|  |
| --- |
| МИНОБРНАУКИ РОССИИ |
| Федеральное государственное бюджетное образовательное учреждение высшего образования **«МИРЭА − Российский технологический университет»**  **РТУ МИРЭА** |

**Институт информационных технологий (ИИТ)**

**Кафедра практической и прикладной информатики (ППИ)**

**ОТЧЕТ ПО ПРАКТИЧЕСКОЙ РАБОТЕ**

по дисциплине «Технологии виртуализации клиент-серверных приложений»

# Практическое занятие № 3

|  |  |  |  |
| --- | --- | --- | --- |
| Студент группы ИНБО-01-17 | *ИКБО-20-19, Московка А.А.* | (подпись) | |
| Старший преподаватель | *Волков М.Ю.* | (подпись) | |
| Отчет представлен | «\_\_\_» октябрь 2022 г. | |  | |

Москва 2022 г

Оглавление

[**Практическое занятие № 3 1**](#_Toc115942041)

[**Цель работы 3**](#_Toc115942042)

[**Ход работы 4**](#_Toc115942043)

[**Вывод 9**](#_Toc115942044)

[**Ответы на вопросы к практической работе 9**](#_Toc115942045)

[**Список использованной литературы 11**](#_Toc115942046)

# Цель работы

1. Создать один или несколько Dockerfile, в которых каждая из нижеприведенных команд будет использована хотя бы 1 раз:

* FROM;
* RUN;
* LABEL;
* CMD;
* EXPOSE;
* ENV;
* ADD;
* COPY;
* ENTRYPOINT;
* VOLUME;
* USER;
* WORKDIR;
* ONBUILD.

1. Составлен Dockerfile с веб-приложением.

* Параметры веб приложения:
  + Написано с помощью фреймворка Spring Boot;
  + Взаимодействует с СУБД PostgreSQL;
  + Реализованы следующие эндпоинты:
    - добавление элемента,
    - вывод списка всех элементов,
    - получение фото герба РТУ МИРЭА.
* При сборке проекта с помощью консольной утилиты wget скачан файл по адресу https://www.mirea.ru/upload/medialibrary/80f/MIREA\_Gerb\_Colour.png. Файл должен быть доступен по эндпоинту веб-сервиса.
* Порт СУБД должен быть получен из переменной окружения, указанной в Dockerfile.
* Dockerfile должен содержать 2 стадии: сборка и запуск jar файла, стадии должны быть изолированы.
* В LABEL должны быть указаны ФИО и группа студента.
* По завершению запуска сервиса произведен вывод строки с вашим ФИО “Сборка и запуск произведены. Автор: {ФИО студента}” с помощью команды ONBUILD.
* Docker образ загружен в DockerHub.

# Ход работы

Для выполнения первой части практической работы требуется создать Dockerfile, содержащий набор из перечисленных в задании команд. Ничего запускать не требуется, поэтому представим, что у нас уже есть приложение на Python, которое необходимо контейнеризовать.

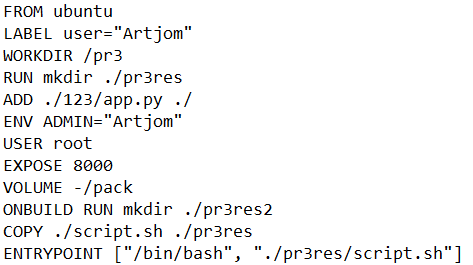


Рис. 1 – Dockerfile для первой части практической работы

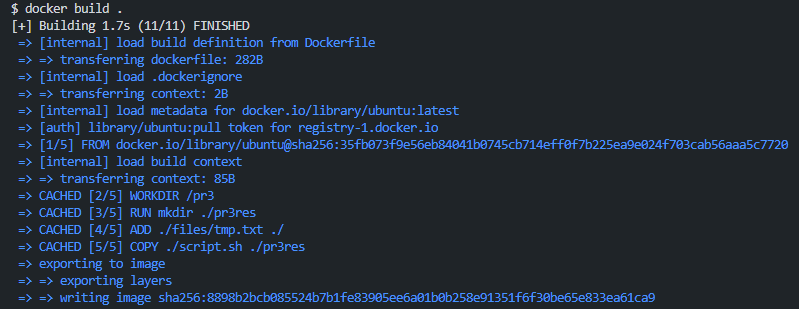


Рис. 2 – Результат выполнения команды docker build

Во второй части задания необходимо создать свой сервис на spring и контейнеризовать его. Ниже приведены dockerfile приложения с раздельными сборкой и запуском jar файла и docker-compose, выполняющий запуск базы данных и приложения.

