**Задание 2.**

**Создание Docker-контейнера с PostgreSQL и Python-приложением**

**Цель: Научиться создавать Docker-контейнеры с базой данных PostgreSQL и Python-приложением для работы с ней. Использовать Docker Compose для координации нескольких контейнеров.**

Описание задания:

* Создайте директорию для проекта и необходимые файлы.
* В файле app.py создайте скрипт, который будет выполнять следующие задачи:

1. Подключение к базе данных PostgreSQL (обязательно только через psycopg2).
2. Создание таблицы.
3. Наполнение таблицы данными.
4. Вывод данных из таблицы.

* В Dockerfile опишите шаги для создания Docker-образа Python-приложения, которое будет работать с PostgreSQL.
* Версия postgreSQL — 13.
* Создайте docker-compose.yml, чтобы оркестрировать контейнеры с PostgreSQL и Python-приложением.
* Теперь, когда все файлы готовы, можно запустить приложение с помощью Docker Compose.
* После выполнения команды docker-compose up Вы увидите в логах вывод от Python-приложения, который должен включать данные из таблицы employees.

################## app.py ##############################################

import psycopg2

# Подключение к базе данных

try:

conn = psycopg2.connect(

dbname="testdb",

user="postgres",

password="postgres",

host="db", # Имя сервиса из docker-compose.yml

port="5432"

)

cursor = conn.cursor()

# Создание таблицы

cursor.execute("""

CREATE TABLE IF NOT EXISTS employees (

id SERIAL PRIMARY KEY,

name VARCHAR(100),

age INT,

department VARCHAR(100)

);

""")

# Наполнение таблицы данными

cursor.execute("INSERT INTO employees (name, age, department) VALUES (%s, %s, %s);",

("Alice", 30, "Engineering"))

cursor.execute("INSERT INTO employees (name, age, department) VALUES (%s, %s, %s);",

("Bob", 25, "Marketing"))

# Сохранение изменений

conn.commit()

# Вывод данных из таблицы

cursor.execute("SELECT \* FROM employees;")

rows = cursor.fetchall()

for row in rows:

print(row)

except Exception as e:

print(f"Ошибка подключения к базе данных: {e}")

finally:

if conn:

cursor.close()

conn.close()

########################################################################

################## Dockerfile ##########################################

# Используем базовый образ Python

FROM python:3.9-slim

# Устанавливаем зависимости

WORKDIR /app

COPY requirements.txt .

RUN pip install --no-cache-dir psycopg2

# Копируем код приложения

COPY app.py .

# Запускаем приложение

CMD ["python", "app.py"]

########################################################################

################## docker-compose.yml ##################################

version: '3.8'

services:

  db:

    networks:

     - pg-network

    image: postgres:13

    container\_name: postgres\_db

    environment:

      POSTGRES\_USER: postgres

      POSTGRES\_PASSWORD: postgres

      POSTGRES\_DB: testdb

    ports:

      - "5433:5432"

    volumes:

      - db\_data:/var/lib/postgresql/data

    healthcheck:

      test: ["CMD-SHELL", "pg\_isready -U postgres"]

      interval: 1s

      timeout: 5s

      retries: 10

  app:

    networks:

     - pg-network

    build:

      context: .

    depends\_on:

      db:

        condition: service\_healthy

    container\_name: python\_app\_it2

volumes:

  db\_data:

networks:

  pg-network:

########################################################################

docker-compose up --build