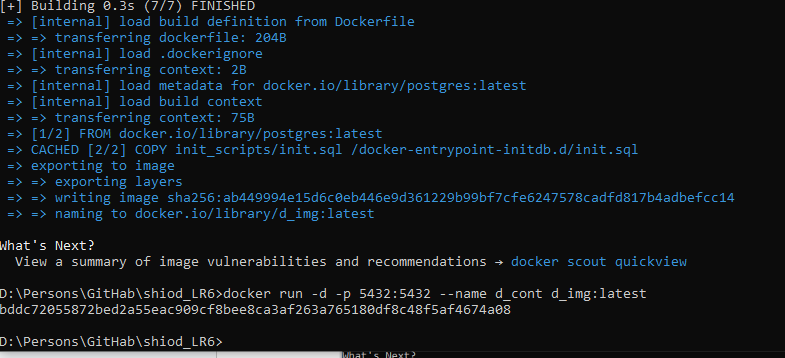
1. Создать образ с помощью команды «docker build -t d\_img:latest .».



1. Запустить контейнер с именем «d\_cont» на основе образа «d\_img:latest» с помощью команды: «docker run -d -p 5432:5432 --name d\_cont d\_img:latest»  
   Эта команда запустит нам контейнер в фоновом (detached) режиме, используя порт 5432:5432, и присвоит ему имя d\_cont на основе образа d\_img:latest (созданном на предыдущем этапе)

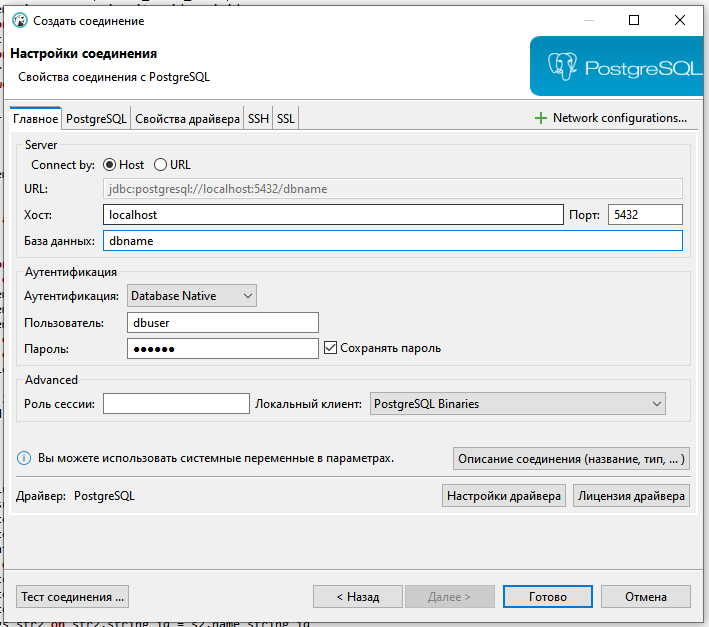
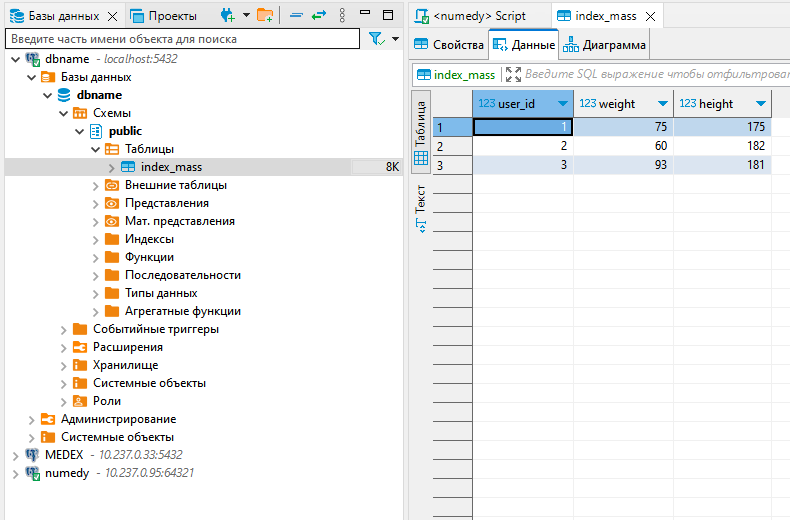
Докер-клиент нашел образ, загрузил контейнер и запустил команду внутри этого контейнера. Но внутри данного контейнера пока команд не было







1. Проверить через DBeaver развёрнутую СУБД.

