

**САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ
ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО**

Дисциплина: Бекенд разработка

Отчет

Лабораторная работа 5

Выполнил:

Фадеев Д.А.

К3439

**Проверил:
Добряков Д. И.**

Санкт-Петербург

2025 г.

Задача

- Выделить самостоятельные модули в приложении
- Провести разделение в вашем API на микросервисы
- Настроить сетевое взаимодействие между микросервисами

Ход Работы

Было выделено 3 микросервиса: auth-service, content-service и book-service.

Auth-service отвечает за авторизацию пользователей и получение информации по ним. Swagger UI по Auth-service представлен на рисунке 1.

The screenshot shows the Swagger UI interface for the auth-service API. At the top, it displays the service name "auth-service" with version "1.0.0" and "OAS 3.0". Below this, there are sections for "Auth", "Health", and "Users".

- Auth:** Contains two POST operations:
 - `POST /auth/sign-in` (Description: Вход пользователя и получение JWT токена)
 - `POST /auth/sign-up` (Description: Регистрация нового пользователя)
- Health:** Contains one GET operation:
 - `GET /health` (Description: Проверяет состояние сервера)
- Users:** Contains three operations:
 - `GET /users` (Description: Получить список всех пользователей)
 - `GET /users/{username}` (Description: Получить пользователя по имени пользователя)
 - `PUT /users/{username}` (Description: Обновить данные пользователя (полная замена))

At the bottom right of the UI, there is an "Authorize" button with a lock icon.

Рисунок 1 – Auth-service

Content-service отвечает за получение url контента (книг в случае реализации только book-service) и поиску контента по тегам. Swagger UI по Content-service представлен на рисунке 2.

content-service 1.0.0 OAS 3.0

/swagger.json
ISC

Servers / Authorize

Content

- POST** /content Создать новый контент
- GET** /content Получить список всего контента
- GET** /content/{content_id} Получить контент по ID
- PUT** /content/{content_id} Обновить контент
- DELETE** /content/{content_id} Удалить контент
- GET** /content/by-tag/{tagName} Получить список контента по тегу

Tags

- POST** /tags Создать новый тег
- GET** /tags Помощь: список всех тегов

Рисунок 2 – Content-service

book-service 1.0.0 OAS 3.0

/swagger.json
ISC

Servers / Authorize

Book Metadata

- POST** /books Создать метаданные книги
- GET** /books Получить список всех метаданных книг
- GET** /books/{content_id} Получить метаданные книги по content_id
- PUT** /books/{content_id} Обновить метаданные книги
- DELETE** /books/{content_id} Удалить метаданные книги

Book Publishers

- POST** /book-publishers Создать издателя книги
- GET** /book-publishers Получить список всех издателей книг
- GET** /book-publishers/{id} Помощь: список всех издателей книг

Рисунок 3 – Book-service

Рисунок 4 – Контент

Вывод

В ходе лабораторной работы выполнено требуемое преобразование backend приложения в микросервисную архитектуру: выделены самостоятельные модули, API разделено на 3 доменных микросервиса, настроено сетевое взаимодействие между сервисами через API Gateway, Docker-сеть и HTTP-вызовы. Полученная архитектура обеспечивает лучшую модульность, масштабируемость и удобство дальнейшей разработки.