

**САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ  
ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО**

**Дисциплина:** Бэк-энд разработка

Отчет

Лабораторная работа: Микросервисы

Выполнил:

Хайрнасов А.К.

К33402

Проверил:

Добряков Д. И.

Санкт-Петербург

2022 г.

## Задача

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

## Ход работы

Авторизация была вынесена в отдельный микрос.

Добавлена в юзер контроллер валидация токена.

```
93  
94 validateToken = async (request: any, response: any) => {  
95   const {accessToken} = request.body;  
96   try {  
97     // @ts-ignore  
98     response.send({valid: true, 'user': await this.userService.getById(jwt.verify(accessToken, jwtOptions.secretOrKey).id)})  
99   } catch (error: any) {  
100     response.status(401).send({valid: false})  
101   }  
102 }  
103 }
```

Добавлена мидлваре для валидации запросов.

```
6 passport.use(  
7   new Strategy(async (token: any, done: any) => {  
8     axios.post(  
9       `http://auth:8080/users/validate`,  
10      {'accessToken': token}  
11    ).then((resp) => {  
12      if (resp.data.valid) {  
13        done(null, resp.data.user)  
14      } else {  
15        done(null, false)  
16      }  
17    }).catch((error) => {  
18      done(error)  
19    })  
20  })  
21 )  
22  
23 export default passport;
```

Пример использования

```
11
12   router.route('/')
13     .get(passport.authenticate('bearer', { session: false }), controller.getAll)
14
```

И был добавлен прокси сервис

```
5
6   app.use('/users', createProxyMiddleware({target: 'http://auth:8080', changeOrigin: true}))
7   app.use('/currency', createProxyMiddleware({target: 'http://currency:8080', changeOrigin: true}))
8
9   app.listen(8080, () => {
10     console.log('Gateway is running on port 8080. Ara-ara~')
11   })
```

## Вывод

В ходе работы был реализован микросервис, отвечающий за ключевую функцию приложения, другой для аутентификации. Это повысило отказоустойчивость и масштабируемость системы. Применение микросервисной архитектуры позволило снизить сложность кода и улучшить производительность.