

Департамент образования и науки города Москвы  
Государственное автономное образовательное учреждение  
высшего образования города Москвы  
«Московский городской педагогический университет»  
Институт цифрового образования  
Департамент информатики управления и технологий

Ли Александр Андреевич БД-241м

**Практическая работа 3-2. Docker-compose**

Направление подготовки/специальность  
38.04.05 - Бизнес-информатика  
Бизнес-аналитика и большие данные  
(очная форма обучения)  
Вариант 12

Москва

2024

## Введение

Практическая работа нацелена на знакомство студентов с основами работы в Linux, установку системы, проведение предварительной настройки системы и настройка SSH на Ubuntu 24.

## Цель

Целью практической работы заключается в создании проекта с использованием Docker-compose для взаимодействия с несколькими контейнерами, что позволяет создать среду, в которой три отдельных сервиса (init, app, db) взаимодействуют друг с другом, что упрощает их развертывание, настройку и управление.

Init – используется для обновления базы данных до последней версии.

Db – база данных

App – создает api приложение

## Взаимодействие

1. Сначала запускается база данных
2. После того, как запустилась bd, активируется init, задачей которого является обновление и настройка структуры базы данных, чтобы она была готова для использования основным приложением.
3. После всего вышесказанного, запускается app, который взаимодействует с базой данных, получая запрос от пользователей и разворачивает страницу через fast api.

1. bd → init (настраивает db) → app (взаимодействует с db)

## 2. Конфигурационные файлы:

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

```
version: '3'
```

```
services:
```

```
  init:
```

```
    container_name: init
```

```
  build:
```

```
    context: ./server
```

```
    dockerfile: Dockerfile
```

```
  env_file: .env
```

environment:

- PYTHONUNBUFFERED=0
- PYTHONPATH=/app/server
- ENVIRONMENT=production

command:

- sh
- -c
- 'alembic upgrade head'

depends\_on:

- db

server:

container\_name: server

build:

context: ./server

dockerfile: Dockerfile

env\_file: .env

environment:

- PYTHONUNBUFFERED=0
- PYTHONPATH=/app/server
- ENVIRONMENT=production
- SERVER\_PORT=\${SERVER\_PORT}
- CLIENT\_PORT=\${CLIENT\_PORT}

restart: always

command:

- sh
- -c
- 'uvicorn server.src.main:app --host 0.0.0.0 --port \${SERVER\_PORT:-8000}'

ports:

- '8000:\${SERVER\_PORT:-8000}'

volumes:

- ./server:/app/server

depends\_on:

- db
- init

healthcheck:

```
test: ["CMD-SHELL", "curl -f http://localhost:${SERVER_PORT:-8000}/health || exit 1"]
interval: 30s
timeout: 10s
retries: 3
networks:
  - my_network
```

db:

```
container_name: db
image: postgres
restart: always
ports:
  - '5432:5432'
env_file: .env
healthcheck:
  test: ["CMD-SHELL", "pg_isready -U ${POSTGRES_USER:-postgres}"]
  interval: 30s
  timeout: 10s
  retries: 5
networks:
  - my_network
```

networks:

```
my_network:
  driver: bridge
```

## **.env.**

```
APP_PORT=8080
DB_USER=user
DB_PASSWORD=pass
POSTGRES_PASSWORD=password
ACCESS_TOKEN_EXPIRE_MINUTES=15
REFRESH_TOKEN_EXPIRE_DAYS=7
SECRET_KEY=your_secret_key
POSTGRES_USER=postgres
POSTGRES_DB=your_database
POSTGRES_HOST=db
POSTGRES_PORT=5432
SERVER_PORT=8000
CLIENT_PORT=3000
```

### 3. Настройка Docker Compose.

#### 1. Описание сервисов.

- init предназначен для подготовки работы базы данных. Выполняет миграцию командой `alembic upgrade head`, чтоб структура базы была актуальной
- server основное приложение , которое разворачивает страницу через fast api и взаимодействие с базой данных
- db управляет хранилищем данных на основе образа postgres

#### 2. Жесткое именованние контейнеров.

Каждый контейнер имеет жесткое именованние , заданное с помощью **container\_name**

```
container_name: db
container_name: server
container_name: init
```

#### 3. Применение depends\_on, volume, прокидка порта наружу, command и/или entrypoint.

-depends\_on

Так как init и server зависят от db, а server от init, необходимо использовать depends\_on, чтобы один сервис должен запускаться после другого. Сделаем что, db запускалась перед init, а init завершал работу перед запуском server.

