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

Кузьмина Дарья Юрьевна 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,
доктор экономических наук

Москва
2024

Содержание

Введение

Цель

Цель: рассмотреть пример и выполнить запуск, а также проверку хорошего dockerfile.

Задачи

1. Рассмотреть пример хорошего Dockerfile в каталоге g_method
2. Выполнить сборку Docker-образа
3. Выполнить проверку

Основная часть

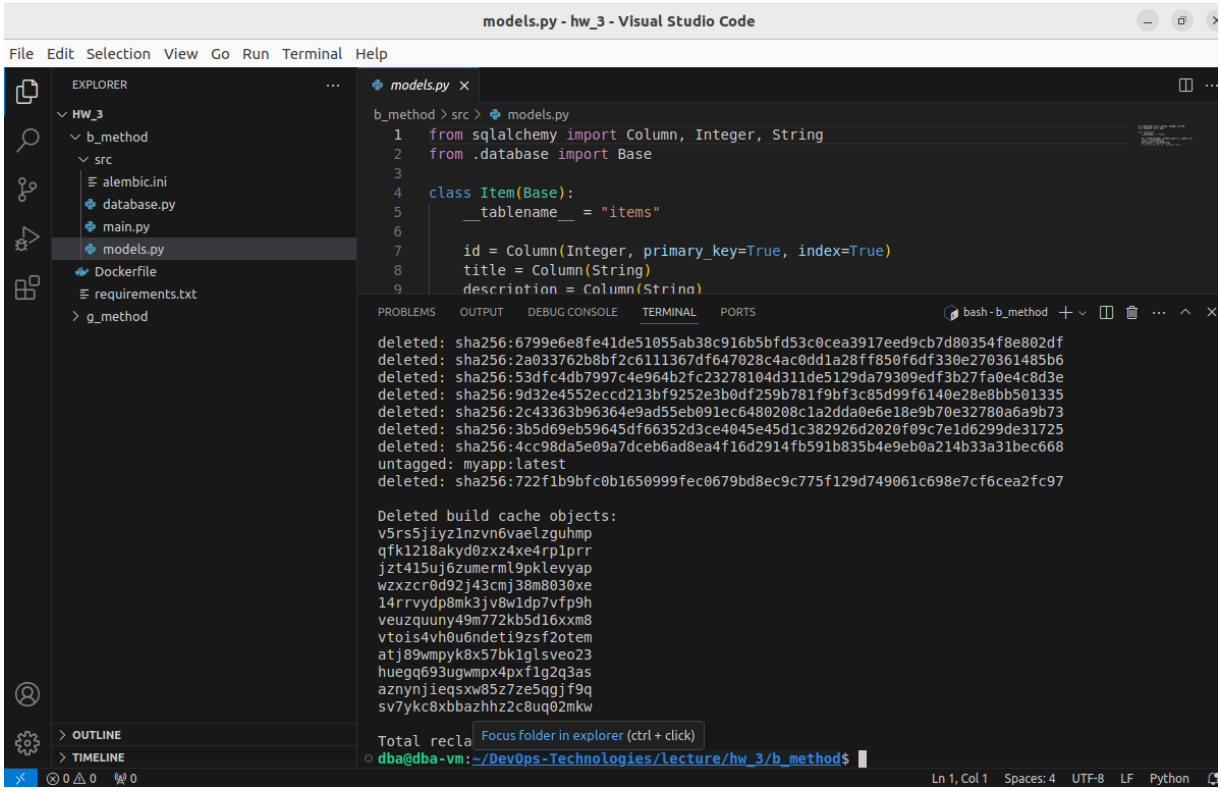

Рисунок 1 «Выполним очистку предыдущей итерации Docker'а»

```
dba@dba-vm:~/DevOps-Technologies/lecture/hw_3/b_method$ cd ..
dba@dba-vm:~/DevOps-Technologies/lecture/hw_3$ cd g_method
dba@dba-vm:~/DevOps-Technologies/lecture/hw_3/g_method$ tree
.
├── docker-compose.yml
├── Dockerfile
├── requirements.txt
└── src
    ├── config.py
    ├── database.py
    ├── __init__.py
    ├── main.py
    ├── models.py
    └── schemas.py

2 directories, 9 files
```

