Министерство науки и высшего образования Российской Федерации Федеральное государственное автономное образовательное учреждение высшего образования «Уральский федеральный университет имени первого Президента России Б.Н. Ельцина» (УрФУ) Институт радиоэлектроники и информационных технологий – РтФ Базовая кафедра ”Автоматизация финансовых систем”

ОТЧЕТ

Лабораторная работа №1

Выполнила: Коляда Алена Владиславовна

Группа: РИМ-150950

Екатеринбург

Цель работы:

Освоить фундаментальные концепции и базовые операции Docker: создание образов, запуск контейнеров, управление ими, работа с сетями и томами. На практике закрепить навыки, запустив изолированную базу данных PostgreSQL и подключившись к ней извне.

Описание задачи:

становить и проверить работу Docker.

зучить базовые команды Docker.

апустить контейнер с PostgreSQL в изолированном режиме.

апустить контейнер с pgAdmin и подключить его к контейнеру с БД через сеть

одключиться к БД из pgAdmin, создать схему и выполнить запросы.

беспечить сохранность данных БД с помощью томов Docker.

Ход выполнения:

Установка и проверка работы Docker Desktop

Для достижения цели был установлен Docker Desktop для windows, затем была проведена проверка работы. Проверка представлена на рисунке 1.

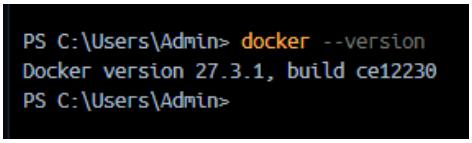


Рисунок 1 - Проверка установленного Docker Desktop

Изучение базовых команд

При изучении был запущен контейнер в фоне. Результат представлен на рисунке 2.

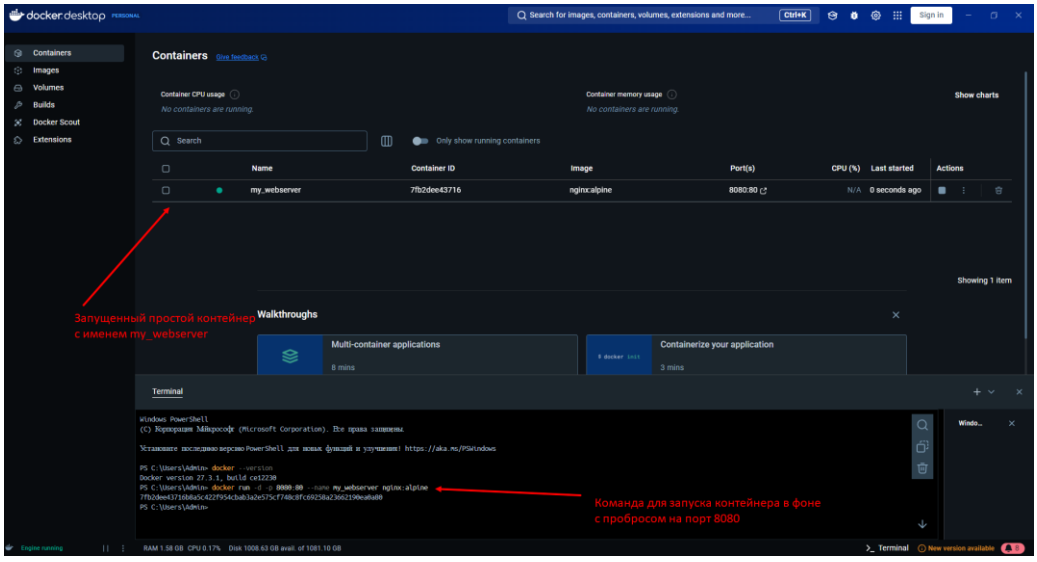


Рисунок 2 - Запуск простого контейнера

Для проверки того, что контейнер работает в браузере открыла: http://localhost:8080 и увидела стартовую страницу Nginx. Результат представлен на рисунке 3.

