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

Институт Информационных Технологий

Кафедра корпоративных информационных систем (КИС)

**ОТЧЕТ**

по Лабораторной работе №6

по дисциплине

«Разработка программного обеспечения для корпоративных информационных систем»

Выполнил студент группы ИКБО-08-18 Валяев Д.А.

Принял ассистент Масленников В.В.

Выполнено «\_\_\_» \_\_\_\_\_\_\_\_\_2021 г.

Зачтено «\_\_\_» \_\_\_\_\_\_\_\_\_2021 г.

**Задание**

На основе ЛР №3 построить систему с микросервисной архитектурой. Разделить систему на (минимум) 3 контейнера:

а) База данных (возможно с промежуточным API)

б) Сервис аутентификации/авторизации (через OAuth-токены или аналогичным образом)

в) Сервис самого форума

Предполагается реализация на платформе Docker, однако разрешается использовать любые средства виртуализации/контейнеризации.

Дополнительные задания (для автомата – первое задание обязательно + второе или третье задание на выбор):

1) Реализовать отдельный контейнер с кэш-прокси, маскирующим ресурс под традиционную архитектуру (по коду клиентской части не должно быть понятно, что используются микросервисы)

3) Реализовать дополнительный сервис в описанной инфраструктуре на ваш выбор — файлы, новости, личные сообщения и т.п.

**Содержание работы**

1. Реализация основного функционала
   1. База данных в Docker
   2. Вынос авторизации в отдельный сервис
   3. Запуск форума в Docker
2. Реализация дополнительного функционала
   1. Реализация новостного сервиса
   2. Реализация отдельного контейнера для маскировки под традиционную архитектуру

**Выполнение работы**

1. **Основной функционал**
   1. Запуск базы данных в Docker производится при помощи команды

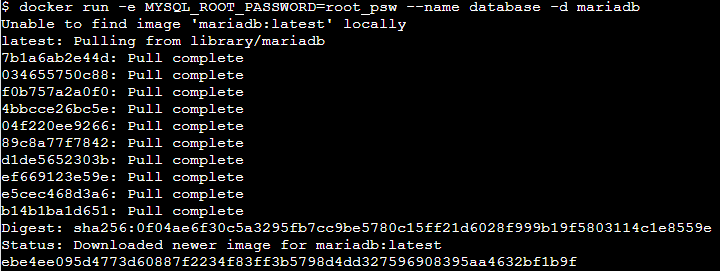


Рис 1.1 – запуск базы данных в docker

Проверим, что контейнер запустился командой «docker ps -a»



Рис 1.2 – проверка списка контейнеров

Для проверки работоспособности базы данных выполним команду «show databases», которая покажет список всех существующим таблиц.

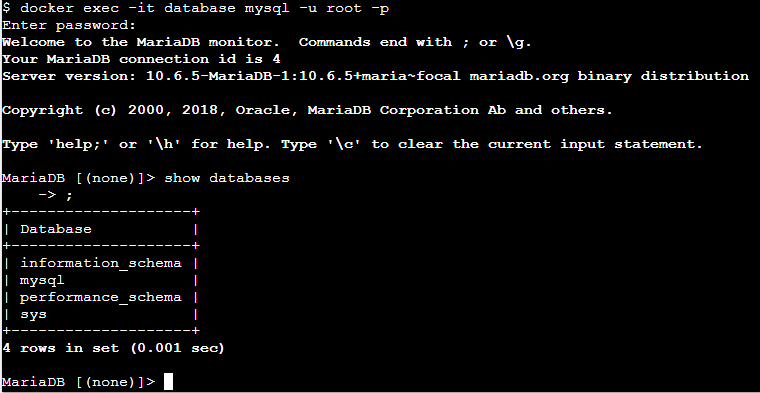


Рис 1.3 – проверка работоспособности базы данных

1.2 Сервис авторизации был вынесен отдельно и практически не изменился. Изменениями стали: добавление авторизации по JWT-токенам; добавление эндпоинта для проверки токена.

