Итоговое домашнее задание

Дисциплина	Семинар наставника
Тема	Разработка и деплой FastAPI-сервисов с использованием Docker
Форма проверки	**Задание с индивидуальной проверкой преподавателем.** *Совет: выполняйте домашнее задание сразу после изучения темы*
Имя преподавателя	Владислав Шевченко
Время выполнения	6 часов
Цель задания	Закрепить навыки разработки микросервисов на базе FastAPI, работы с Docker и GitHub.
Инструменты для выполнения ДЗ	**1. FastAPI** - для разработки микросервисов. **2. Python** (версия 3.8 или выше) - основной язык программирования. **3. SQLite** - база данных для хранения данных. **4. Docker** - для контейнеризации приложений. **5. Git** - для управления версиями кода. Ссылка на репозиторий на GitHub. Настройки репозитория должы быть public **6. GitHub** - для публикации кода в репозиториях. **7. Postman** или **curl** - для тестирования API. **8. Uvicorn** - ASGI-сервер для запуска FastAPI-приложений.
Правила приёма работы	Прикрепить в ЛМС ссылки на репозиторий GitHub и опубликованные образы Docker Hub. Настройки репозитория должны быть public.
Критерии оценки	10 (отлично, выше ожидаемого): · Студент реализовал дополнительные функции в обоих проектах (например, авторизацию, дополнительные эндпоинты, расширенные тесты или улучшенную структуру проекта). · Оба сервиса демонстрируют стабильную работу без

	ошибок. · README.md оформлен подробно и включает дополнительную информацию (скриншоты, диаграммы, примеры запросов). 9 (отлично): · Студент реализовал дополнительные функции только в одном из проектов. · Все требования задания выполнены полностью. · Код читаемый, структурированный, без серьёзных недочётов. · README.md содержит полные инструкции для запуска и тестирования сервисов. 8 (отлично): · Все требования задания выполнены полностью. · Docker-контейнеры запускаются корректно, данные сохраняются после перезапуска. 7 (хорошо): · Все основные требования выполнены, но есть небольшие недочёты (например, недостаточно подробная документация или неидеальная структура кода). · Сервисы работают, но незначительные ошибки требуют доработки. 6 (хорошо): · Выполнена большая часть задания, но отсутствуют отдельные части (например, один из сервисов или не реализован Docker). · Присутствуют ошибки, мешающие полноценной работе, но они могут быть исправлены с минимальными усилиями. 5-4 (удовлетворительно): · Реализован только один из проектов (либо TODO-сервис, либо сервис сокращения URL). · Код имеет ошибки или недочёты, но основная функциональность работает.

	3 и ниже (неудовлетворительно): · Задание не выполнено или выполнено с существенными ошибками, из-за которых сервисы не работают. · Большая часть требований задания не соблюдена.
Дедлайн	Мягкий дедлайн 24.12.24, крайний дедлайн - 27.12.24

Задание

- Разработать два микросервиса на базе FastAPI:
  - **TODO-сервис**: Реализует CRUD-операции для списка задач с хранением данных в SQLite.
  - **Сервис сокращения URL (Short URL)**: Позволяет создавать короткие ссылки для длинных URL, перенаправлять по короткому идентификатору и предоставлять информацию о ссылке. Также хранение данных в SQLite.
- Упаковать оба сервиса в Docker-контейнеры, используя именованные Docker-тома для сохранения данных.
- После успешного локального запуска — залить исходный код в GitHub (в публичный репозиторий).
- Собрать образы и опубликовать их на Docker Hub.

Требования к сервисам

Общие требования:

- Каждый сервис — отдельное FastAPI-приложение.
- Сервисы должны иметь автоматическую документацию по адресу /docs.
- Данные должны храниться в SQLite. Файл базы данных должен находиться в директории, которая будет подключена как том при запуске контейнера.