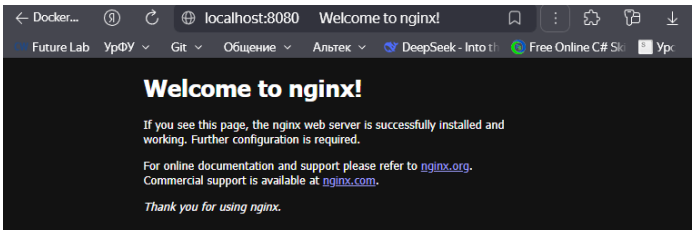


Рисунок 3 - Стартовая страница Nginx

Проверка завершена, останавливаем и удаляем контейнер за ненадобностью. Результат представлен на рисунках 4 - 5.

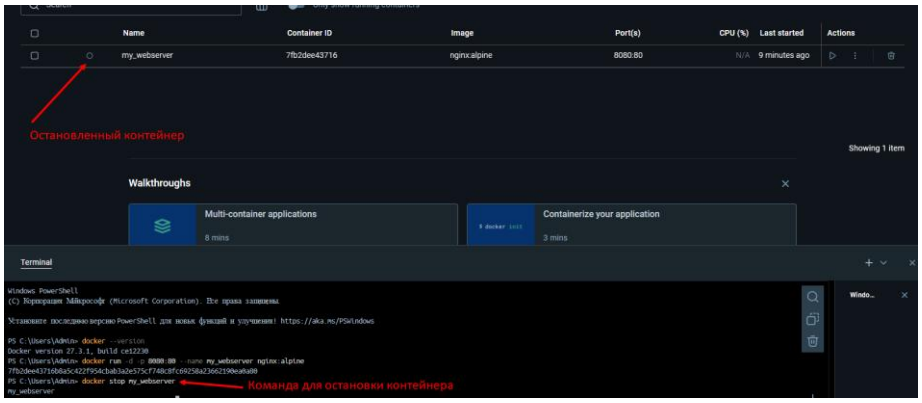


Рисунок 4 - Остановка контейнера

Рисунок 5 - Удаление контейнера

Запуск PostgreSQL в контейнере. Результат представлен на рисунке 6.