Рисунок 2 «Исследуем структуру при помощи команды tree»

```
dba@dba-vm:~/DevOps-Technologies/lecture/hw_3/g_method$ sudo docker compose up -d
[sudo] password for dba:
[+] Running 2/2
 ✔ Container g_method-db-1   Running                                                0.0s
 ✔ Container g_method-web-1  Started                                                1.5s
dba@dba-vm:~/DevOps-Technologies/lecture/hw_3/g_method$ curl http://localhost:8000
{"message":"Welcome to FastAPI Project"}dba@dba-vm:~/DevOps-Technologies/lecture/hw_3/g_method$ cu
curl http://localhost:8000/docs

    <!DOCTYPE html>
    <html>
    <head>
    <link type="text/css" rel="stylesheet" href="https://cdn.jsdelivr.net/npm/swagger-ui-dist@3/sw
agger-ui.css">
    <link rel="shortcut icon" href="https://fastapi.tiangolo.com/img/favicon.png">
```

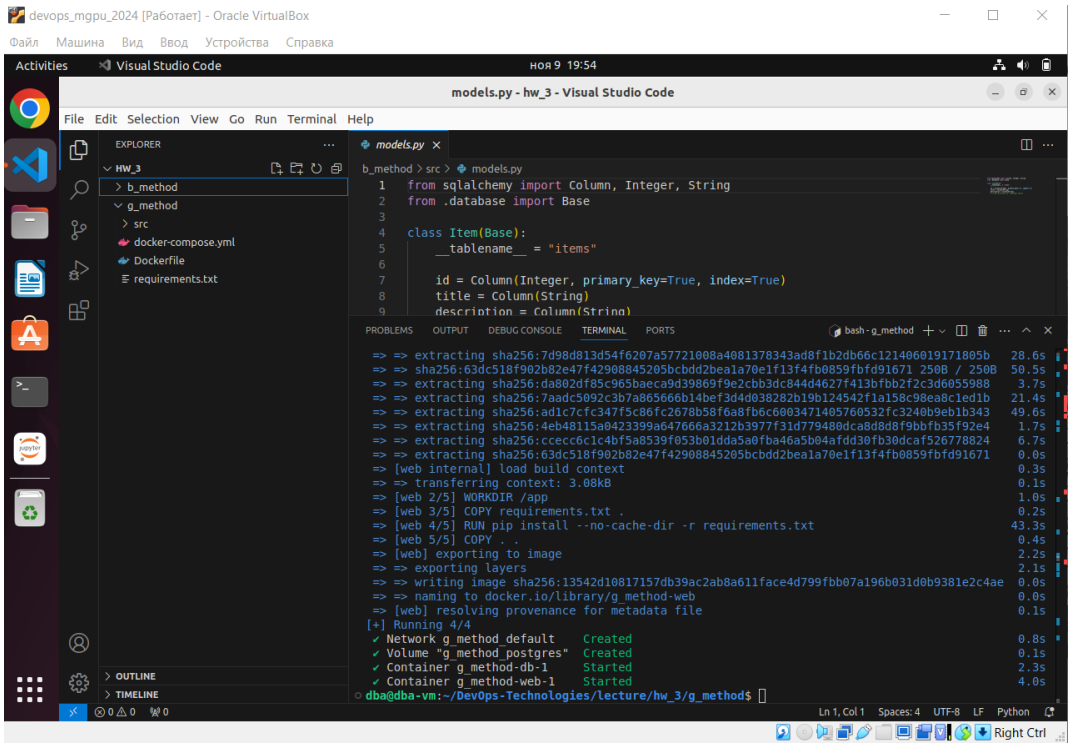

Рисунок 3 «Построим приложение при помощи compose»

```
dba@dba-vm:~/DevOps-Technologies/lecture/hw_3/b_method$ sudo docker run -d --name myapp --link myd
b:db -p 8000:8000 myapp
eb4d18417d37294251d5c8d8592d6040b306ab4eea67e97c18461b828409e74f
dba@dba-vm:~/DevOps-Technologies/lecture/hw_3/b_method$
```

Рисунок 4 «Запускаем сервер и проверим подключение»