1. Написать специальную команду docker, которая позволит подключаться к работающему контейнеру, запускать интерфейс psql и вносить новые данные «на лету»: «docker exec -it d\_cont psql -d -U usr dbn».

Команда «docker exec -it test psql -U user -d database» - запуск контейнера в интерактивной среде с интерпретатором СУБД postgres под пользователем с именем «user»

Подробнее:

· docker exec – подключение к работающему контейнеру.

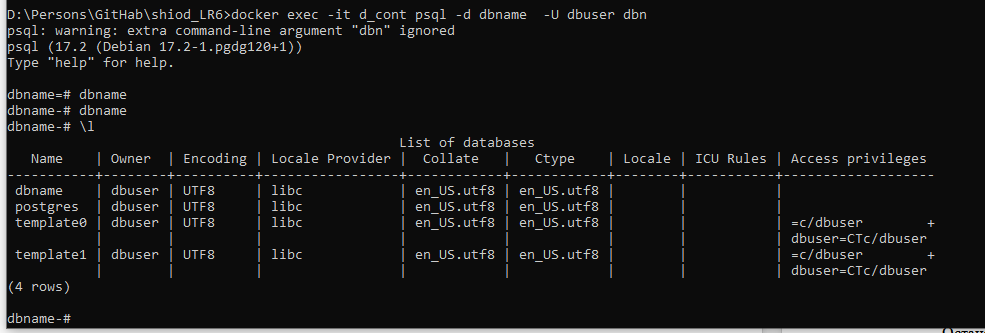
· флаг -it – для того чтобы работала интерактивная среда прямо в терминале

· test – название контейнера

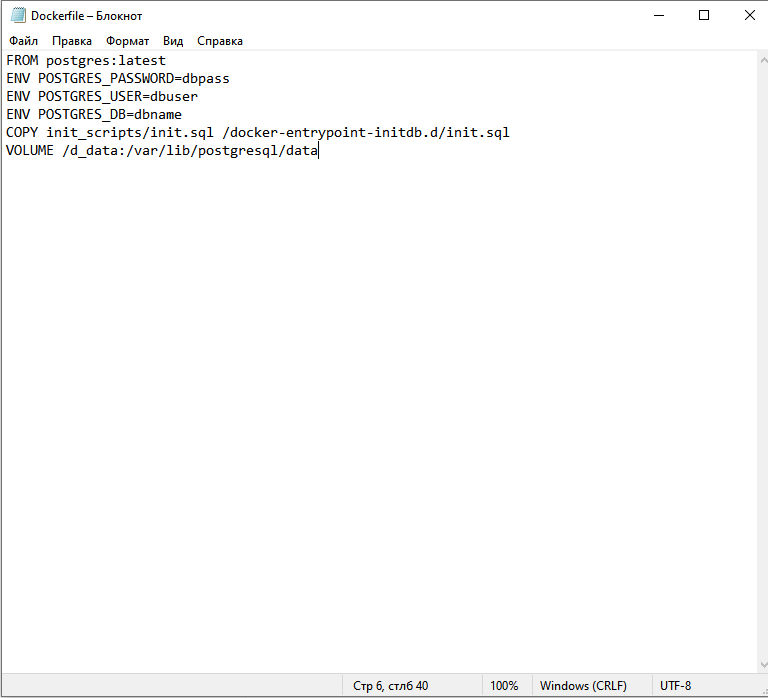
· psql – команда, которая говорит, что в нашей интерактивной среде (в командной строке) нужно запустить интерпретатор postgres

· -U – флаг указания имени пользователя (user)

-d – флаг указания наименования базы данных



1. Чтобы данные, измененные в процессе работы контейнера, были доступны после удаления контейнера (во вновь созданном контейнере) необходимо создать том (volume)



