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

Мареев Георгий Александрович БД-241м

Практическая работа 3-2. Docker Compose

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

Руководитель дисциплины:
Босенко Т.М., доцент департамента
информатики, управления и технологий,
кандидат технических наук

Москва
2024

Введение

Цель: освоение основ работы с Docker Compose, создание многоконтейнерного приложения. В ходе работы мы будем использовать Docker Compose для развертывания приложения, состоящего из нескольких сервисов, включая веб-сервер, базу данных и кэш.

Задачи

1. Изучить структуру файла docker-compose.yml и его ключевые директивы.
2. Создать файл docker-compose.yml для простого приложения, включающего (минимум 1 init + 2 app сервиса)
3. Произвести запуск

Основная часть

[Ссылка на видео]

1. Первичный запуск, проверка состояния

При первой попытке создания docker compose была обнаружена следующая проблема:

```
george@george-VirtualBox:~/Documents/devops/DevOps-Technologies/Lesson 3 Containerizing Applications/lab3_2$ docker-compose up
Failed to load /home/george/Documents/devops/DevOps-Technologies/Lesson 3 Containerizing Applications/lab3_2/.env: open /home/george/Documents/devops/DevOps-Technologies/Lesson 3 C
ontainerizing Applications/lab3_2/.env: no such file or directory
```

Рисунок 1Отсутствие. env файла

2. Создание .env файла, с данными для подключения в бд:

```
# Data Base
POSTGRES_USER=postgres
POSTGRES_PASSWORD=password
POSTGRES_DB=your_database
POSTGRES_PORT=5432
POSTGRES_HOST=db
```

3. Создание docker compose

```
server  |     from server.src.core.settings import settings
server  |   File "/app/server/src/core/settings.py", line 54, in <module>
server  |     settings = get_settings()
server  |   File "/app/server/src/core/settings.py", line 51, in get_settings
server  |     return Settings()
server  |   File "/usr/local/lib/python3.10/site-packages/pydantic_settings/main.py", line 71, in __init__
server  |     super().__init__(
server  |   File "/usr/local/lib/python3.10/site-packages/pydantic/main.py", line 212, in __init__
server  |     validated_self = self.__pydantic_validator__.validate_python(data, self_instance=self)
server  | pydantic_core._pydantic_core.ValidationError: 10 validation errors for Settings
server  | access_token_expire_minutes
server  |   Field required [type=missing, input_value={'environment': 'production'}, input_type=dict]
server  |     For further information visit https://errors.pydantic.dev/2.9/v/missing
server  | refresh_token_expire_days
server  |   Field required [type=missing, input_value={'environment': 'production'}, input_type=dict]
server  |     For further information visit https://errors.pydantic.dev/2.9/v/missing
server  | secret_key
server  |   Field required [type=missing, input_value={'environment': 'production'}, input_type=dict]
server  |     For further information visit https://errors.pydantic.dev/2.9/v/missing
server  | postgres_user
server  |   Field required [type=missing, input_value={'environment': 'production'}, input_type=dict]
server  |     For further information visit https://errors.pydantic.dev/2.9/v/missing
server  | postgres_password
server  |   Field required [type=missing, input_value={'environment': 'production'}, input_type=dict]
server  |     For further information visit https://errors.pydantic.dev/2.9/v/missing
server  | postgres_db
server  |   Field required [type=missing, input_value={'environment': 'production'}, input_type=dict]
server  |     For further information visit https://errors.pydantic.dev/2.9/v/missing
server  | postgres_host
server  |   Field required [type=missing, input_value={'environment': 'production'}, input_type=dict]
server  |     For further information visit https://errors.pydantic.dev/2.9/v/missing
server  | postgres_port
server  |   Field required [type=missing, input_value={'environment': 'production'}, input_type=dict]
server  |     For further information visit https://errors.pydantic.dev/2.9/v/missing
server  | server_port
server  |   Field required [type=missing, input_value={'environment': 'production'}, input_type=dict]
server  |     For further information visit https://errors.pydantic.dev/2.9/v/missing
server  | client_port
server  |   Field required [type=missing, input_value={'environment': 'production'}, input_type=dict]
server  |     For further information visit https://errors.pydantic.dev/2.9/v/missing
db exited with code 0
```