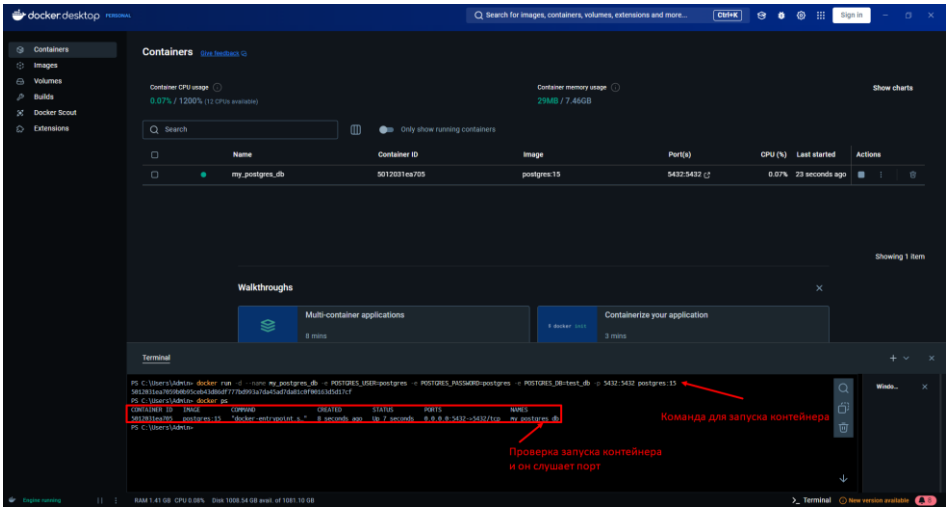


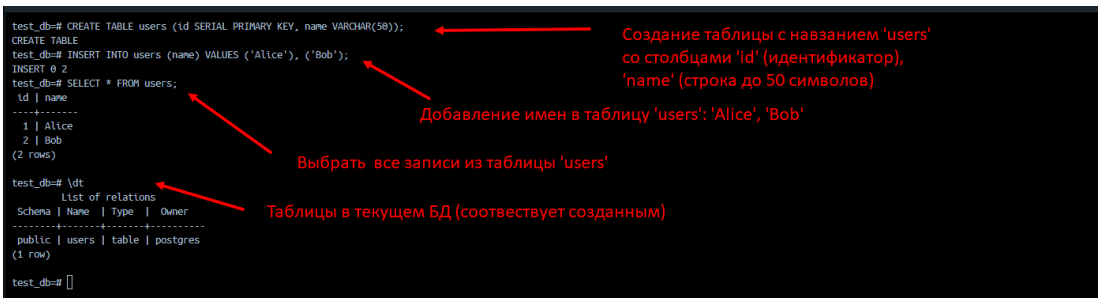
Рисунок 6 - Запуск контейнера

Подключение к БД из контейнера. Результат представлен на рисунках 7 - 8.

```
PS C:\Users\Admin> docker exec -it my_postgres_db psql -U postgres -d test_db
psql (15.10 (Debian 15.10-1.pgdg120+1))
Type "help" for help.

test_db=# \q
```

Рисунок 7 - Подключение к БД из контейнера



Р

Подключение к БД через pgAdmin из второго контейнера. Создание сети Docker. Результат представлен на рисунке 9. *с*

```
PS C:\Users\Admin> docker network create my_network
348b3f5f4a3252250bf6c3ad6724f446ea8cf9b3740a8068d5c7104e16f70aec
PS C:\Users\Admin> docker network ls
NETWORK ID     NAME         DRIVER    SCOPE
dcb58970455e   bridge       bridge    local
6de44eaf54db   host         host      local
348b3f5f4a32   my_network   bridge    local
8dd5312761d1   none         null      local
PS C:\Users\Admin>
```

Рисунок 9 - Создание сети

Подключение контейнера с PostgreSQL к сети. Результат представлен на рисунке 10.

```
PS C:\Users\Admin> docker network connect my_network my_postgres_db
```

Рисунок 10 - Подключение контейнера к сети

Запуск pgAdmin в той же сети, представлено на рисунке 11.

```
PS C:\Users\Admin> docker run -d --name my_pgadmin -e PGADMIN_DEFAULT_EMAIL=admin@example.com -e PGADMIN_DEFAULT_PASSWORD=admin -p 8080:80 --network my_network dpage/pgadmin4
Unable to find image 'dpage/pgadmin4:latest' locally
latest: Pulling from dpage/pgadmin4
8f9b27100efb: Download complete
ff0eff1c5d48: Download complete
28a8ddc2abd2: Download complete
5dbfdc003cfd: Download complete
7dc5f1c3188a: Download complete
4aa794a497e1: Download complete
133a0b74b0e0: Download complete
4c4d3d6a532b: Download complete
9f2fe31f1f30: Download complete
81cac31f42a7: Download complete
0eecab1d578f: Download complete
c2fc03c4d3e1: Download complete
1519b7b7923d: Download complete
2425492cad43: Download complete
Digest: sha256:d115bcd737940a6cfb61a54439d50de8b850e0782e2363102c9fa761f4022f49
Status: Downloaded newer image for dpage/pgadmin4:latest
8c37a5f6218d5fec55a95c058ea50770f184e7522372ea4049edde7467f392af
```

Рисунок 11 - Запуск pgAdmin в сети my_network

Было настроено подключение в pgAdmin. Добавлен новый сервер. Выполнен запрос в pgAdmin. Результат представлен на рисунке 12.