|  |
| --- |
|  |
| Рис. 2 – Dockerfile для второй части практики |

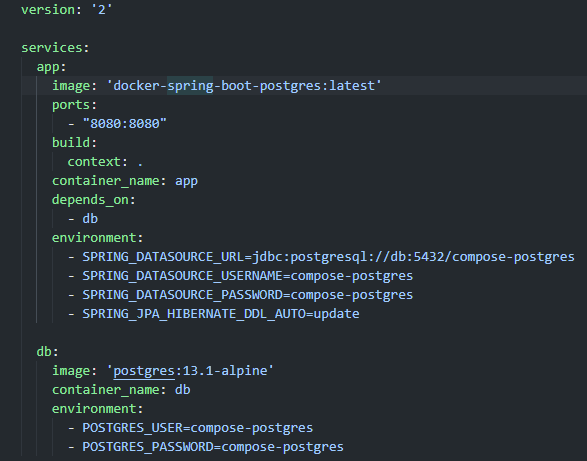


Рис. 3 – Содержание docker-compose.yml

Также необходимо было создать 3 конечные точки. Они представлены на рисунке ниже.

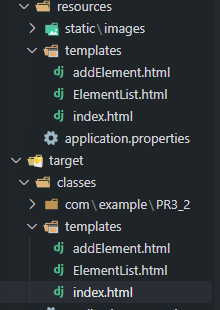


Рис. 4 – Конечные точки проекта

После запуска docker-compose необходимо перейти по адресу localhost:8080, чтобы убедиться в работоспособности проекта