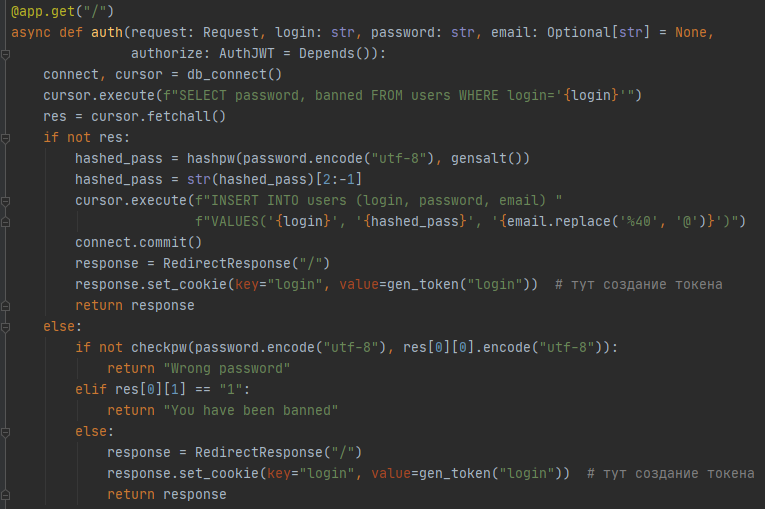


Рис 1.4 – код авторизации

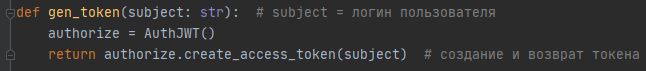


Рис 1.5 – функция генерации токена

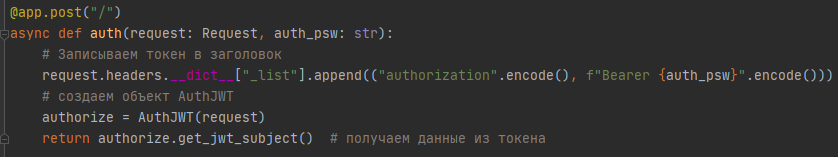


Рис 1.6 – эндпоинт проверки токена

Далее необходимо запустить сервис в докере, для этого создаем докер файл, в котором указываем рабочую директорию (в нашем случае /src) файл с зависимостями (requirements.txt) и команду для запуска сервиса. После чего запускаем сервис с использованием Dockerfile в docker.

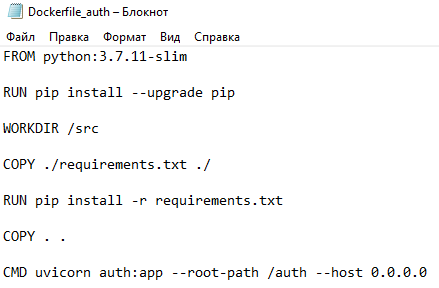


Рис 1.7 – текст Dockerfile\_auth

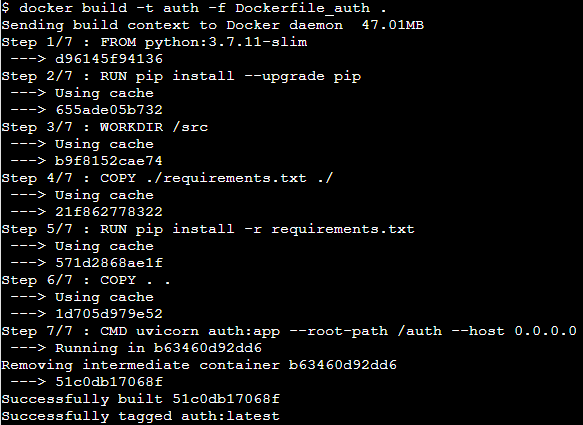


Рис 1.8 – создание контейнера сервиса авторизации



Рис 1.9 – запуск контейнера сервиса авторизации

1.3 Для запуска форумной части создадим dockerfile, схожий с тем, что был создан для сервиса авторизации, только с другой командой запуска.

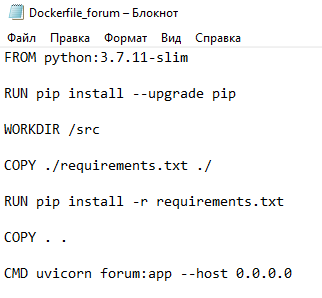


Рис 1.10 – текст Dockerfile\_forum

Код форумной части сервиса остался практически без изменений, был вынесен сервис авторизации и все запросы теперь отправляются на него.

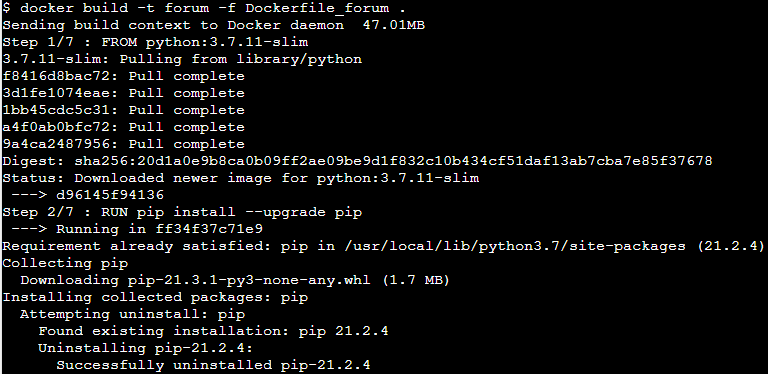


Рис 1.11 – Создание контейнера форума



Рис 1.12 – запуск контейнера форума

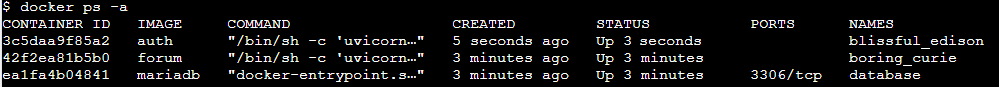


Рис 1.13 – проверка списка контейнеров

1. **Дополнительный функционал**

Кроме возможности авторизации и добавления записи были реализованы дополнительные функции, такие как:

* 1. Реализация новостного сервиса. Сервис имеет два эндпоинта с методами GET-для получения новостей и POST-для добавления новых новостей.