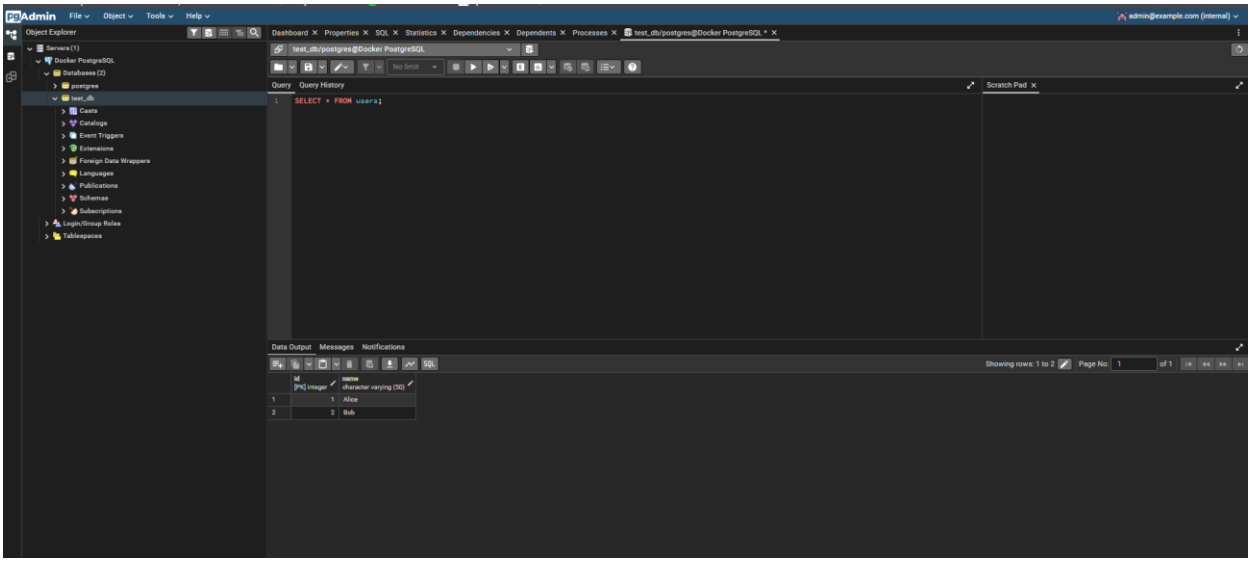


Рисунок 12 - Запрос в pgAdmin, в котором видна таблица users, созданная через консоль и данные в ней

Запуск нового контейнера с подключенным томом, представлен на рисунке 13.

Рисунок 13 - Запуск нового контейнера

Изначально контейнер с PostgreSQL был пустым. После создания таблицы и записей через pgAdmin, все эти данные были записаны в том Docker, который подключен к контейнеру. Теперь данные хранятся независимо от контейнера, поэтому при его удалении или перезапуске информация сохранится в томе и будет доступна новому контейнеру при подключении к этому же тому.

Чтобы доказать, что данные сохраняются в томе, а не в контейнере, я создам новый контейнер с именем my_postgres_db и подключу его к тому же тому, что использовался для контейнера my_postgres_db_persistent. После этого проверю, что все таблицы и записи остались нетронутыми. Результаты показаны на рисунках 14-15.

