САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО

Дисциплина: Бэк-энд разработка

Отчет

Лабораторная работа №3

Выполнил:

Соловьева П.А.

Группа К3344

Проверил:
Добряков Д. И.

Санкт-Петербург

2025 г.

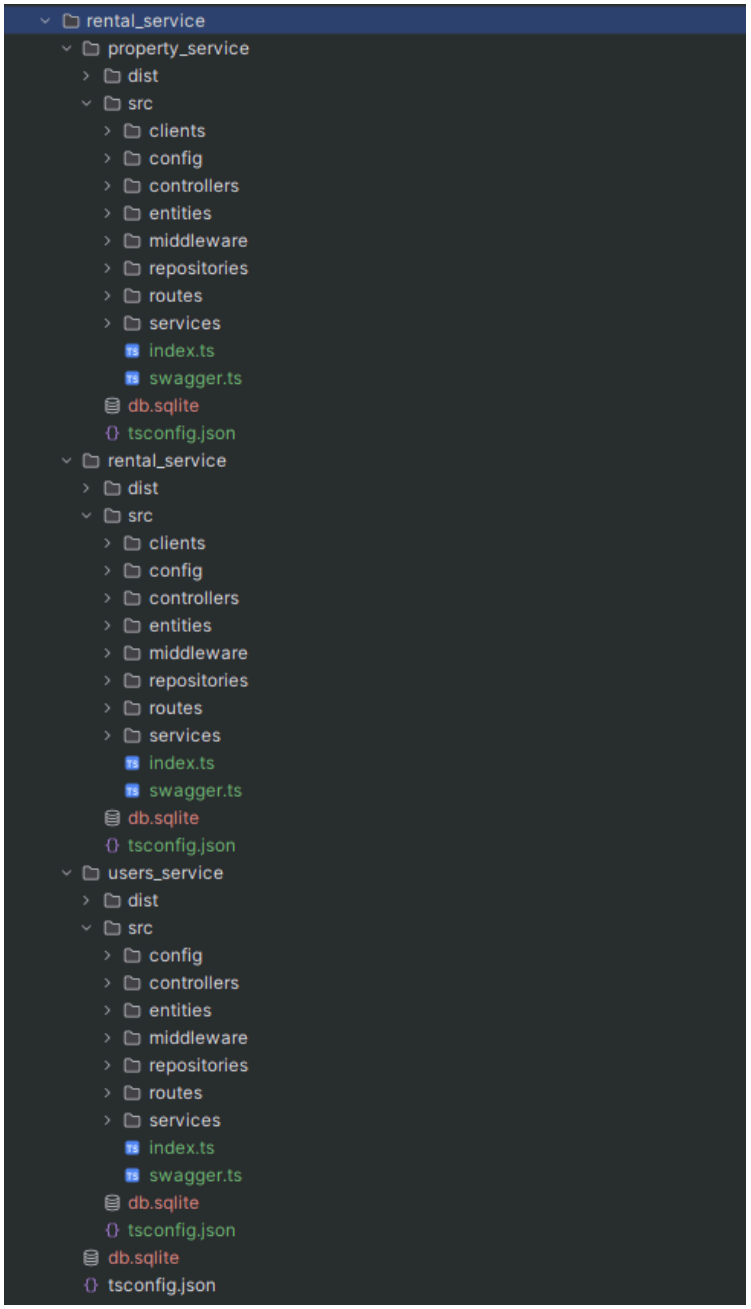
Задача

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

Ход работы

1. Структура проекта

Проект был реорганизован на три микросервиса (users_service, rental_service и property_service):



2. Общение между сервисами

Для общения сервисов были использованы HTTP-клиенты с помощью axios

Пример клиента пользователя:

```
import axios from "axios";

class UserClient {
 private baseUrl: string;

 constructor() {
   this.baseUrl =
     process.env.USERS_SERVICE_URL || "http://users-service:5001/api/users";
 }

 async getUserById(userId: string, token: string) {
   try {
     const response = await axios.get(`${this.baseUrl}/${userId}`, {
       headers: { Authorization: `Bearer ${token}` },
     });
     return response.data;
   } catch (error: any) {
     console.error("Error fetching user:", error.response?.data ||
error.message);
     throw new Error("Failed to fetch user");
   }
 }

 async createUser(userData: any, token: string) {
   try {
     const response = await axios.post(this.baseUrl, userData, {
       headers: { Authorization: `Bearer ${token}` },
     });
     return response.data;
   } catch (error: any) {
     console.error("Error creating user:", error.response?.data ||
error.message);
     throw new Error("Failed to create user");
   }
 }
}

export const userClient = new UserClient();
```

Вывод

В ходе выполнения задачи проект был успешно реорганизован с целью выделения самостоятельных модулей и разделения API на микросервисы. В результате были реализованы три микросервиса:

1. users_service – отвечает за управление данными пользователей.
2. rental_service – отвечает за управление арендой объектов недвижимости.

3. property_service – отвечает за управление данными объектов недвижимости.

Для взаимодействия между микросервисами использовались HTTP-запросы с помощью библиотеки axios, что позволило организовать сетевое взаимодействие и обмен данными между сервисами.

Разделение на микросервисы повысило модульность проекта, упростило поддержку кода и позволило более гибко масштабировать отдельные компоненты системы. Настроенное сетевое взаимодействие обеспечивает корректное взаимодействие сервисов друг с другом и позволяет интегрировать новые функциональные возможности без значительных изменений в архитектуре приложения.