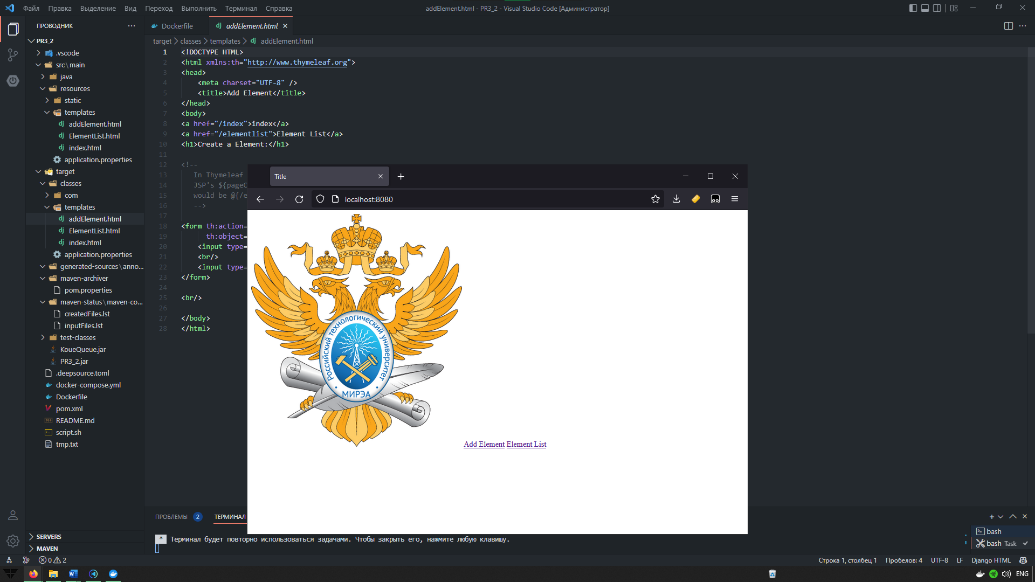


Рис. 5 – Вывод логотипа на стартовой странице проекта

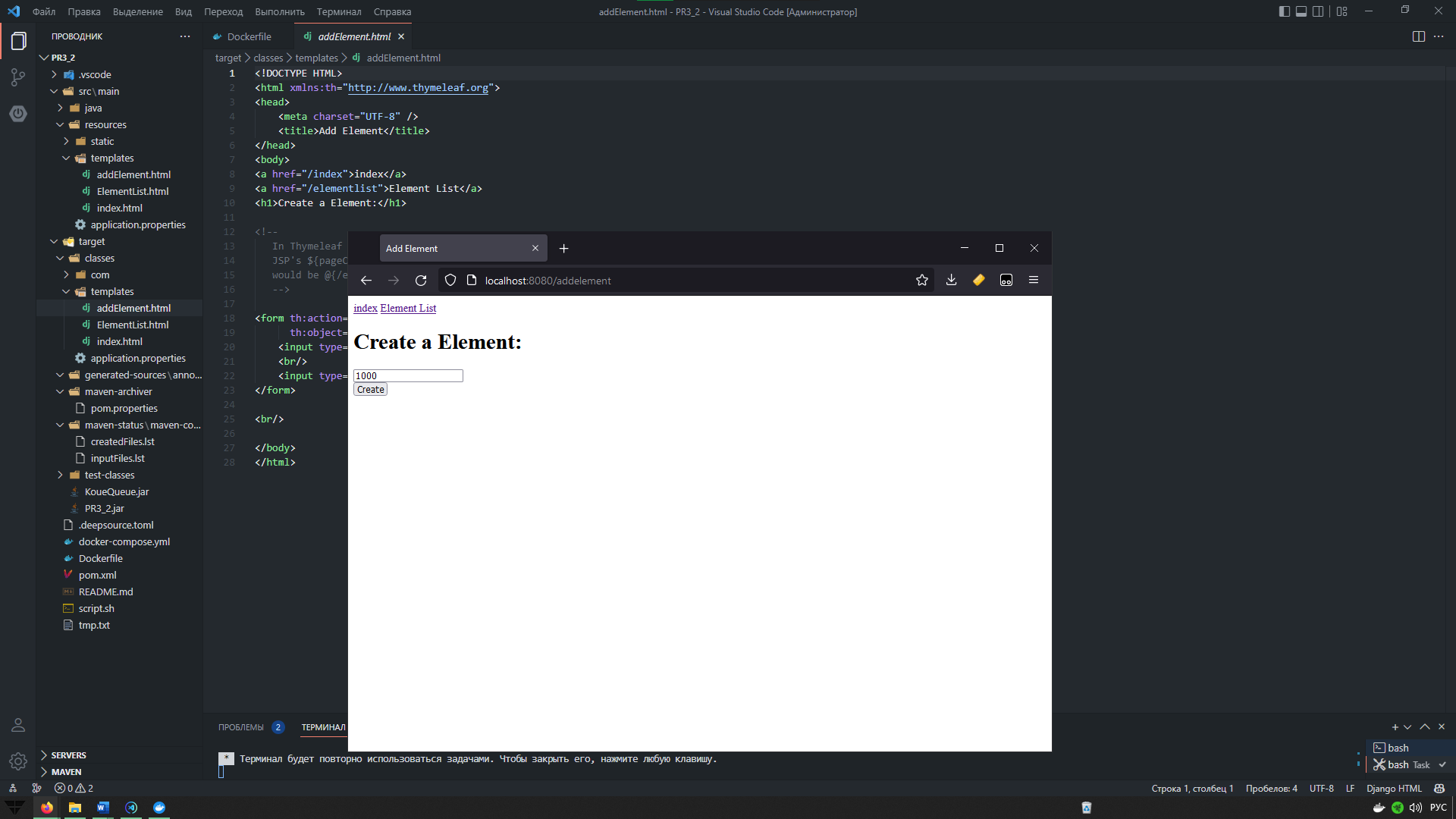