Рисунок 14 - Создание нового контейнера с подключенным томом

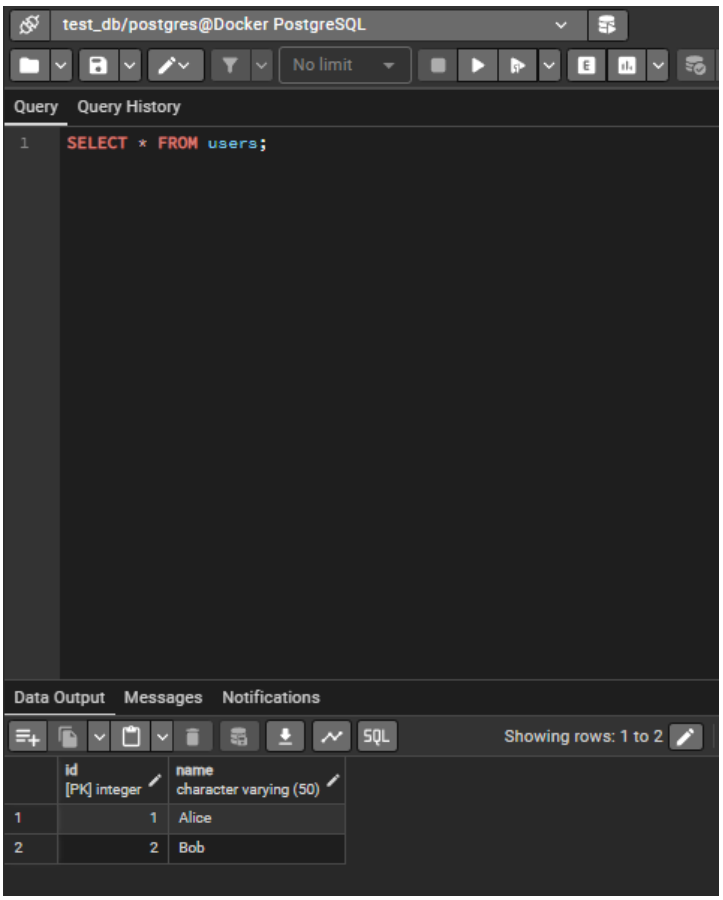


Рисунок 15 - Данные из таблицы users

еренос конфигурации контейнеров в docker-compose.yaml

Файл конфигурации docker-compose.yaml, используемый для развёртывания сервиса базы данных PostgreSQL и инструмента администрирования pgAdmin, представлен в листинге 1.

Листинг 1 – Конфигурационный файл docker-compose.yaml

```yaml
version: '3.8'

services:
  db:
    image: postgres:15
    container_name: my_postgres_db
    environment:
      POSTGRES_USER: postgres
      POSTGRES_PASSWORD: postgres
      POSTGRES_DB: test_db
    volumes:
      - postgres_data:/var/lib/postgresql/data
    networks:
      - my_network
    ports:
      - "5432:5432"
```

```yaml
  pgadmin:
    image: dpage/pgadmin4
    container_name: my_pgadmin
    environment:
      PGADMIN_DEFAULT_EMAIL: admin@example.com
      PGADMIN_DEFAULT_PASSWORD: admin
    ports:
      - "8080:80"
    depends_on:
      - db
    networks:
      - my_network

volumes:
  postgres_data:


networks:
  my_network:
```

После создания конфигурационного файла был проведён эксперимент: запущена среда с помощью docker-compose, произведена настройка сервера и создана таблица users с тестовыми записями. Результат проделанной работы показан на рисунке 16.