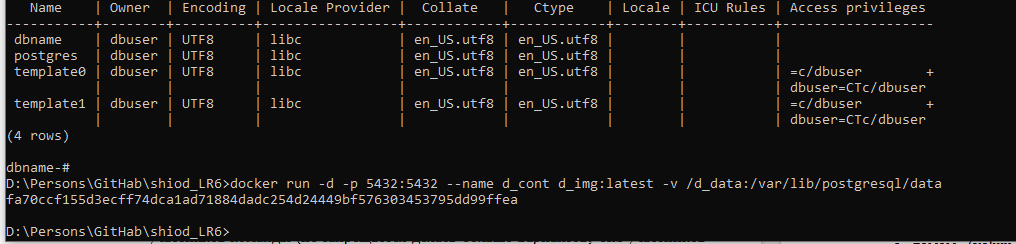
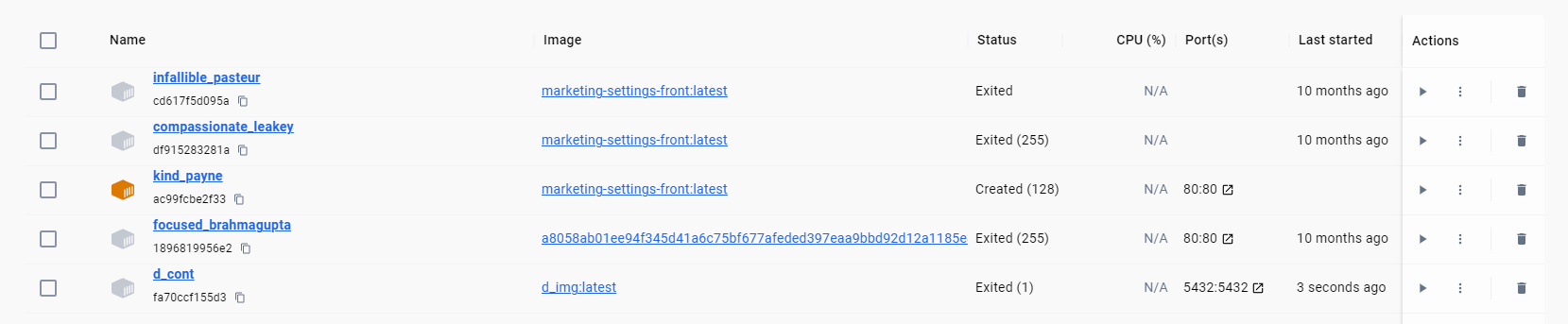
Volume подключается с помощью флага -v, за которым следую две папки (два места хранения) через двоеточие (/data:/var/lib/postgresql/data):

/data – это папка, которая находится в нашей операционной системе; можно указывать любую папку на компьютере;