Рис. 6 – Добавление элемента в таблицу

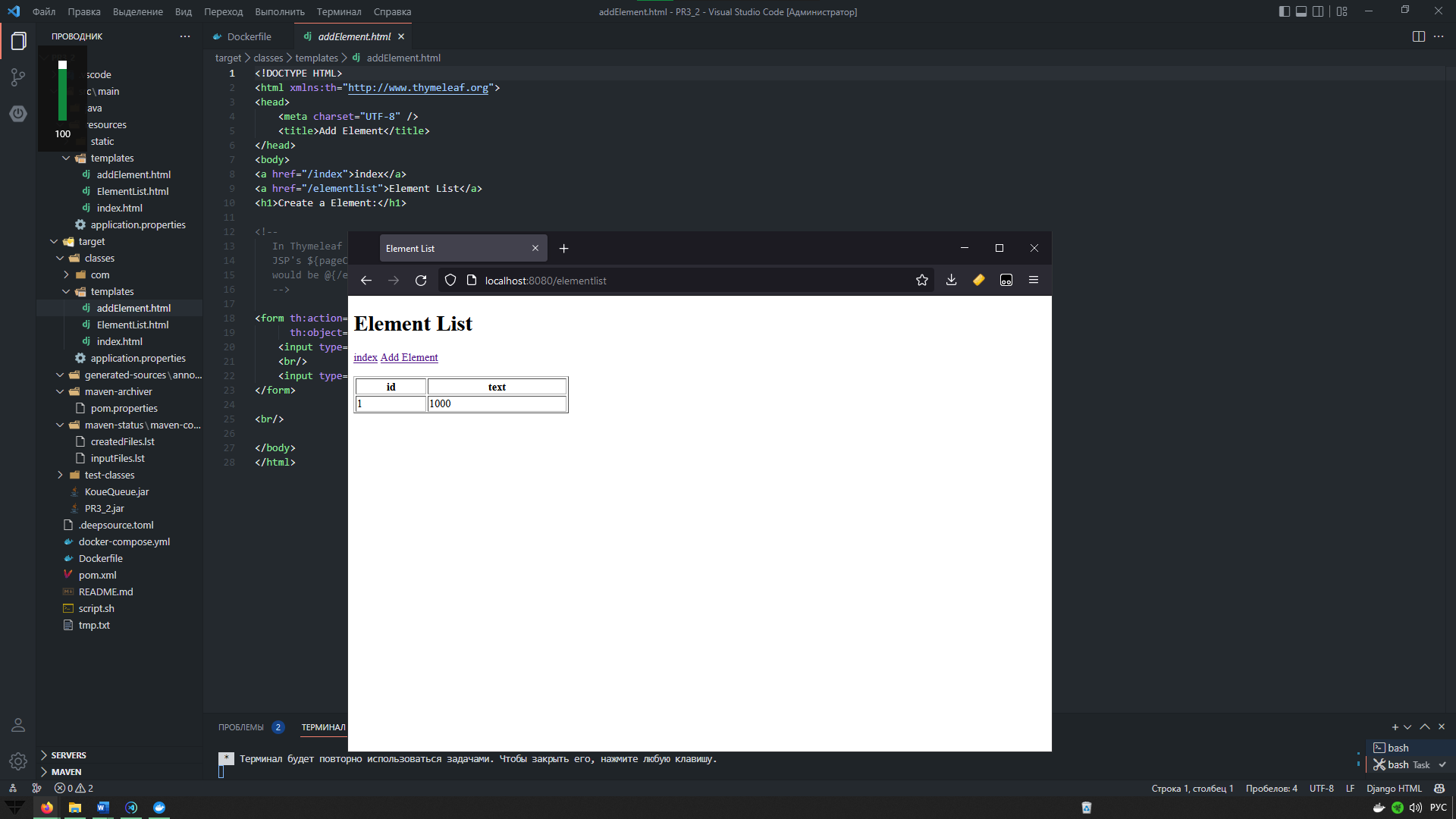


Рис. 7 – Новый элемент в таблице

