МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ

ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

ВЫСШЕГО ОБРАЗОВАНИЯ

«ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

#### Факультет информационных технологий и компьютерной безопасности

### Кафедра Компьютерных интеллектуальных технологий проектирования

# ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 6

по дисциплине «Системы хранения и обработки данных»

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

Разработал студент группы мИИВТ-241 Давлетшин К.В.

Подпись, дата Инициалы, фамилия

## Принял Короленко В.В.

Подпись, дата Инициалы, фамилия

Защищена \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Оценка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

дата

2024

Цель:

изучить основы работы программного средства для автоматизации развёртывания и управления приложениями в средах с поддержкой контейнеризации Docker на примере решения задачи развёртывания СУБД Postgres.

Ход лабораторной работы:

Первым шагом в ходе выполнения данной лабораторной работы стало установка docker-engine, для чего я воспользовался [инструкцией](https://www.kali.org/docs/containers/installing-docker-on-kali/). После чего скачал и установил Docker Desktop.

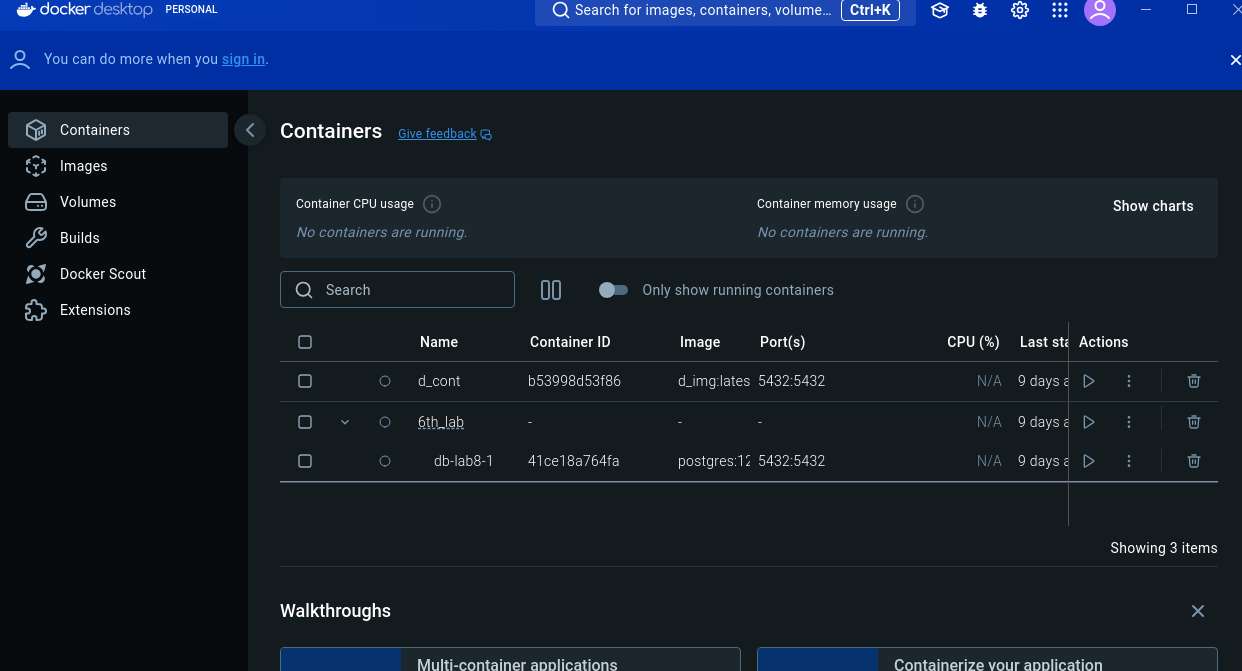


Рисунок 1 – Docker Desktop

Для работы с базой данных была установлена IntelliJ IDEA, которая предоставляет удобный интерфейс для удобного и быстрого взаимодействия с различными СУБД.

