САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО

Дисциплина: Бэк-энд разработка

Отчет

Лабораторная работа 3

Выполнил:

Иконенко Данил

Группа K33402

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     }
```

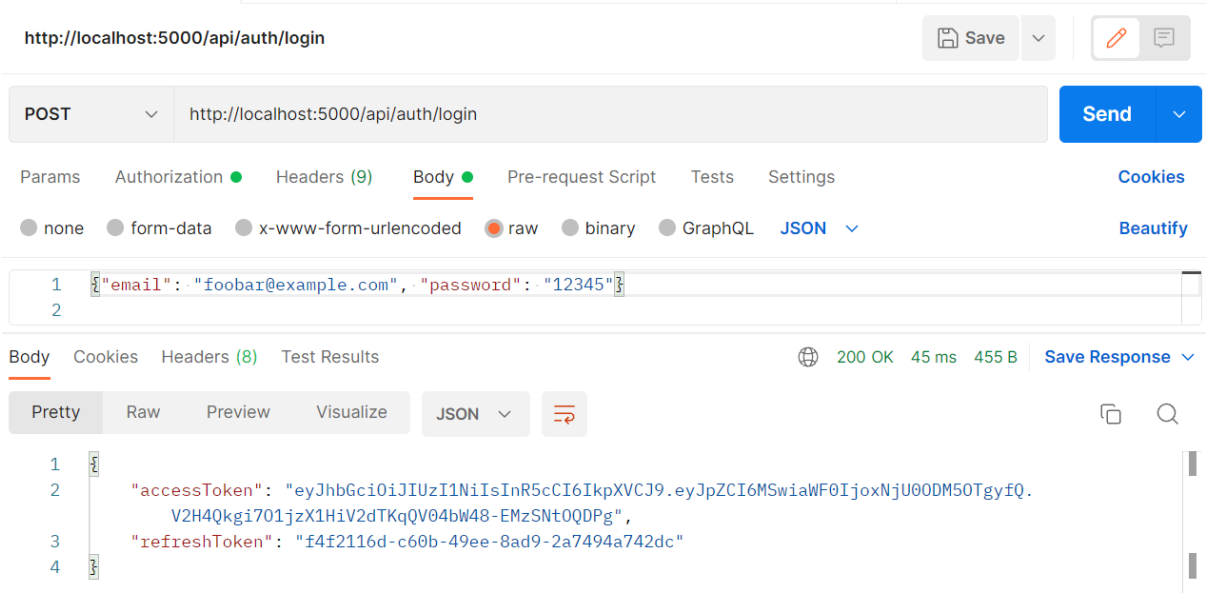
Для пользователя сохранена структура API из ЛР №2. Перенесенные эндпоинты проксируются с помощью библиотеки Axios:

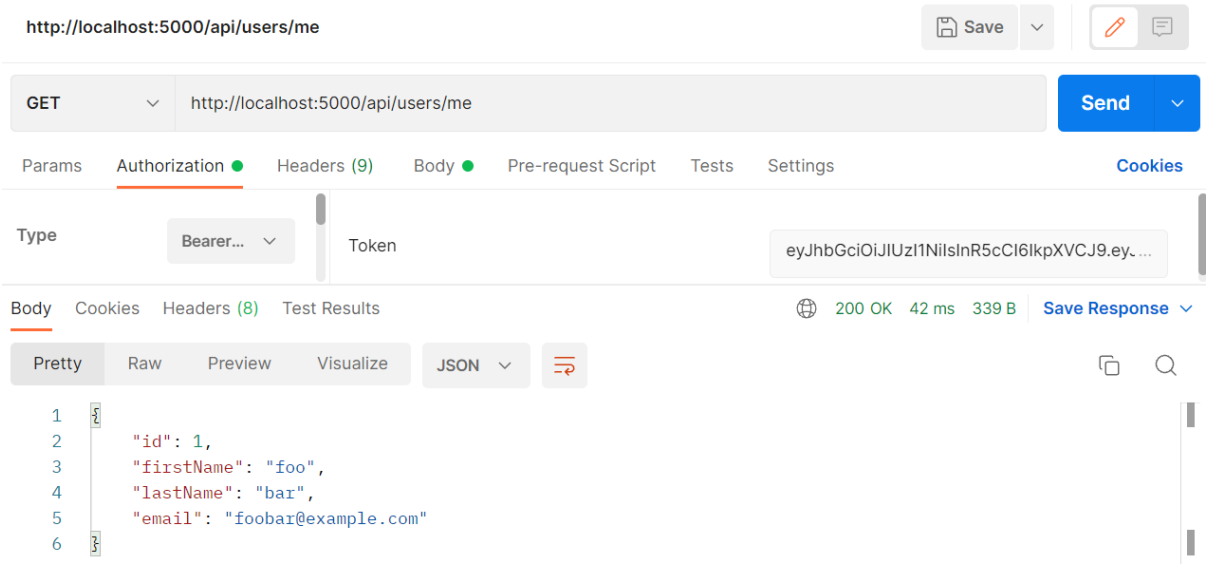
```
5   class UserController {
6       private async requestProxy(url: string, method: any, expressResponse: any, body: any = null) {
7           axios( config: {
8               method: method,
9               url: `http://${authConfig.host}:${authConfig.port}${url}`,
10              data: body
11          }).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:

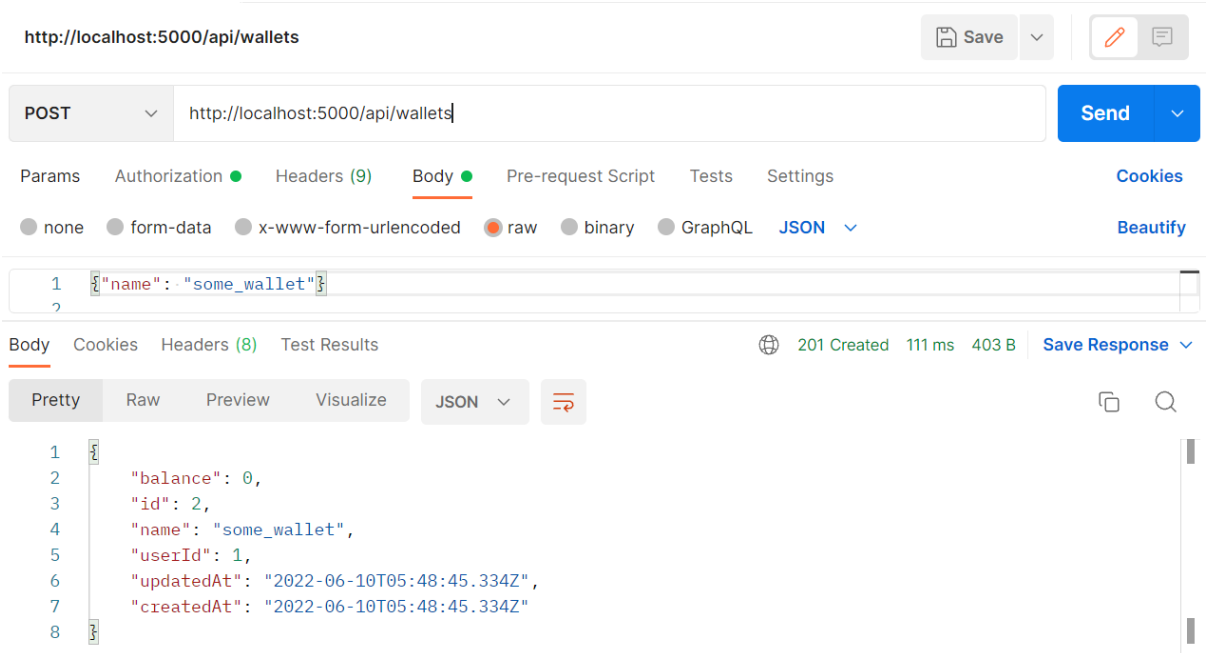
```
const customJwtStrategy = new Strategy( verify: async function (token: any, done: any) {
    axios.post(
        url: `http://${authConfig.host}:${authConfig.port}/api/auth/validate`,
        data: {'accessToken': token}
    ).then((resp : AxiosResponse<any> ) => {
        if (resp.status == 200 && resp.data.valid) {
            const user = resp.data.user
            done(null, user)
        } else {
            done(null, false)
        }
    }).catch((error) => {
        done(error)
    })
})
```

Пример работы перенесенных эндпоинтов:





Пример работы эндпоинтов основного сервиса:



Вывод

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