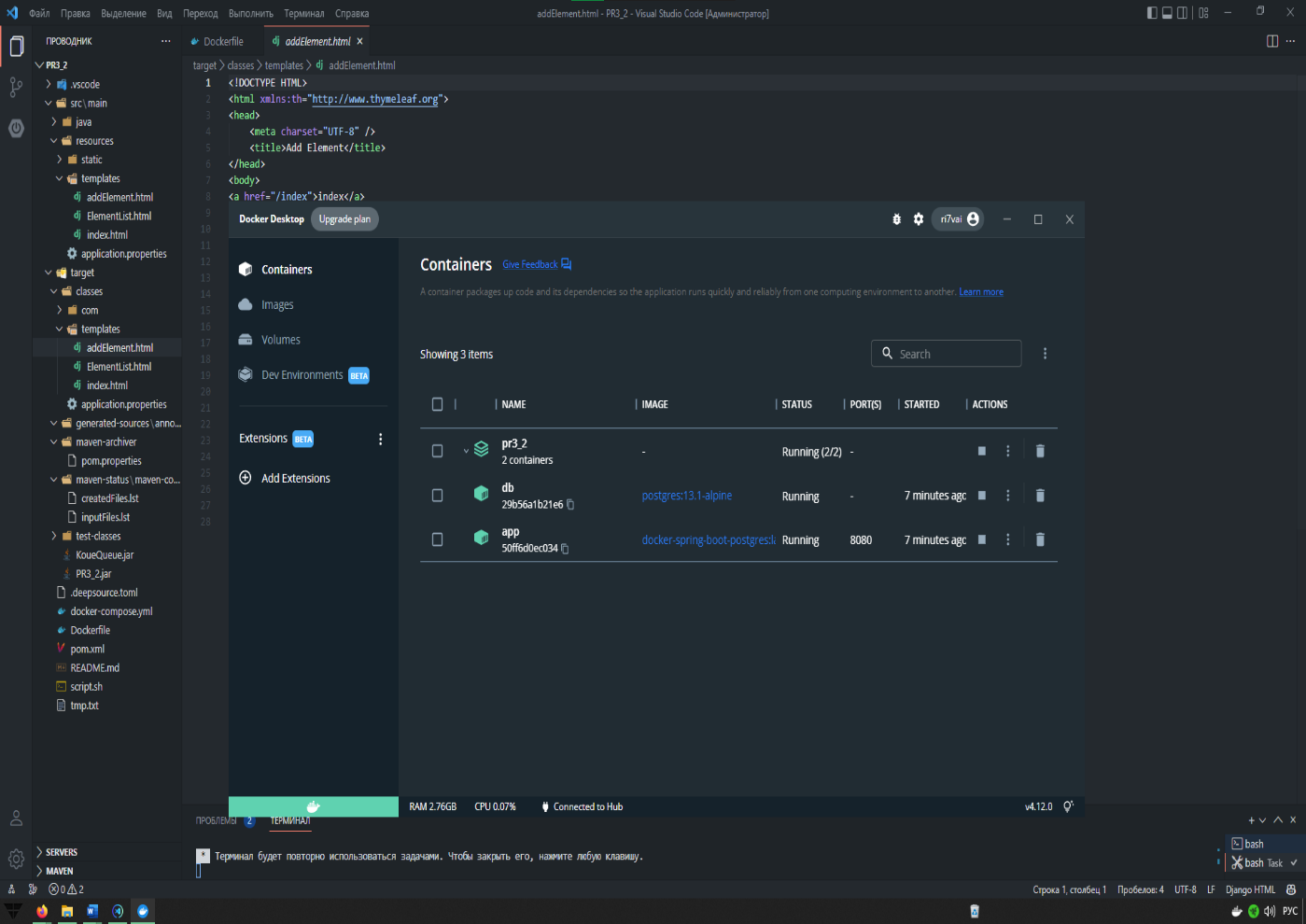


Рис. 8 – Запущенное приложение

Также было необходимо загрузить образ на Docker Hub.