- volume

Используется для того, чтобы данные из локальной машины были доступны внутри контейнера. Volume монтирует локальную папку /server в контейнер server. Это удобно, так как любые изменения файлов в локальной папке сразу видны внутри контейнера без необходимости пересборки.

-прокидка порта наружу

В контейнере server пробрасываем порт наружу ( ' 8000:\${SERVER\_PORT:-8000}' ), для того чтобы, подключаться к серверу извне, используя порт 8000.

-command

В init с помощью команды `alembic upgrade head` выполняется миграция базы данных,а для server запускает выполнения приложения на FastAPI с помощью `uvicorn`

#### 4. Описание healthcheck и сети.

Healthcheck используется в server для того, чтобы проверять доступность сервера по URL

`http://localhost:${SERVER_PORT:-8000}/health`.

Все сервисы подключены к сети `my_network`, что дает возможность сервисам взаимодействовать друг с другом.

#### 4. Процесс запуска.

Для запуска используется команда

##### **Sudo docker-compose up --build**

docker-compose запускает сервисы в порядке, указанном в docker-compose.yml. Сначала запускаются db, затем init, и server.

```
dba@dba-vm:~/Desktop/lab3_2$ sudo docker-compose up
[sudo] password for dba:
Starting db ... done
Starting init ... done
Starting server ... done
Attaching to db, init, server
db      |
db      | PostgreSQL Database directory appears to contain a database; Skipping initialization
db      |
db      | 2024-11-08 21:09:50.013 UTC [1] LOG:  starting PostgreSQL 17.0 (Debian 17.0-1.pgdg120+1) on x86_64-pc-linux-gnu, compiled by gcc (Debian 12.2.0-14) 12.2.0, 64-bit
db      | 2024-11-08 21:09:50.013 UTC [1] LOG:  listening on IPv4 address "0.0.0.0", port 5432
db      | 2024-11-08 21:09:50.013 UTC [1] LOG:  listening on IPv6 address "::", port 5432
db      | 2024-11-08 21:09:50.062 UTC [1] LOG:  listening on Unix socket "/var/run/postgresql/.s.PGSQL.5432"
db      | 2024-11-08 21:09:50.146 UTC [28] LOG:  database system was interrupted; last known up at 2024-11-08 20:46:53 UTC
init    | Traceback (most recent call last):
init    |   File "/usr/local/lib/python3.10/site-packages/sqlalchemy/engine/bas
se.py", line 145, in __init__
```

Рис.1 Запуск docker-compose

```
init    | sqlalchemy.exc.OperationalError: (psycopg2.OperationalError) could not translate host name "db" to address: Temporary failure in name resolution
init    |
init    | (Background on this error at: https://sqlalche.me/e/20/e3q8)
db      | 2024-11-08 21:09:50.731 UTC [28] LOG:  database system was not properly shut down; automatic recovery in progress
db      | 2024-11-08 21:09:50.733 UTC [28] LOG:  redo starts at 0/194C188
db      | 2024-11-08 21:09:50.733 UTC [28] LOG:  invalid record length at 0/194C1C0: expected at least 24, got 0
db      | 2024-11-08 21:09:50.733 UTC [28] LOG:  redo done at 0/194C188 system usage: CPU: user: 0.00 s, system: 0.00 s, elapsed: 0.00 s
db      | 2024-11-08 21:09:50.739 UTC [26] LOG:  checkpoint starting: end-of-recovery immediate wait
db      | 2024-11-08 21:09:50.755 UTC [26] LOG:  checkpoint complete: wrote 3 buffers (0.0%); 0 WAL file(s) added, 0 removed, 0 recycled; write=0.005 s, sync=0.002 s, total=0.018 s; sync files=2, longest=0.001 s, average=0.001 s; distance=0 kB, estimate=0 kB; lsn=0/194C1C0, redo lsn=0/194C1C0
db      | 2024-11-08 21:09:50.762 UTC [1] LOG:  database system is ready to accept connections
server  | INFO:      Started server process [6]
server  | INFO:      Waiting for application startup.
server  | INFO:      Application startup complete.
server  | INFO:      Uvicorn running on http://0.0.0.0:8000 (Press CTRL+C to quit)
```