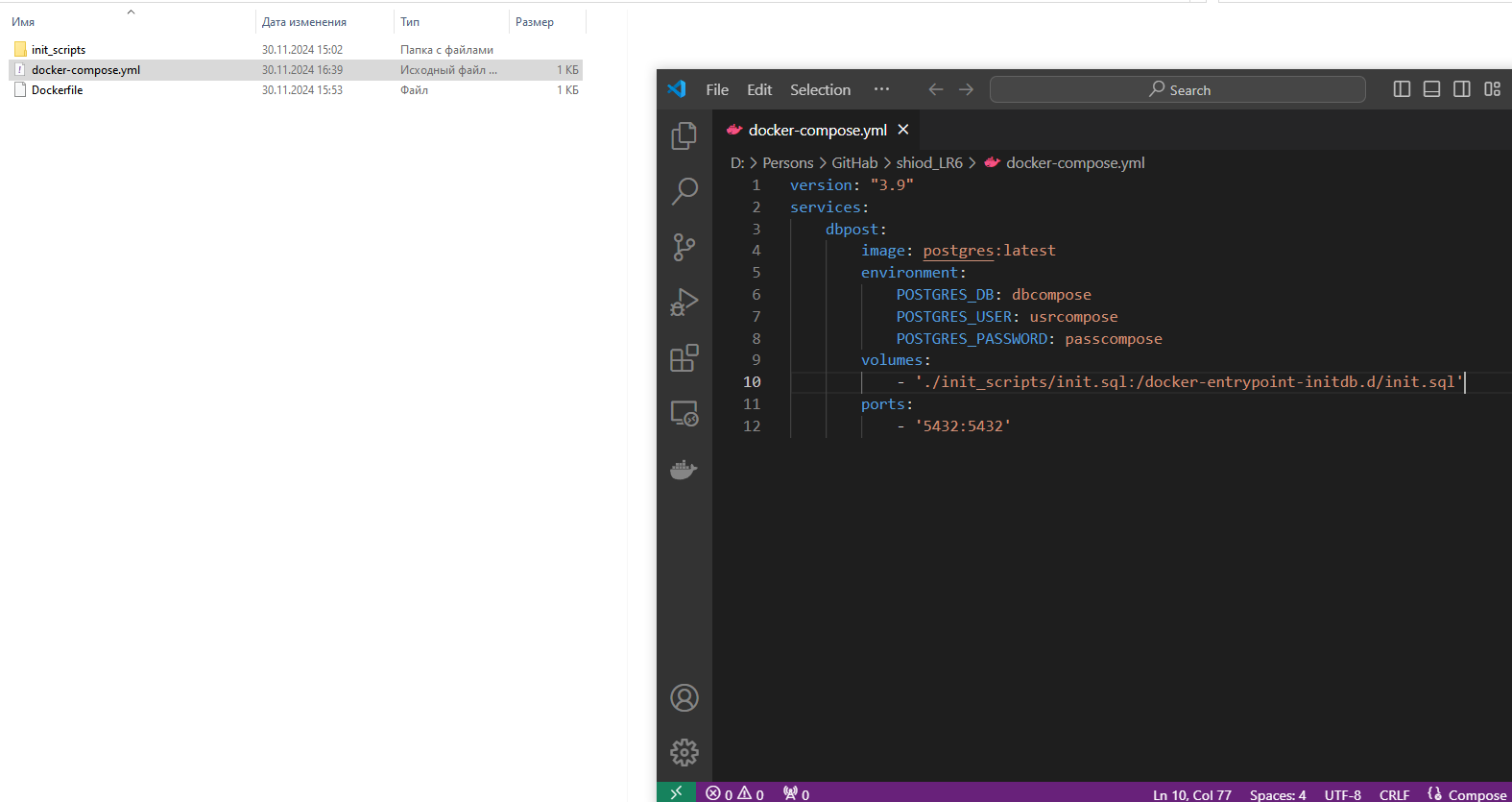
/var/lib/postgresql/data – это папка, которая находится внутри операционной системы контейнера; эта папка постоянная, она не меняется.

Между этими папками устанавливается синхронизация. То есть новые данные будут сохраняться в папку на компьютере (в нашем случае /data)

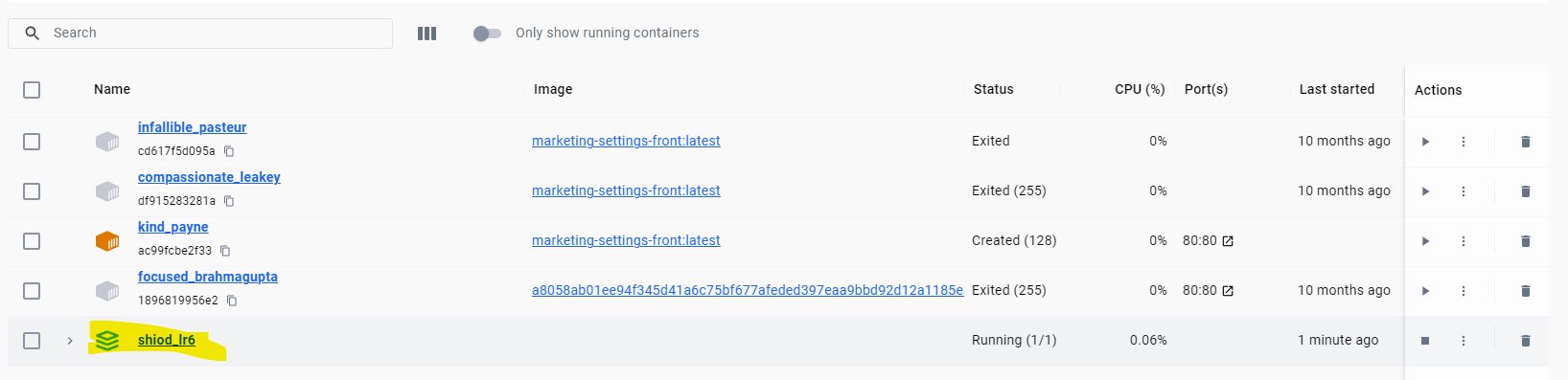
1. Удалить созданный контейнер. Создать контейнер с помощью команды: «docker run -d -p 5432:5432 --name d\_cont d\_img:latest -v /d\_data:/var/lib/postgresql/data»