Следующим шагом стало скачивание docker image postgres:12.20, что было реализовано командой docker pull postgres:12.20

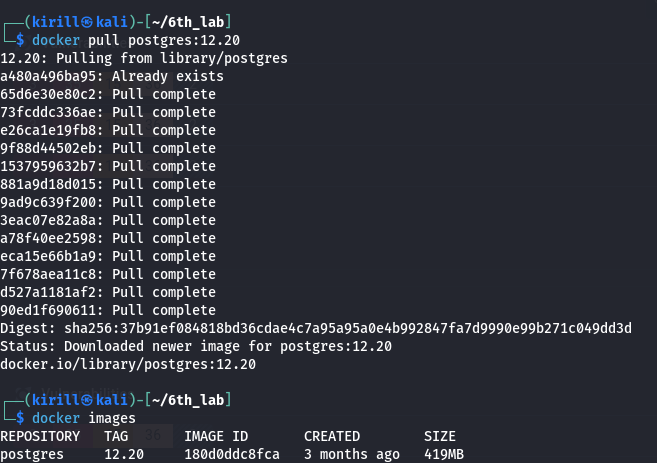


Рисунок 2 – Скачивание docker image postgres:12.20

Следующим шагом стало создание Dockerfile. Для Создания Dockerfile необходимо понимать что это такое и по средством каких команд можно создавать Dockerfile. Dockerfile – это текстовый файл, который содержит набор инструкций для создания, настройки и конфигурации Docker Image. В первой строке Dockerfile указывается какой image будет использоваться в качестве основы. В случае отсутствия необходимого образа, он будет скачен из DockerHub. Команда ENV устанавливает переменные окружения для пользователя, пароли и таблицы БД. В последней строке копируется init.sql из локальной директории init\_scripts в директорию /docker-entrypoint-initdb.d/ внутри контейнера. Все SQL-скрипты, помещенные в эту директорию, будут автоматически выполнены при первом запуске контейнера, если база данных еще не была инициализирована. Команда VOLUME /d\_data:/var/lib/postgresql/data в Dockerfile (или в Docker-командах) используется для создания точки монтирования, которая связывает директорию на хосте с директорией внутри контейнера.

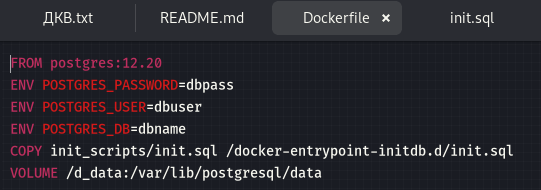


Рисунок 3 – Создание DockerFile

Затем необходимо создать init.sql, где описать скрипт создания таблицы и заполнения ее данными. CREATE TABLE IF NOT EXISTS создает таблицу, если она еще не создана, INSERT INTO <table name>() VALUES вставляет данные в таблицу.

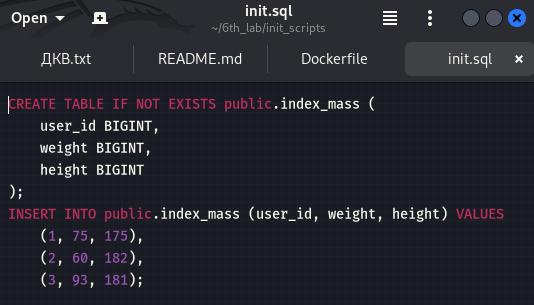


Рисунок 4 – Создание init.sql

Следующим шагом стало создание Docker Image, при помощи команды docker build –t d\_img:latest.