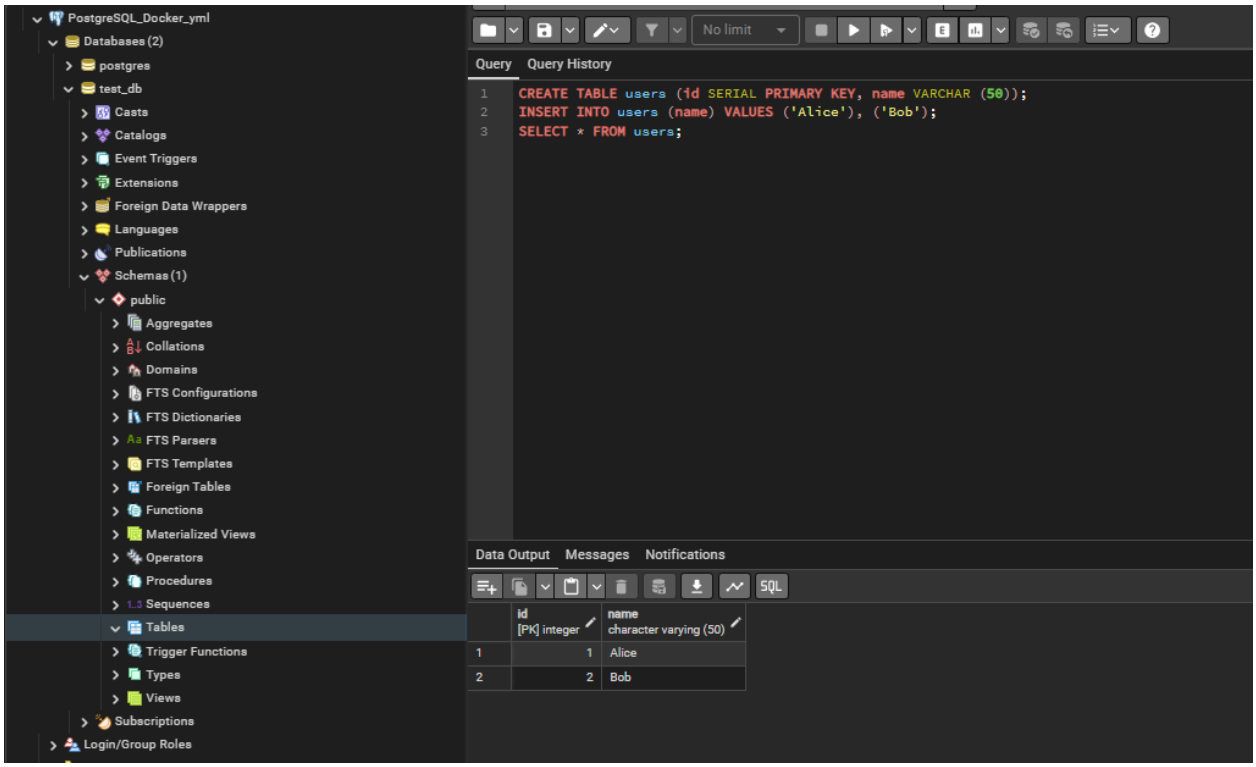


Рисунок 16 - Результат создания таблицы users

Далее среда выполнения (контейнер lr1) была удалена, после чего эксперимент повторён. При повторном запуске сразу отобразились ранее добавленные записи в таблице

, что подтверждает корректное сохранение данных в выделенном томе postgres_data. Результат представлен на рисунке 17.