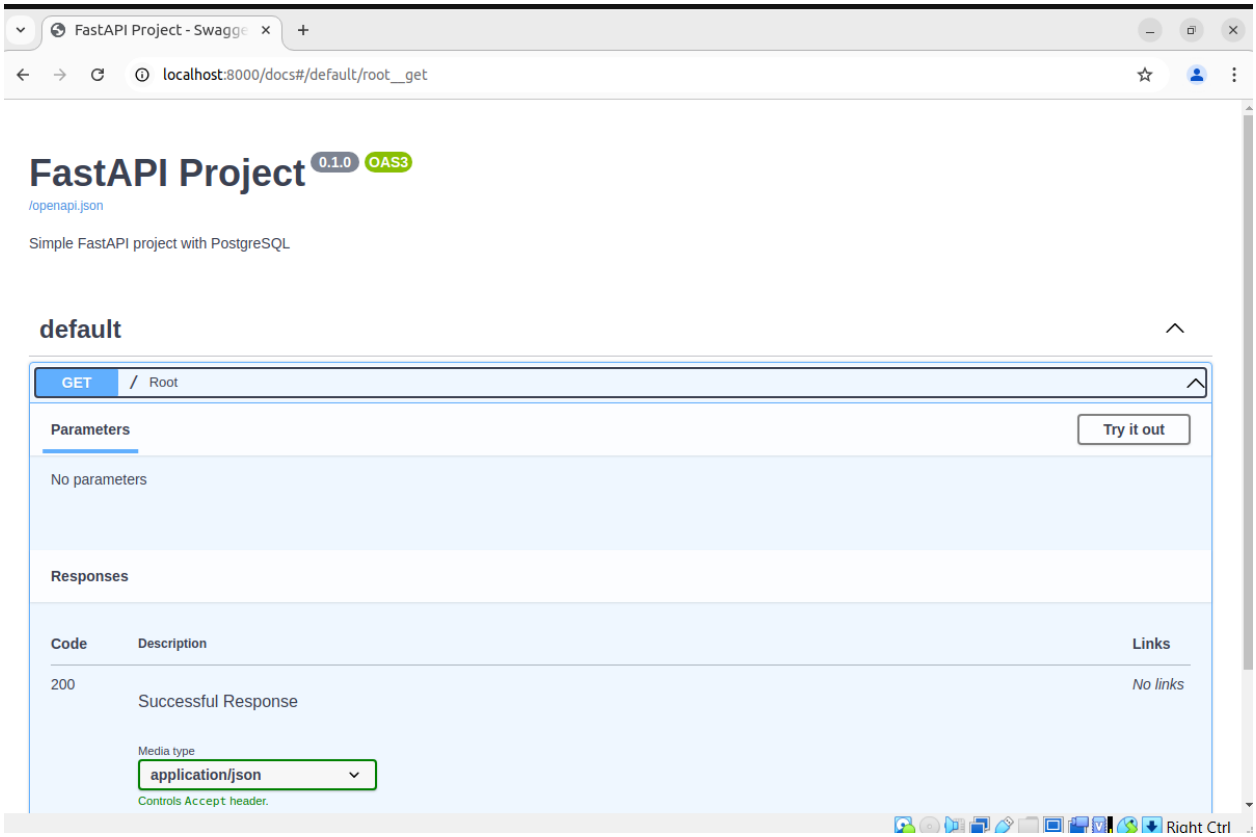

Рисунок 5 «Проверим доступность сервера»

Заключение

В ходе выполнения практической работы были изучены основные концепции и команды Docker, а также принципы создания и использования Dockerfile для автоматизации развертывания приложений в контейнерах. В процессе работы были достигнуты поставленные цели и выполнены все задачи, поставленные перед студентом.

Создание двух Dockerfile — «хорошего» и «плохого» — позволило продемонстрировать различия в подходах к разработке контейнеризированных приложений. «Хороший» Dockerfile характеризовался оптимизированными слоями, минимальным размером

образа и соблюдением лучших практик безопасности, тогда как «плохой» Dockerfile содержал избыточные команды, неэффективное использование слоев и потенциальные уязвимости.

Анализ созданных файлов показал, как неправильная организация Dockerfile может привести к увеличению времени сборки, росту размера образа и снижению безопасности приложения. Это подчеркнуло важность следования рекомендациям по написанию Dockerfile для обеспечения эффективного и безопасного развертывания контейнеров.

Практическое выполнение заданий по запуску контейнеров, настройке сетевых подключений и интеграции с базой данных MySQL позволило приобрести ценные навыки работы с Docker Compose и управления контейнеризированными сервисами. В процессе выполнения работы возникли некоторые трудности, связанные с конфигурацией сети и взаимодействием между контейнерами, однако они были успешно преодолены путем тщательного анализа ошибок и применения соответствующих решений.

В результате выполнения лабораторной работы были получены практические навыки, которые являются основополагающими для дальнейшей работы с контейнеризацией и оркестрацией приложений. Понимание принципов работы Docker и умение создавать оптимизированные Dockerfile способствует повышению эффективности разработки и развертывания программных решений.

В будущем можно расширить полученные знания, изучив более продвинутые инструменты оркестрации контейнеров, такие как Kubernetes, а также углубиться в темы безопасности контейнеров и автоматизации CI/CD процессов с использованием Docker.

Таким образом, выполнение данной лабораторной работы способствовало углублению понимания технологий контейнеризации и подготовки к их применению в реальных проектах.