Министерство науки и высшего образования Российской Федерации ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «Национальный исследовательский университет ИТМО» (Университет ИТМО)

Факультет Инфокоммуникационных технологий (ИКТ)

Образовательная программа Мобильные и сетевые технологии

ОТЧЕТ

Back-end лабораторная работа №3

Тема задания: Микросервисы»

Выполнил: Мамедов Тогрул, группа К33402

Проверил: Добряков Давид Ильич

Задача

Необходимо реализовать отдельный микросервис, выполняющий какую-либо содержательную функцию из всего арсенала функций вашего приложения.

Ход работы:

В проекте было выделено 2 микросервиса: один – для регистрации и авторизации пользователя, второй – для мероприятий и записи на них.

```
    > cerents_microservice
    > cerents_microservice
    > cerents_microservice
```

Клиент отправляет запрос на порт 8000, на котором запущен gateway, он в свою очередь отправляет запросы либо к микросервису пользователей (порт 9091), либо к микросервису мероприятий (порт 9000), в зависимости от искомого URL

Микросервисы, отвечающие за пользователей и мероприятия, функционируют в изоляции и запускаются независимо друг от друга, каждый со своей собственной базой данных. В контроллер записей мероприятий (создание, удаление, получение записей) была внесена дополнительная логика для проверки авторизации пользователей, так как система аутентификации с использованием passport.authenticate находится в другом микросервисе.

```
post = async (request: Request, response: Response) : Promise<void> => {
    const {body : Request } = request // only target event; id comes from token via auth service

    try {
        const auth_resp = await axios({
            url: "localhost:8000/users/profile",
            headers: request.headers
        })
        if (auth_resp.status === 200) {
            body.userId = auth_resp.data.id
            const enroll : UserEnrolledEvent | EnrolledEv... = await this.enrollService.create(body)
            response.status(201).send(enroll)
        } else {
            response.status(auth_resp.status).send()
        }
    } catch (error: any) {
        response.status(400).send({error: error.message})
    }
}
```

В процессе работы я изучил различия между монолитной и микросервисной архитектурами, ознакомился с преимуществами микросервисной архитектуры и понял, почему ее реже используют для небольших приложений. Кроме того, я разделил свой проект на два микросервиса, применяя шлюз для их взаимодействия.