

**САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ  
ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО**

**Дисциплина:** Бэк-энд разработка

**Отчет**

**Лабораторная работа №5**

**Выполнил:**

**Захарчук Александр**

**К3441**

**Проверил:  
Добряков Д. И.**

**Санкт-Петербург**

**2026 г.**

## **Задача**

- Выделить самостоятельные модули в вашем приложении;
- Провести разделение своего API на микросервисы (минимум, их должно быть 3);
- Настроить сетевое взаимодействие между микросервисами.

## **Ход работы**

Монолитный сервис для управления рецептами из прошлой лабораторной работы был разделен на 3 микросервиса: user-service, recipe-service, social-service.

### **User Service**

Отвечает за регистрацию, аутентификацию, авторизацию и управление пользователями.

Основные функции:

- Регистрация и вход пользователя.
- Обновление профиля и удаление аккаунта.
- Валидация JWT токенов по запросу других сервисов.

### **Recipe Service**

Управляет рецептами и связанными с ними ингредиентами.

Основные функции:

- CRUD-операции для рецептов.
- Поддержка категорий (тип блюда, уровень сложности).
- Ассоциация рецептов с ингредиентами.

### **Social Service**

Обрабатывает социальные действия, связанные с рецептами: лайки, комментарии, а также подписки пользователей.

Основные функции:

- Добавление, удаление и получение лайков.
- Комментирование рецептов (создание, чтение, обновление, удаление).

- Подписка и отписка от пользователей.
- Получение списка подписок и подписчиков.

В результате разделения сервисы стали независимыми, они имеют изолированный код, а также индивидуальную базу данных для каждого из них.

**Users**

- GET** /users Get all users
- GET** /users/{id} Get user by ID
- PATCH** /users/{id} Update user profile
- DELETE** /users/{id} Delete user by ID
- POST** /users/register Register a new user
- POST** /users/login Login user and return token
- POST** /users/validate Validate user token

**Subscriptions**

- POST** /subscriptions Subscribe to another user
- GET** /subscriptions Get users followed by a current user
- DELETE** /subscriptions/{id} Unsubscribe from a user

Рисунок 1 - Swagger для user-service

**Ingredients**

- GET** /ingredients Get all ingredients
- POST** /ingredients Create new ingredient
- GET** /ingredients/{id} Get ingredient by ID
- PATCH** /ingredients/{id} Update ingredient by ID
- DELETE** /ingredients/{id} Delete ingredient by ID

**recipes**

- GET** /recipes Get all recipes
- POST** /recipes Create new recipe
- GET** /recipes/{id} Get recipe by ID
- PATCH** /recipes/{id} Update recipe by ID
- DELETE** /recipes/{id} Delete recipe by ID

Рисунок 2 - Swagger для recipe-service

Likes

POST /likes Create a new like

GET /likes Get all likes by user ID

DELETE /likes/{id} Delete like by ID

Comments

POST /comments Create a new comment

GET /comments/{id} Get comment by ID

PATCH /comments/{id} Update a comment

DELETE /comments/{id} Delete a comment

GET /comments/recipe/{recipeId} Get all comments for a recipe

Рисунок 3 - Swagger для social-service

## Вывод

В ходе работы был произведён переход от монолитной архитектуры к микросервисной. Приложение было разделено на отдельные сервисы: управление пользователями, рецептами, подписками и социальными взаимодействиями (лайки, комментарии, сохранения). Каждый микросервис стал независимым, с собственной базой данных и четко определёнными API. Это повысило масштабируемость, упростило поддержку и тестирование, а также обеспечило гибкость в дальнейшем развитии проекта.