Рис.2 Запуск docker-compose

#### 5. Результат работы.

The terminal window shows the following commands and output:

```
dba@dba-vm: ~/Desktop/lab3_2$ ls
client      docs        nginx       requirements.txt
docker-compose.yml  __init__.py  README.md  server
dba@dba-vm: ~/Desktop/lab3_2$ docker ps
permission denied while trying to connect to the Docker daemon socket at unix:///var/run/docker.sock: Get "http://%2Fvar%2Frun%2Fdocker.sock/v1.47/containers/json": dial unix /var/run/docker.sock: connect: permission denied
dba@dba-vm: ~/Desktop/lab3_2$ sudo docker ps
[sudo] password for dba:
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS              PORTS
950759967e02   lab3_2_server  "sh -c 'uvicorn serv..." 6 minutes ago  Up 6 minutes (healthy)  0.0.0.0:8000->8000/tcp, :::8000->8000/tcp
66fa1fa37abf   postgres       "docker-entrypoint.s..." 6 minutes ago  Up 6 minutes (healthy)  0.0.0.0:5432->5432/tcp, :::5432->5432/tcp
```

Рис.3 Проверка работы контейнеров командой

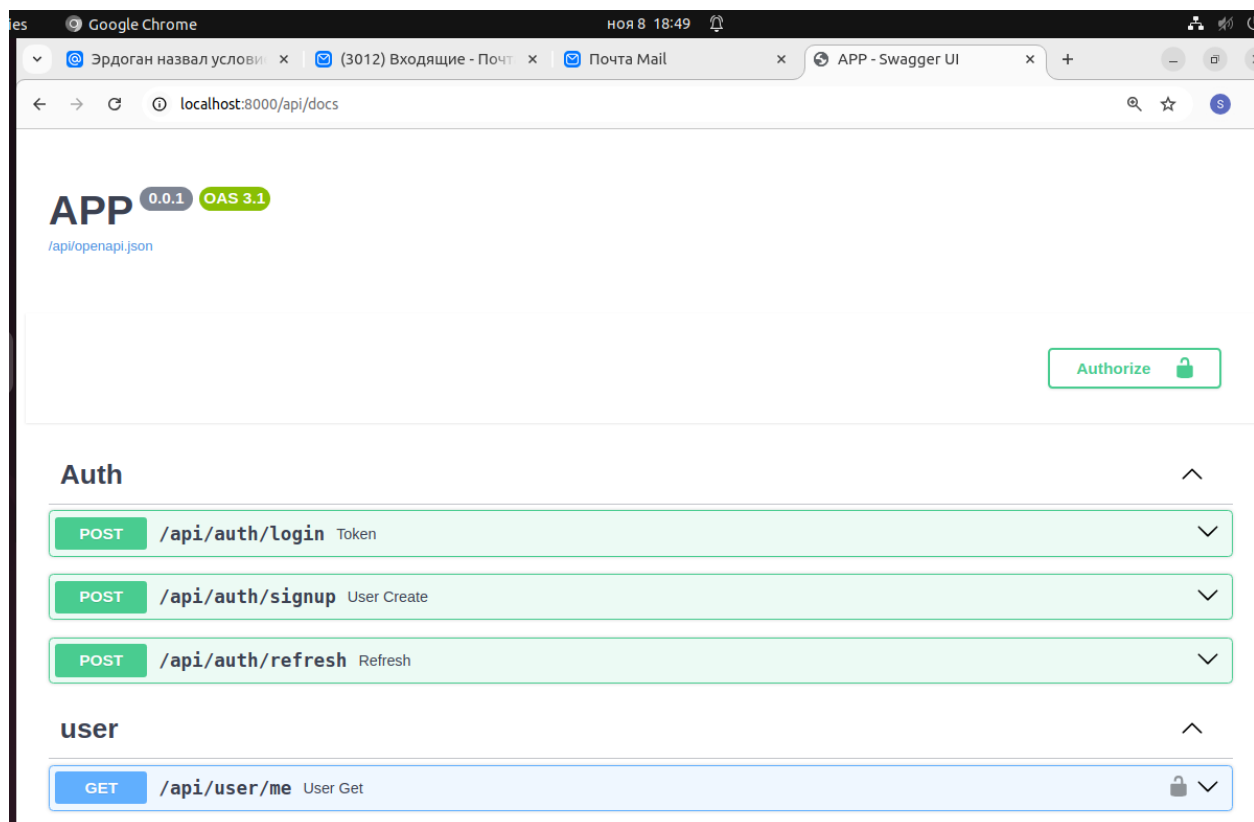


Рис.4 Проверка работы контейнеров в браузере

- Результаты тестирования healthcheck.