  
Команда создает докер контейнер по инструкции Dockerfile на порту 5432, дает имя d\_cont на основе образа d\_img:latest, привязывая изменения с докер к volume.

1. Создать текстовый файл cmd.txt, в котором описать все команды, которые необходимо использовать для развертывания базы данных Postgres с помощью Dockerfile (создание образа, запуск контейнера, создание тома).
2. Создать контейнер с использованием файла docker-compose.yml



Для того, чтобы начать исполнение инструкций docker-compose нужно ввести команду docker-compose up --build



version: "3.9" // версия compose  
services: // перечисление сервисов, которые будут использоваться  
dbpost:   
image: postgres:latest //указание образа  
environment: // переменные окружения  
POSTGRES\_DB: dbcompose  
POSTGRES\_USER: usrcompose  
POSTGRES\_PASSWORD: passcompose  
volumes:./init\_scripts/init.sql:/docker-entrypoint-initdb.d/init.sql // куда будут записываться изменения  
ports: - 5432:5432 // порт, на котором работать  
  
Docker Compose используется для одновременного управления несколькими контейнерами, входящими в состав приложения.

# **Контрольные вопросы**

1. Что такое Docker?

Docker – это проект с открытым исходным кодом для автоматизации развертывания приложений в виде переносимых автономных контейнеров, выполняемых в облаке или локальной среде. Одновременно с этим, Docker — это компания, которая разрабатывает и продвигает эту технологию в сотрудничестве с поставщиками облачных служб, а также решений Linux и Windows, включая корпорацию Майкрософт

1. Зачем нужен Docker?

Docker нужен для разработки, доставки и запуска контейнерных приложений. Он позволяет создавать контейнеры, автоматизировать их запуск и развертывание, управляет жизненным циклом. Для избежания ошибок при развертывании программ.

1. Что такое docker-образ?

Docker-образ — это шаблон (физически — исполняемый пакет), из которого создаются Docker-контейнеры.

Образ хранит в себе всё необходимое для запуска приложения, помещенного в контейнер: код, среду выполнения, библиотеки, переменные окружения и конфигурационные файлы

1. Что такое docker-контейнер?

Docker-контейнер — это легкий, автономный исполняемый пакет программного обеспечения, который включает в себя всё необходимое для запуска приложения: код, среду выполнения, системные инструменты, системные библиотеки и настройки.

Как правило, в одном контейнере запускают одно приложение или даже отдельный его компонент — модуль, функцию или микросервис.

1. Что такое volume и зачем он нужен?

Docker Volume – это постоянное хранилище для контейнеров. Этот инструмент отключает привязку данных к жизненному циклу контейнера, позволяя получить доступ к контейнерным данным в любой момент. Таким образом, сделанные в контейнерах записи остаются доступными после уничтожения содержавшего их контейнера и могут повторно использоваться в других.

1. Что такое docker-compose?

Docker-compose — это надстройка над докером, приложение написанное на Python, которое позволяет запускать множество контейнеров одновременно и маршрутизировать потоки данных между ними.

1. В чем разница между dockerfile и docker-compose?

dockerfile управляет одним докер контейнером, docker-compose

1. Какая команда позволяет отправлять различные задания в запущенный докер-контейнер?

Команда docker exec позволяет отправлять различные задания в запущенный Docker-контейнер

1. С помощью какого инструмента можно сохранить важные данные после аварийного отключения контейнера?

Docker volumes, так как отключает привязку данных к жизненному циклу контейнера, позволяя получить доступ к контейнерным данным в любой момент. Таким образом, сделанные в контейнерах записи остаются доступными после уничтожения содержавшего их контейнера и могут повторно использоваться в других.