

**САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ
ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО**

Дисциплина: Бэк-энд разработка

Отчет

Лабораторная работа 3

Выполнил:

Иконенко Данил

Группа 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

Решено выделить механизм аутентификации и модель пользователя в отдельный микросервис.

Для проверки того, что пользователь авторизован, добавлен новый эндпоинт:

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

Для пользователя сохранена структура API из ЛР №2. Перенесенные эндпоинты проксируются с помощью библиотеки Axios:

```
5     class UserController {
6         private async requestProxy(url: string, method: any, expressResponse: any, body: any = null) {
7             axios({ config: {
8                 method: method,
9                 url: `http://${authConfig.host}:${authConfig.port}${url}`,
10                data: body
11            }).then((resp: AxiosResponse<any>) => {
12                expressResponse.status(resp.status).send(resp.data)
13            }).catch((error) => {
14                expressResponse.status(error.response.status).send(error.response.data)
15            })
16        }
17
18        post = async (request: any, response: any) => {
19            const {body} = request
20            await this.requestProxy(url: '/api/users', method: 'post', response, body)
21        }
```

Middleware для авторизации переписан для создания запроса к микросервису авторизации:

```
7  const customJwtStrategy = new Strategy( verify: async function (token: any, done: any) {
8      axios.post(
9          url: `${authConfig.host}:${authConfig.port}/api/auth/validate`,
10         data: {'accessToken': token}
11     ).then((resp : AxiosResponse<any> ) => {
12         if (resp.status == 200 && resp.data.valid) {
13             const user = resp.data.user
14             done(null, user)
15         } else {
16             done(null, false)
17         }
18     }).catch((error) => {
19         done(error)
20     })
21 })
```

Пример работы перенесенных эндпоинтов:

The screenshot shows a REST client interface with the following details:

- URL:** `http://localhost:5000/api/auth/login`
- Method:** `POST`
- Body:** `{ "email": "foobar@example.com", "password": "12345" }`
- Response:** `{ "accessToken": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZCI6MSwiaWF0IjoxNjU0ODM5OTgyfQ.V2H4Qkgi701jzX1HiV2dTKqQV04bW48-EMzSnt0QDPg", "refreshToken": "f4f2116d-c60b-49ee-8ad9-2a7494a742dc" }`
- Status:** `200 OK`, `45 ms`, `455 B`

http://localhost:5000/api/users/me

GET http://localhost:5000/api/users/me

Params Authorization Headers (9) Body Pre-request Script Tests Settings Cookies

Type Bearer... Token eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ...

Body Cookies Headers (8) Test Results 200 OK 42 ms 339 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "id": 1,
3   "firstName": "foo",
4   "lastName": "bar",
5   "email": "foobar@example.com"
6 }
```

Пример работы эндпоинтов основного сервиса:

http://localhost:5000/api/wallets

POST http://localhost:5000/api/wallets

Params Authorization Headers (9) Body Pre-request Script Tests Settings Cookies Beautify

none form-data x-www-form-urlencoded raw binary GraphQL JSON

```
1 { "name": "some_wallet" }
2
```

Body Cookies Headers (8) Test Results 201 Created 111 ms 403 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "balance": 0,
3   "id": 2,
4   "name": "some_wallet",
5   "userId": 1,
6   "updatedAt": "2022-06-10T05:48:45.334Z",
7   "createdAt": "2022-06-10T05:48:45.334Z"
8 }
```

Вывод

Я разделил монолитное веб-приложение на node.js на два микросервиса, взаимодействующих при помощи API