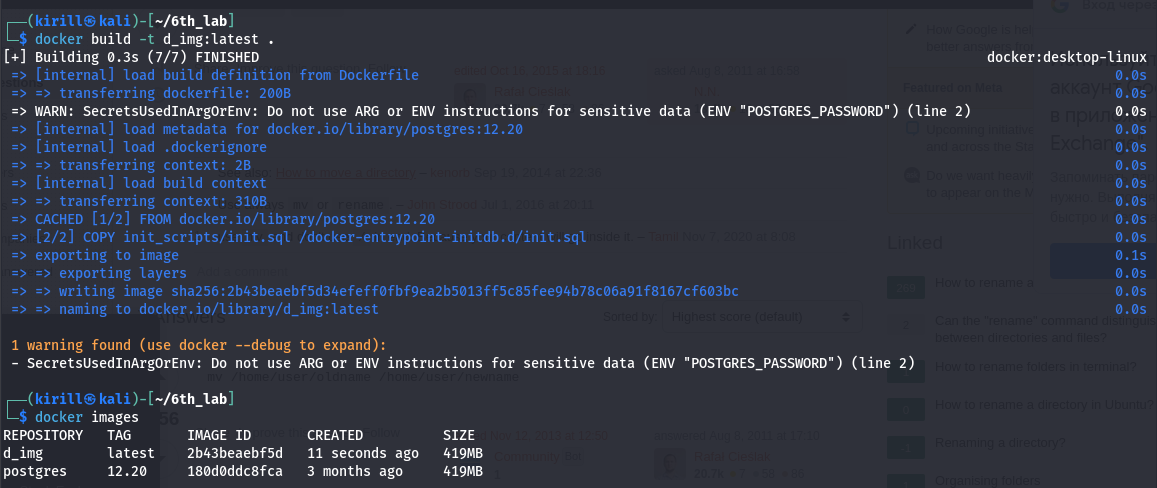


Рисунок 5 – Создание docker image

Далее необходимо запустить созданный контейнер при помощи команды docker run –d –p 5432:5432 –name d\_cont d\_img:latest. После чего посмотреть, были ли созданы таблицы и были ли заполнены данные.

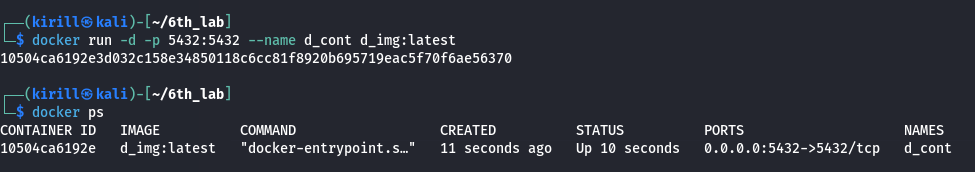


Рисунок 6 – запуск Docker Container

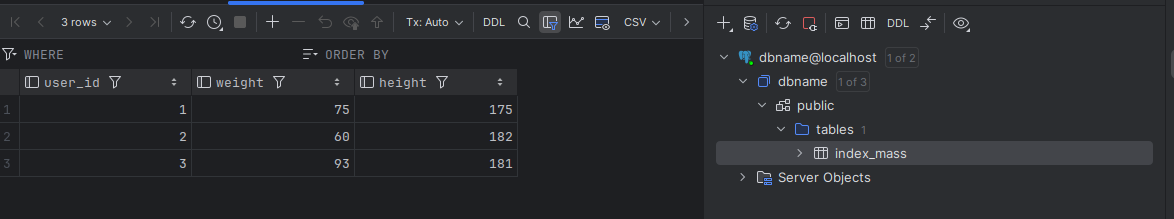


Рисунок 7 – Демонстрация работы Docker Container

Следующим шагом стало подключение к работающему контейнеру, запуск интерфейс psql и внесение новых данных «на лету».

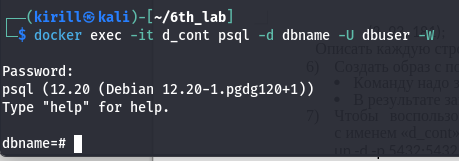


Рисунок 8 – Подключение к запущенному контейнеру

Далее необходимо было отредактировать image, остановить, удалить и создать заново контейнер, что было сделано при помощи команд docker stop <container id>, docker rm <container\_name>, и уже знакомая команда запуска docker run –d –p 5432:5432 –name d\_cont d\_img:latest.