TODO-сервис:

- Эндпоинты:
  - POST /items: Создание задачи (title, description?, completed=false).
  - GET /items: Получение списка всех задач.
  - GET /items/{item_id}: Получение задачи по ID.
  - PUT /items/{item_id}: Обновление задачи по ID.
  - DELETE /items/{item_id}: Удаление задачи.
- Все операции должны работать с базой SQLite.

- Перед запуском должен автоматически создается таблица, если она не существует.

Сервис сокращения URL:

1. Эндпоинты:
    a. POST /shorten: Принимает полный URL (JSON: {"url":"..."}) и возвращает короткую ссылку.
    b. GET /{short_id}: Перенаправляет на полный URL, если он существует.
    c. GET /stats/{short_id}: Возвращает JSON с информацией о полном URL.
2. Данные о сокращенных ссылках (short_id -> full_url) хранятся в SQLite.
3. При запуске также автоматически создается таблица.

Требования к Docker

- Для каждого сервиса написать Dockerfile.
- Пример Dockerfile должен:
  - Установить зависимости (pip install -r requirements.txt).
  - Скопировать код сервиса.
  - Объявить VOLUME /app/data для хранения данных.
  - Запускать uvicorn-приложение на порту 80.

Запуск сервисов через команду docker run:

docker run -d -p 8000:80 -v todo_data:/app/data <имя_образа_todo>

docker run -d -p 8001:80 -v shorturl_data:/app/data

<имя_образа_shorturl>

- Где todo_data и shorturl_data — заранее созданные именованные тома.

Проверка работоспособности

1. TODO-сервис:
    a. POST /items для создания задачи.
    b. GET /items для получения списка задач.
    c. Должен сохранять задачи после перезапуска контейнера (благодаря тому, что база в томе).

2. Сокращение ссылок:
    a. POST /shorten для создания короткой ссылки.
    b. GET /{short_id} для редиректа.

c. GET /stats/{short_id} для просмотра данных о ссылке.
d. Данные о ссылках должны сохраняться после перезапуска контейнера.

Загрузка кода на GitHub

- Создайте публичный репозиторий на GitHub, например my-fastapi-project.
- Загрузите туда исходный код обоих сервисов, каждый в своей директории (например, todo_app и shorturl_app).
- Убедитесь, что в репозитории присутствуют Dockerfile, requirements.txt, main.py и другие нужные файлы.
- Добавьте README.md с описанием, как запускать сервисы локально и через Docker.

Публикация образов на Docker Hub

- Создайте учетную запись на Docker Hub, если ее еще нет.

Соберите образы локально:

```
docker build -t <ваш_логин_hub>/todo-service:latest todo_app/

docker build -t <ваш_логин_hub>/shorturl-service:latest shorturl_app/
```

- Войдите в Docker Hub:

```
docker login
```

- Введите свой логин и пароль.

Запушьте образы:

```
docker push <ваш_логин_hub>/todo-service:latest

docker push <ваш_логин_hub>/shorturl-service:latest
```

- После этого образы будут доступны в Docker Hub, и вы сможете запускать их на любом хосте, имеющем доступ к Docker Hub:

```
docker run -d -p 8000:80 -v todo_data:/app/data <ваш_логин_hub>/todo-service:latest

docker run -d -p 8001:80 -v shorturl_data:/app/data <ваш_логин_hub>/shorturl-service:latest
```

Чеклист самопроверки

Критерии выполнения задания	Отметка о выполнении
Реализован TODO-сервис и сервис сокращения ссылок на FastAPI	
Подключён SQLite для хранения данных	
Написан Dockerfile для каждого сервиса	
Использованы тома для сохранения данных	
Проведено тестирование локально с помощью curl или веб версию /docs	
Исходный код загружен на GitHub	
Собран и опубликован Docker-образ в репозитории Docker Hub	
Предоставить ссылки на GitHub-репозиторий и образы на Docker Hub в отчет и прикрепить в ЛМС	