Рисунок 2 Ошибки в server

```
init    |   Field required [type=missing, input_value={'environment': 'production'}, input_type=dict]
init    |     For further information visit https://errors.pydantic.dev/2.9/v/missing
init    | postgres_port
init    |   Field required [type=missing, input_value={'environment': 'production'}, input_type=dict]
init    |     For further information visit https://errors.pydantic.dev/2.9/v/missing
init    | server_port
init    |   Field required [type=missing, input_value={'environment': 'production'}, input_type=dict]
init    |     For further information visit https://errors.pydantic.dev/2.9/v/missing
init    | client_port
init    |   Field required [type=missing, input_value={'environment': 'production'}, input_type=dict]
init    |     For further information visit https://errors.pydantic.dev/2.9/v/missing
db exited with code 0
```

Рисунок 3Ошибки в init

4. В .env файл были добавлены следующие строки:

```
#Added access token для настройки механизма аутентификации и базопасности
ACCESS_TOKEN_EXPIRE_MINUTES=30
REFRESH_TOKEN_EXPIRE_DAYS=7
SECRET_KEY=secret_key

#Server Port для указания порта, на котором серверный код будет слушать входящие запросы
SERVER_PORT=8000

#Client Port , на котором будет запущено клиенского приложение
CLIENT_PORT=3000

# Настройка окружения , в каком режиме будет запущено приложение, будем считать что продакшн
ENVIRONMENT=production
```

5. Изменения в docker-compose:

   Контейнер server:

   environment:

   добавлены следующие переменные окружения:

```
- SERVER_PORT=${SERVER_PORT}
- CLIENT_PORT=${CLIENT_PORT}
```

   Контейнер db:

   environment:

   добавлены следующие переменные окружения:

```
POSTGRES_USER: ${POSTGRES_USER}
POSTGRES_PASSWORD: ${POSTGRES_PASSWORD}
POSTGRES_DB: ${POSTGRES_DB}
```

6. Создание docker compose и проверка работоспособности проложения:

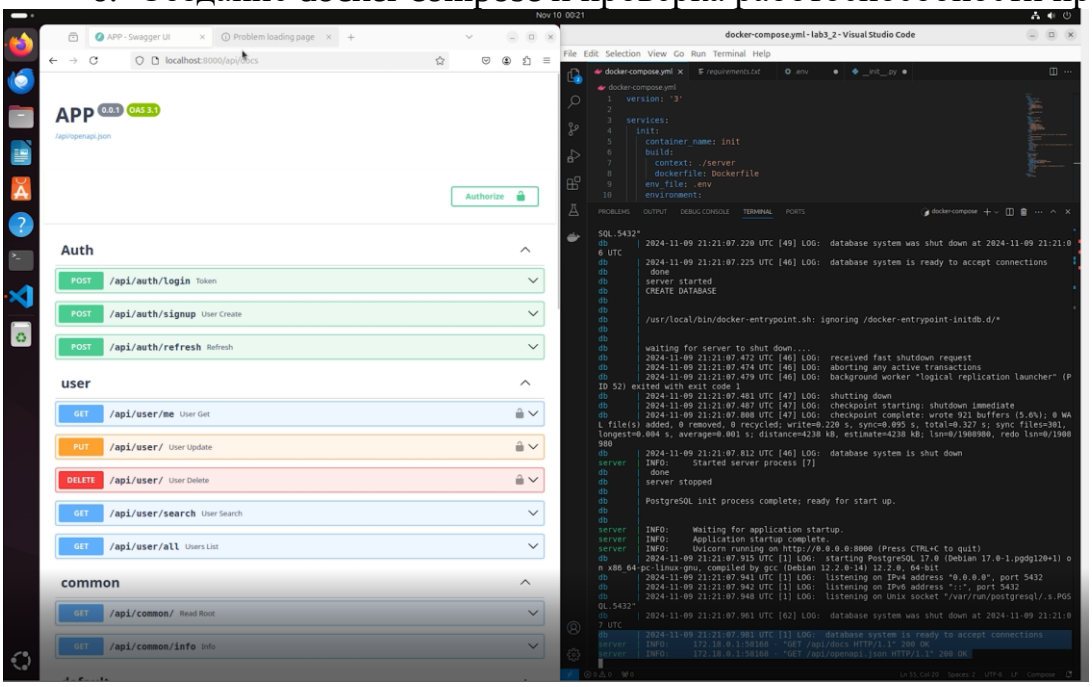

Рисунок 4 Проверка

Заключение

В результате выполнения данной работы, были получены практические и теоретические навыки по работе с Docker, Dockerfile, Docker Compose, а также произведен запуск многоконтейнерного приложения