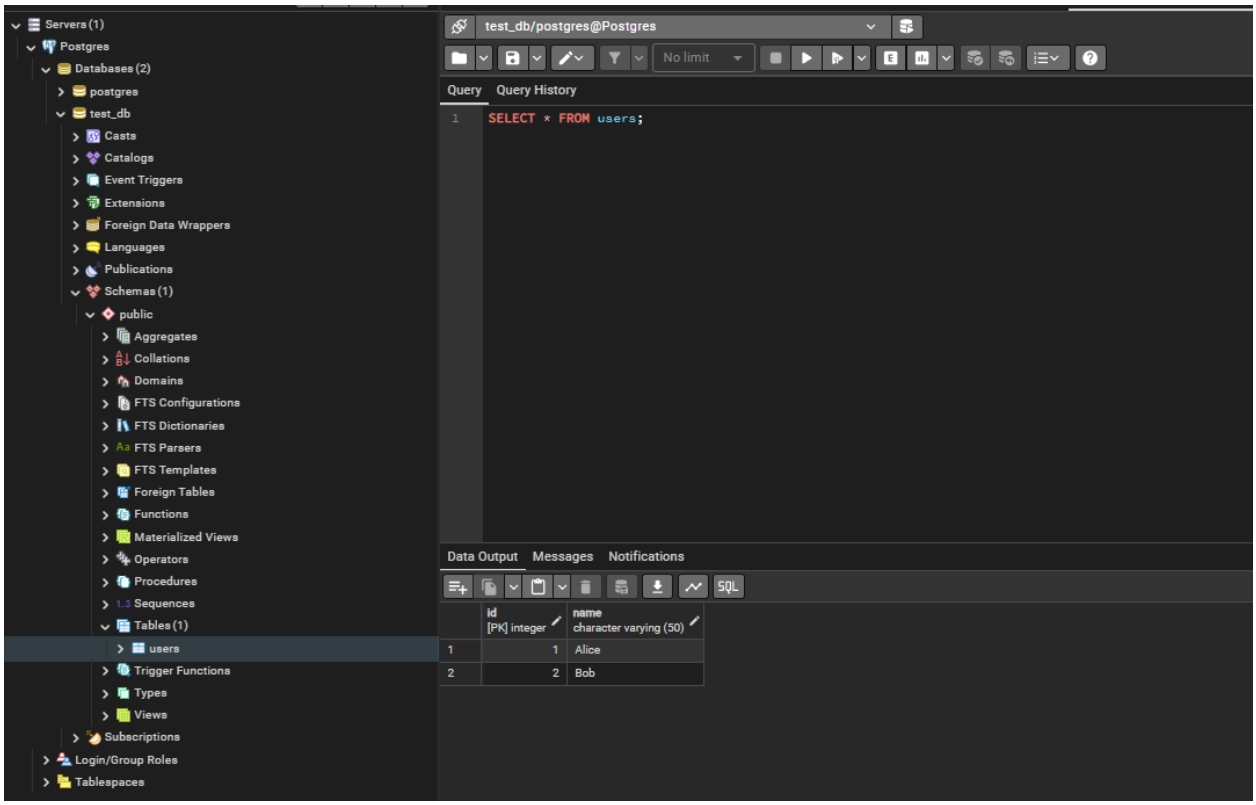


Рисунок 17 - Данные из таблицы users

Ответы на вопросы:

1. Что такое docker?

D

o

c

k

er – это платформа для разработки, тестирования и развертывания приложений в средах с поддержкой контейнеризации. Позволяет упаковывать приложение со всеми его зависимостями в контейнер, а затем запускать его в системе.

2. Для чего нужны тома и сети docker?

Тома нужны для долговременного хранения данных контейнеров. Данные не пропадают при остановке или удалении контейнера.

Сеть нужна для связи между контейнерами, чтобы они могли взаимодействовать друг с другом. Создаётся изолированная сеть, в которой контейнеры видят друг друга по именам.

3. Как подключиться к контейнеру, находящемуся в состоянии running?

Используется команда: docker exec -it <container_name> bash

или, если внутри контейнера нет bash:

docker exec -it <container_name> sh

После этого открывается интерактивная оболочка, и можно выполнять команды внутри контейнера.

4. Для чего нужен pgAdmin?

это графический инструмент для управления PostgreSQL. Он облегчает работу с базой данных: позволяет создавать таблицы, выполнять SQL-запросы, управлять пользователями и схемами, отслеживать состояние сервера через удобный интерфейс, без необходимости писать все команды вручную в консоли.

Ссылка на github:

Вывод:

В ходе работы были освоены базовые принципы работы с Docker: установка и запуск контейнеров, взаимодействие сервисов через сеть и использование томов для обеспечения сохранности данных. На практике были развернуты два контейнера - PostgreSQL и pgAdmin - объединённые в общую сеть. С помощью pgAdmin выполнено подключение к базе данных, создана схема и добавлены записи.

Повторный запуск среды показал сохранность данных в томе, что подтверждает корректность использования механизма постоянного хранилища Docker. Таким образом, цели работы достигнуты: получены практические навыки работы с образами, контейнерами, сетями и томами в Docker, а также выполнена настройка полноценной среды для работы с базой данных PostgreSQL.