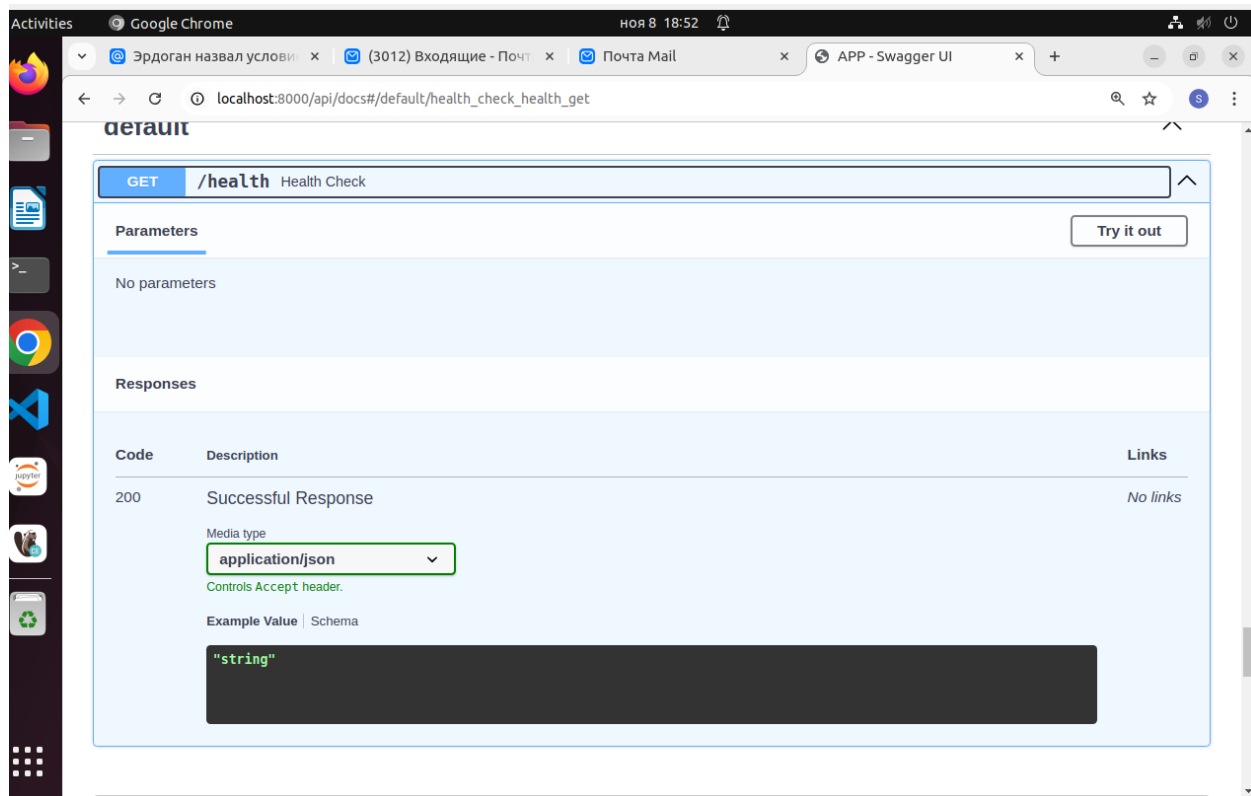


Рис.5 Результаты тестирования healthcheck

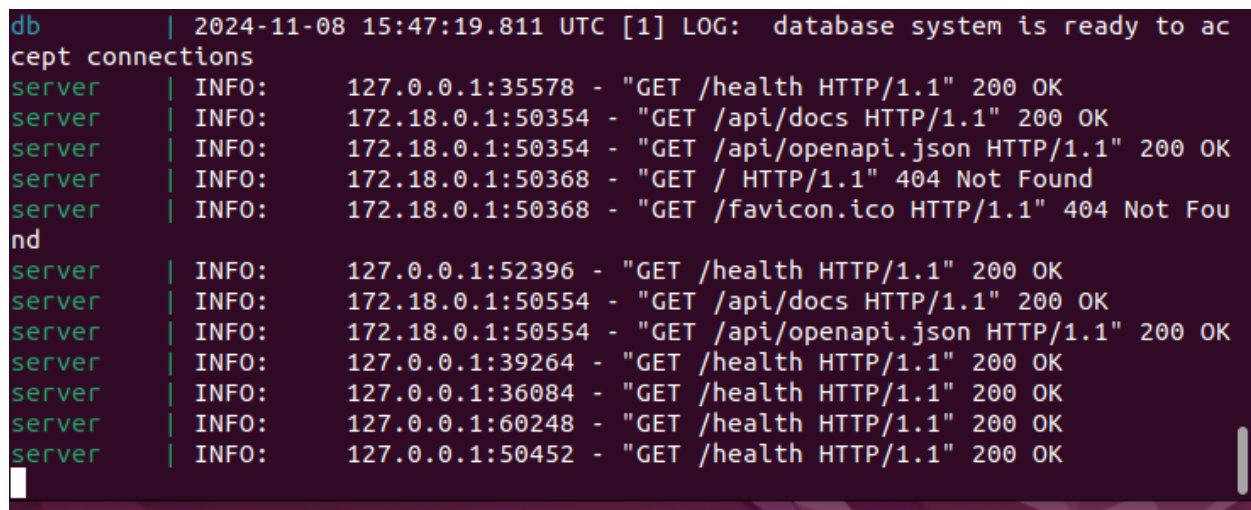


Рис.6

## 6. Ошибки и исправления.

В процессе запуска была ошибка связанная с postgres, не были заданы имя, пароль для postgres и другими параметрами. Чтобы решить проблему в `.env` и `.yml` были добавлены данные строки

`.yml` в контейнер сервера

- `SERVER_PORT=${SERVER_PORT}`
- `CLIENT_PORT=${CLIENT_PORT}` Эти переменные задают порты для сервера и клиента



.env

POSTGRES\_PASSWORD=password  
Пароль для входа в базу данных PostgreSQL.

ACCESS\_TOKEN\_EXPIRE\_MINUTES=15  
Время действия токена доступа

REFRESH\_TOKEN\_EXPIRE\_DAYS=7  
Время действия токена обновления

SECRET\_KEY=your\_secret\_key  
Ключ для шифрования данных и токенов в приложении.

POSTGRES\_USER=postgres  
Имя пользователя для подключения к базе данных PostgreSQL

POSTGRES\_DB=your\_database  
Название базы данных

POSTGRES\_HOST=db  
имя хоста для подключения к базе данных

POSTGRES\_PORT=5432  
Порт для подключения к базе данных PostgreSQL

SERVER\_PORT=8000  
Порт, на котором запускается сервер приложения

CLIENT\_PORT=3000  
Порт, на котором работает клиентское приложени

## **Заключение**

В ходе выполнения практической работы, был создан docker-compose.yml файл с тремя сервисами, настроенны в соответствии с заданными требованиями. Что позволило понять, как работает Docker Compose для координации нескольких контейнеров. Также были встречены ошибки, и приведено их решение