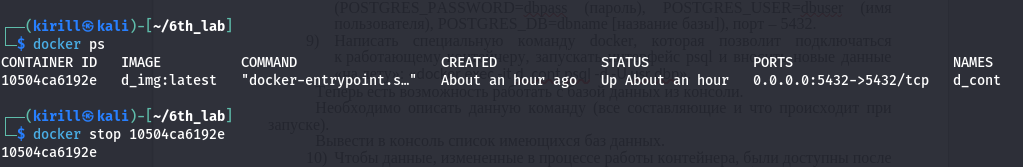


Рисунок 9 – Остановка работающего Контейнера

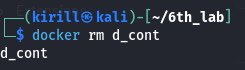


Рисунок 10 – Удаление контейнера

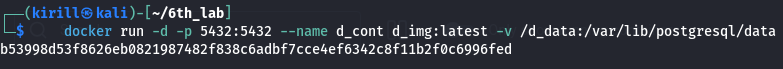


Рисунок 11 – Запуск контейнера с новым Docker Image

Следующим этапом стало создание docker-compose.yaml файла. version: Указывает версию формата файла Docker Compose. В данном случае используется версия 3.9, которая поддерживает различные функции и возможности, доступные в Docker Compose. services: Определяет секцию, в которой перечисляются все сервисы (контейнеры), которые будут запущены. Каждый сервис будет работать в своем собственном контейнере. Dbpost – имя сервиса. Это имя будет использоваться для обращения к контейнеру в рамках Docker Compose. image: Указывает, какой образ Docker будет использоваться для создания контейнера. environment: Определяет переменные окружения, которые будут переданы в контейнер. volumes: Определяет тома, которые будут смонтированы в контейнер. Это позволяет сохранять данные между перезапусками контейнера и использовать локальные файлы. ports: Определяет, какие порты на хост-машине будут перенаправлены на порты контейнера. Это позволяет получить доступ к сервису из внешней сети.

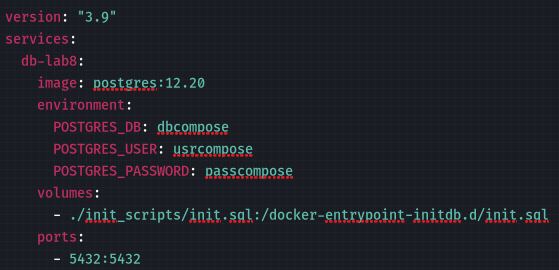


Рисунок 12 – Создание docker-compose.yaml

Для запуска контейнера необходимо выполнить команду docker compose up –d, после чего будут скачены все необходимые images, созданы и запущены все необходимые контейнеры, после чего результат можно посмотреть в терминале при помощи команды docker ps или в графическом интерфейсе Docker Dektop.

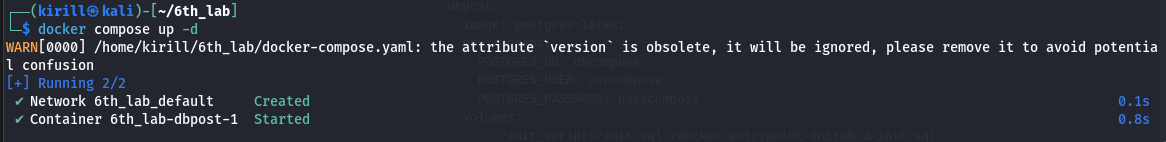


Рисунок 13 – Запуск docker-compose.yaml

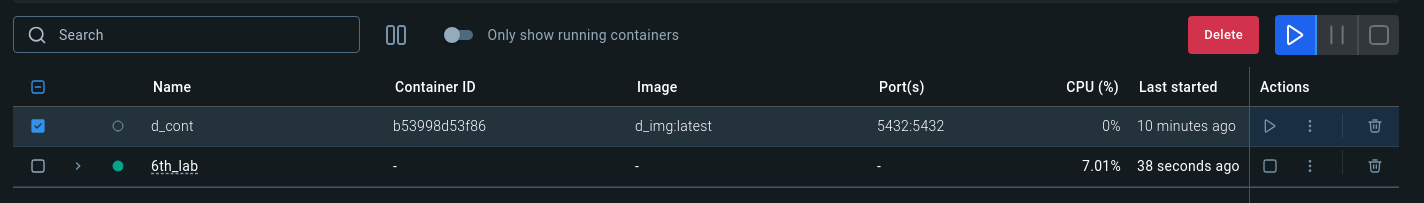


Рисунок 14 – Демонстрация работы docker-compose.yaml

Вывод: в результате выполнения лабораторной работы были изучены и использованы на практике базовые команды Docker, созданы Docker images и Docker containers, а также был создан Docker container при помощи docker-compose.yaml.