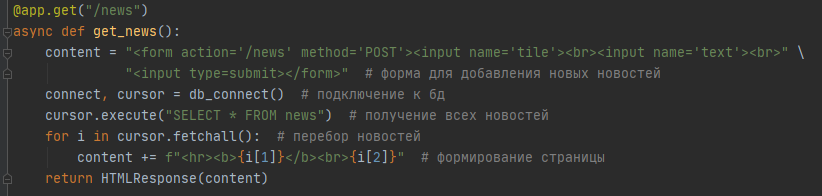


Рис 2.1 – получение новостей

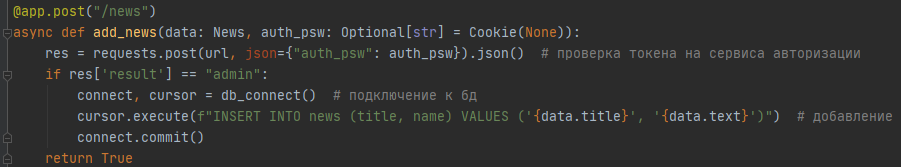


Рис 2.2 – добавление новостей

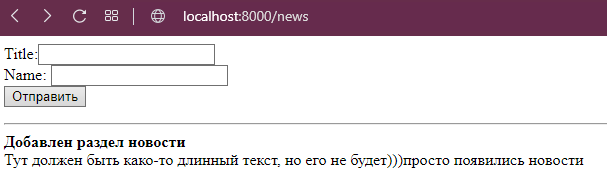


Рис 2.3 – страница новостей

2.2 Создание кэш-прокси для маскировки под традиционную архитектуру. Для начала необходимо создать Dockerfile, который позволит запустить все сервисы в одном контейнере.

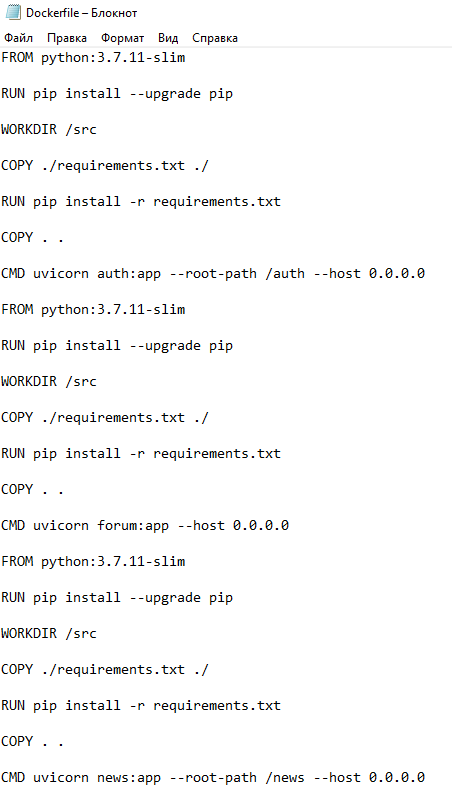


Рис 2.4 – общий Dockerfile

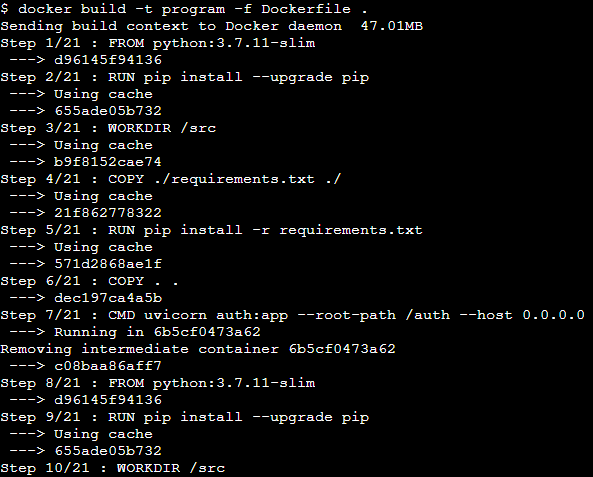


Рис 2.5 – создание контейнера



Рис 2.6 – создание прокси

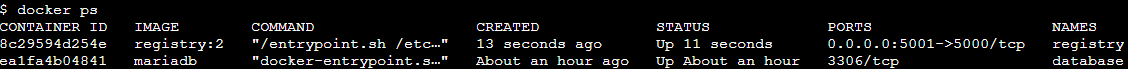


Рис 2.7 – проверка прокси

Далее последовательно выполняем команды «docker tag program localhost:5000/program», «docker push localhost:5000/program», теперь контейнер можно запускать посредством выполнения команды «docker pull localhost:5000/program».

**Вывод**

В данной лабораторной работе были приобретены навыки работы с Docker, создания и запуска контейнеров, а также навыки по организации взаимодействия микросервисов.