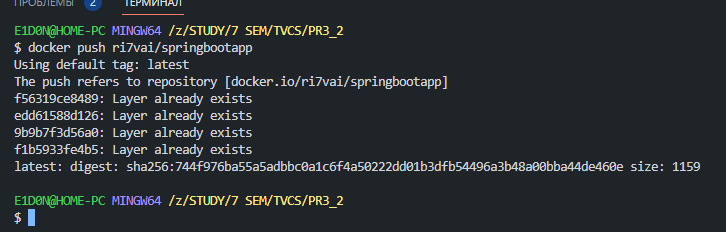


Рис. 9 – Процесс загрузки образа

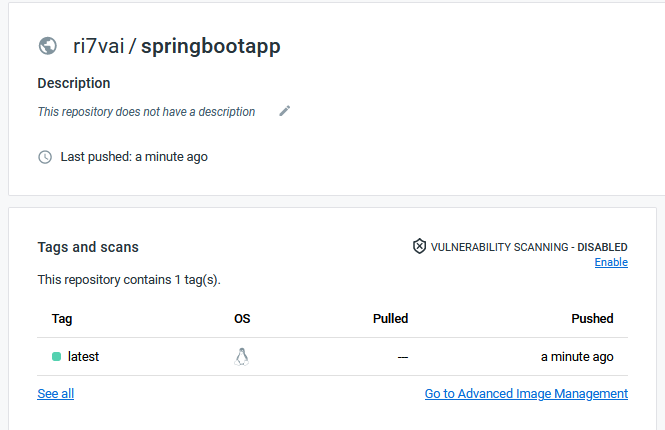


Рис. 10 – Образ на Docker Hub

# Вывод

В результате выполнения третьей практической работы были получены навыки работы с Docker-контейнерами. Также были изучены принципы создания Dockerfile и docker-compose.yml и запуска контейнеров. Произведена работа по контейнеризации приложения spring boot.

# Ответы на вопросы к практической работе

1. Опишите процесс запуска приложения внутри контейнера Linux, используя Docker.

Для запуска контейнера используется команда docker run с именем контейнера. Запуск приложений внутри контейнера возможен через консоль работающего контейнера или с помощью выполнения команды CMD в Dockerfile при сборке контейнера.

2. Что такое образ Docker и для чего он нужен?

Docker — это программная платформа для разработки, доставки и запуска контейнерных приложений. Он позволяет создавать контейнеры, автоматизировать их запуск и развертывание, управляет жизненным циклом. С помощью Docker можно запускать множество контейнеров на одной хост-машине.

3. Как соотносятся между собой файлы Dockerfile и Docker-Compose?

Dockerfile - это обычный текстовый файл, содержащий инструкции по созданию образов Docker. Существует стандарт Dockerfile, которому они следуют, и демон Docker в конечном итоге отвечает за выполнение Dockerfile и генерацию образа.

Docker Compose - это инструмент для определения и запуска многоконтейнерных приложений Docker. Используя файл конфигурации YAML, Docker Compose позволяет нам настраивать несколько контейнеров в одном месте. Затем мы можем запускать и останавливать все эти контейнеры одновременно, используя одну команду.

4. Что такое Dockerfile?

Dockerfile содержит инструкции для сборки образов, которые передаются в Docker. Также его можно описать как текстовый документ, содержащий все возможные команды, с помощью которых пользователь, последовательно их запуская, может собрать образ.

5. Опишите политики перезапуска контейнера.

Существуют следующие политики перезапуска контейнеров Docker:

* no: по умолчанию контейнеры не запускаются автоматически.
* always: всегда перезапускать остановленный контейнер, если контейнер не был остановлен явно
* unless-stopped: перезапустить контейнер, если контейнер не был в остановленном состоянии до остановки демона Docker (объяснено позже).
* on-failure: перезапустите контейнер, если он завершился с ненулевым кодом выхода или если демон докера перезапустился.

6. Назовите все возможные состояния контейнеров.

Контейнер — базовая единица программного обеспечения, покрывающая код и все его зависимости для обеспечения запуска приложения прозрачно, быстро и надежно независимо от окружения. Контейнер Docker может быть создан с использованием образа Docker. Это исполняемый пакет программного обеспечения, содержащий все необходимое для запуска приложения, например, системные программы, библиотеки, код, среды исполнения и настройки.

Все возможные состояния контейнера Docker:

● Created — контейнер создан, но не активен.

● Restarting — контейнер в процессе перезапуска.

● Running — контейнер работает.

● Paused — контейнер приостановлен.

● Exited — контейнер закончил свою работу.

● Dead — контейнер, который сервис попытался остановить, но не смог.

# Список использованной литературы

1. 50 вопросов по Docker, которые задают на собеседованиях, и ответы на них | Хабр. — Текст: электронный [сайт]. — URL: <https://habr.com/ru/company/southbridge/blog/528206/>

2. Docker Documentation | Docker Documentation — Текст: электронный [сайт]. — URL: <https://docs.docker.com/>

3. Что такое режим Docker Swarm и когда его использовать? — CloudSavvy ИТ | Cpab. — Текст: электронный [сайт]. — URL: <https://cpab.ru/chtotakoe-rezhim-docker-swarm-i-kogda-ego-ispo4lzovat-cloudsavvy-it/>

4. Dockerfile reference | Docker Documentation — Текст: электронный [сайт]. — URL: <https://docs.docker.com/engine/reference/builder/>