Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования «Ульяновский государственный технический университет»

Кафедра «Вычислительная техника»

**Отчет по лабораторным работам №9**

Дисциплина: «Разработка профессиональных приложений»

Docker

Вариант № 12

Выполнил:

студент группы ИВТАСбд-21

Клишин К. С.

Проверил:

преподаватель кафедры

«Вычислительная техника»

Исхаков И.И.

Ульяновск

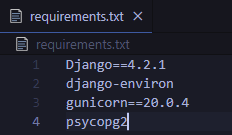
[2023](https://github.com/ghub-ayrtom/DPA-lab5)

[**Постановка задачи**](https://github.com/ghub-ayrtom/DPA-lab5)

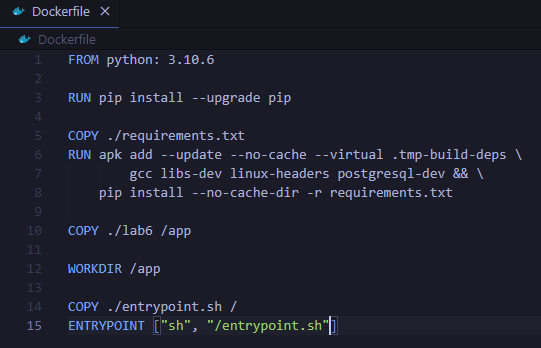
Для лабораторной работы №9 необходимо подготовить Docker Compose файл, позволяющий запустить 2 контейнера: 1. с автоматическим развертыванием всех необходимых библиотек и запуска gunicorn сервера; 2. для СУБД postgresql.

**Описание реализации**

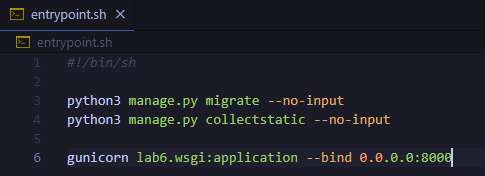
В файле requirements были прописаны все необходимые библиотеки



Далее создаём Dockerfile. Он содержит набор инструкций для записи образа. При выполнении docker run идёт проверка, есть ли образ на диске.



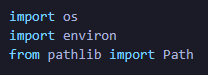
Содержимое скрипта entrypoint:

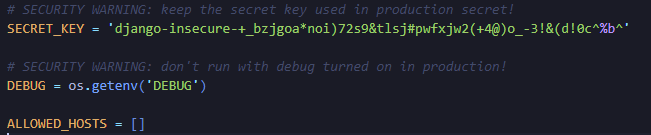


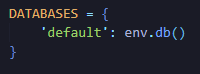
Далее создан конфигурационный файл docker-compose.yml.



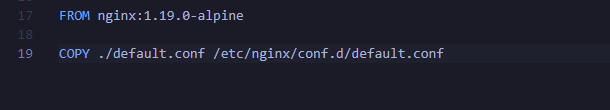
Также был доработан файл settings.py







Для начала, в папке с python проектом DPA-lab6 был создан подкаталог nginx и в нём 2 файла: default.conf и ещё один Dockerfile, который имеет следующую структуру:

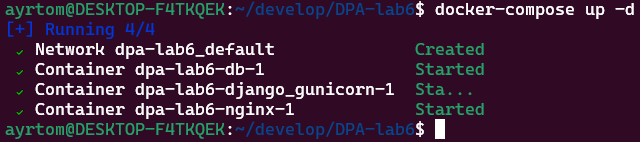


В нём мы командой FROM импортируем nginx версии 1.19.0. и заменяем конфиг nginx по умолчанию тем конфигом, который мы создали сами и который выглядит следующим образом:

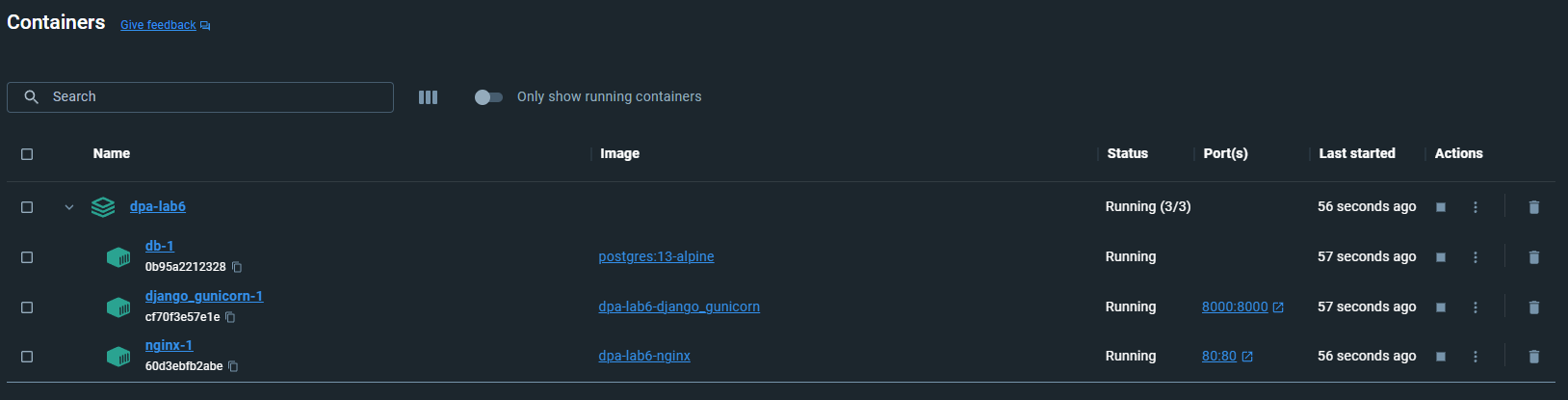


Как видно, веб-сервер nginx занимает 80 порт, в свою очередь WSGI-сервер gunicorn занимает стандартный порт 8000, он расчитан на более серьёзную нагрузку, нежели локальный django-сервер, а также является более безопасным.

После того как все шаги выше были успешно выполнены, можно попробовать запустить docker-compose файл следующей командой в терминале WSL:



Развернулось 3 Docker-контейнера, которые выглядят следующим образом:



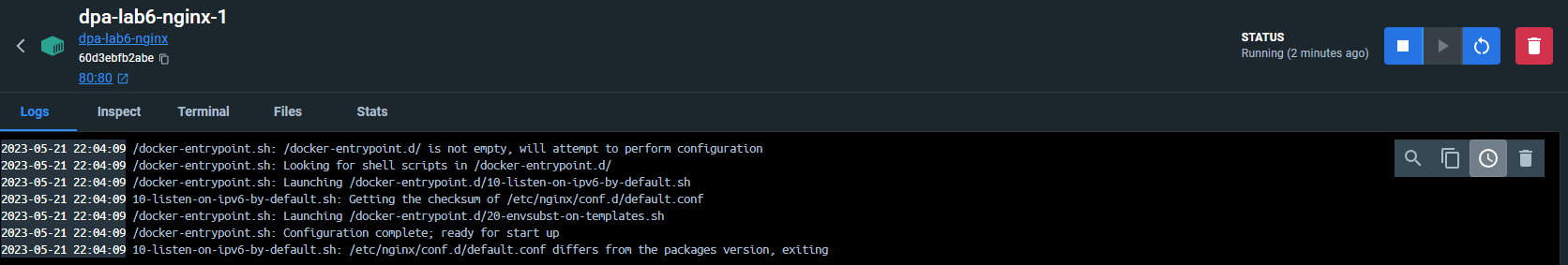
Контейнер dpa-lab6-db-1 отвечает за развёртывание СУБД postgresql:



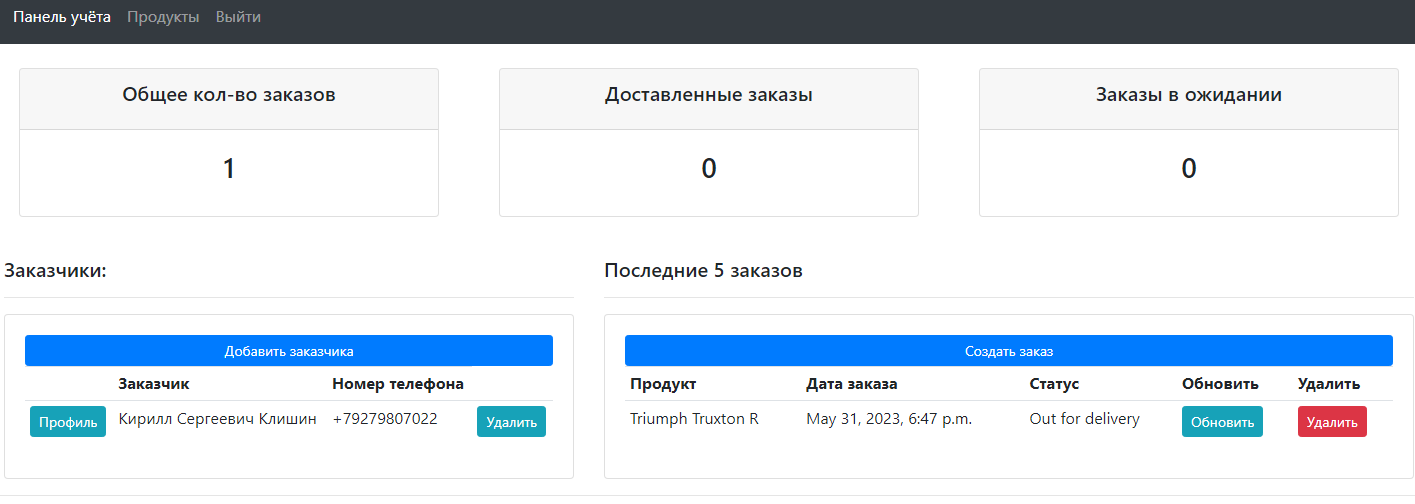
Контейнер dpa-lab6-django-gunicorn-1 отвечает за автоматическое развёртывание всех необходимых библиотек и запуска gunicorn сервера:



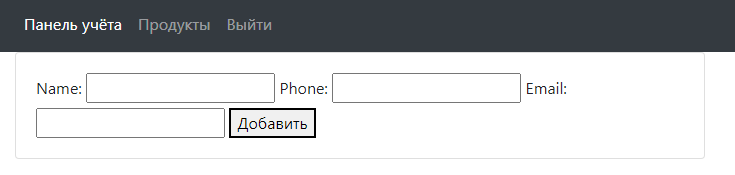
Контейнер dpa-lab6-nginx-1 отвечает за запуск nginx-сервера:

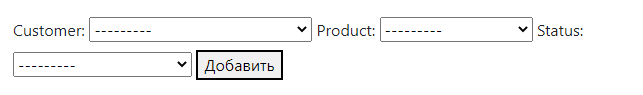


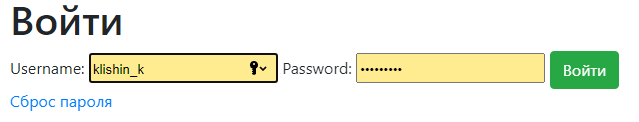
Теперь можно попробовать зайти на сам локальный сервер gunicorn (порт: 8000) и посмотреть, всё ли работает:

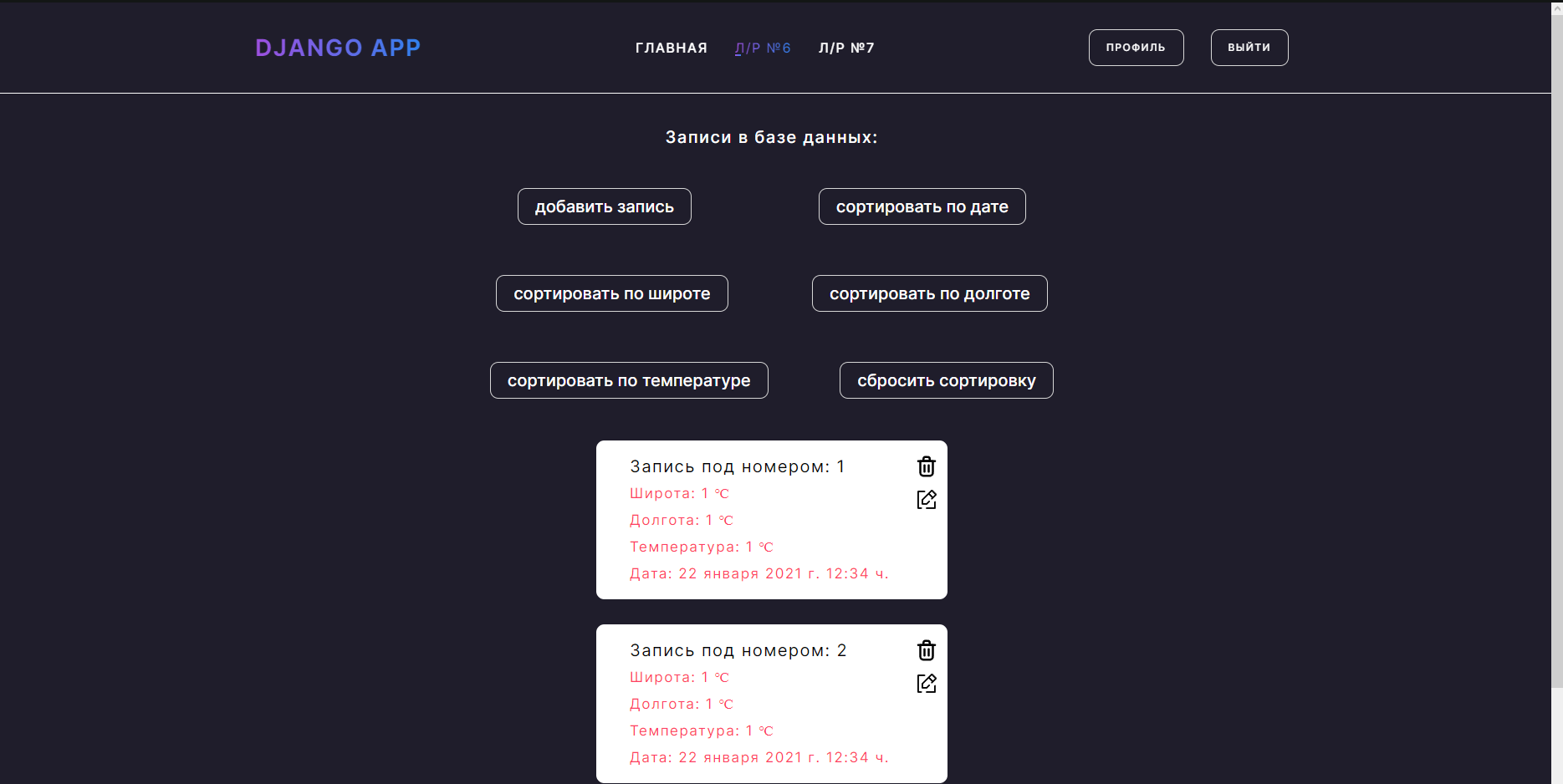


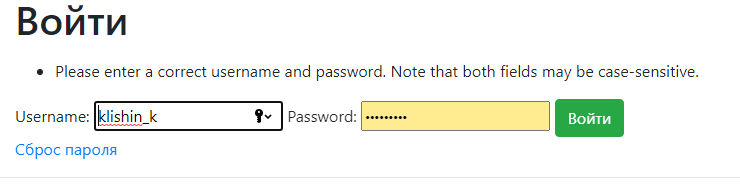
Или же зайти на nginx-сервер (порт: 80) и увидеть всё то же самое, что и на gunicorn-сервере:











**Описание возникших затруднений**

Основные затруднения при выполнении данной лабораторной работой были связаны с первичным знакомством с технологией контейнеризации Docker, так как до этого мне не приходилось с ней работать. Было достаточно непросто понять с чего именно нужно начать, с создания какого файла, а после этого было трудно разобраться как во внутренней структуре самих файлов, так и в том, в каком порядке они должны создаваться и подключаться друг к другу.

Также, немало проблем возникло с подключением и установкой необходимых библиотек и их совместимостью.

**Вывод**

По итогу выполнения данной лабораторной работы был создан Docker Compose файл и множество вспомогательных для него файлов, которые позволяют запустить 3 контейнера: 1. с автоматическим развертыванием всех необходимых библиотек и запуска gunicorn сервера; 2. для СУБД postgresql и 3. с запуском nginx сервера