

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

Факультет Инфокоммуникационных технологий

Образовательная программа 09.03.03 Мобильные и сетевые технологии

О Т Ч Е Т

по лабораторной работе № 3
по дисциплине «Бэк-энд разработка»

Выполнил:

Мосин З. И.

Проверил:

Добряков Д. И.



УНИВЕРСИТЕТ ИТМО

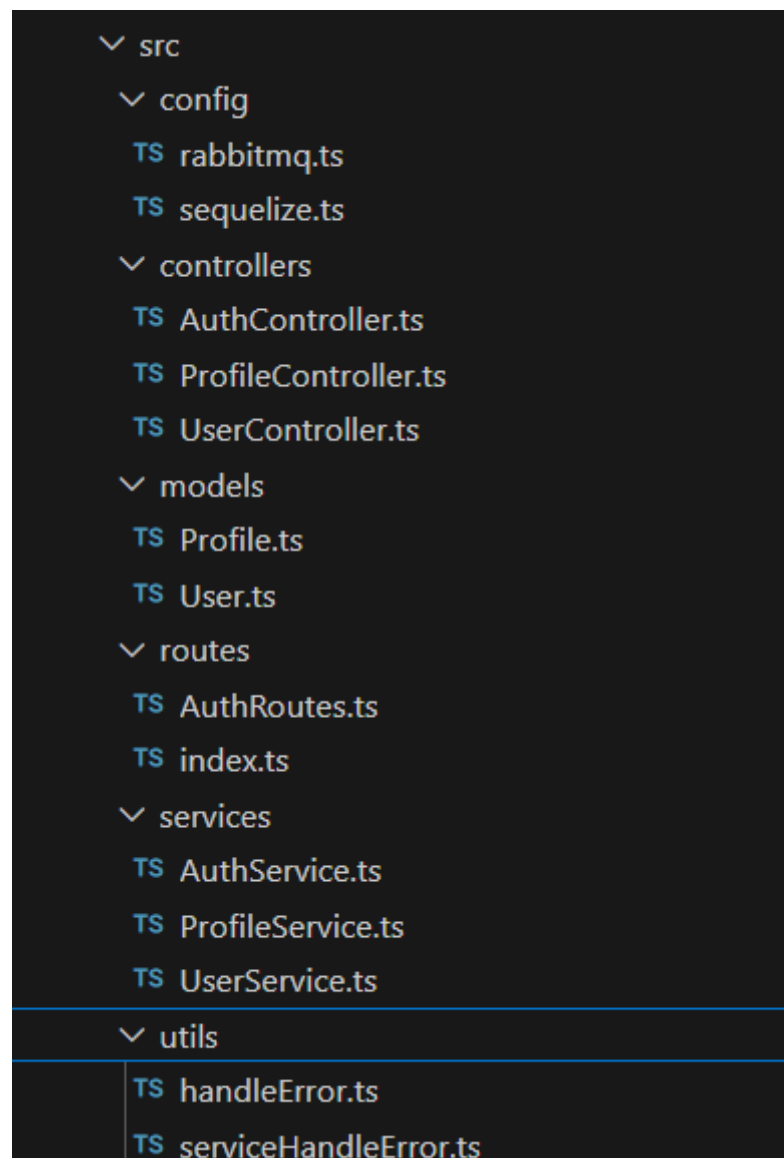
Санкт-Петербург, 2024

Цели работы

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

Ход работы

Для данной работы в качестве микросервера я выбрал функционал авторизации и регистрации. Все связанные с этим части приложения были вынесены в отдельную папку, и запускаются отдельно от остального кода.



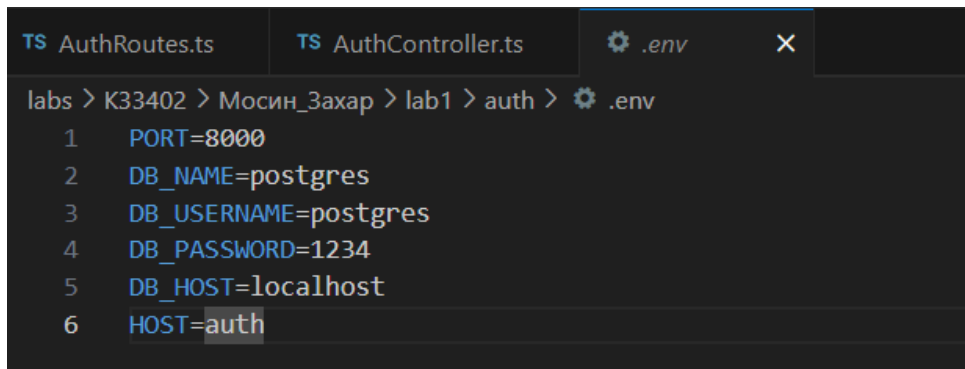
В этой части приложения осуществляется работа с пользователем, профилем, а также токенами, которые выдаются при авторизации.

```
TS AuthRoutes.ts X
labs > K33402 > Мосин_Захар > lab1 > auth > src > routes > TS AuthRoutes.ts > [E] userRoutes
1  import express from 'express'
2
3  import AuthController from '../controllers/AuthController'
4  import ProfileController from '../controllers/ProfileController'
5  import UserController from '../controllers/UserController'
6
7  const userRoutes = express.Router()
8
9  userRoutes.post('/reg', AuthController.reg)
10 userRoutes.post('/login', AuthController.login)
11 userRoutes.post('/verify', AuthController.verify)
12
13 userRoutes.get('/profile', ProfileController.getProfileById)
14 userRoutes.post('/profile', ProfileController.createOrUpdateProfile)
15
16 userRoutes.get('/user/:id', UserController.getUserById)
17 userRoutes.post('/user', UserController.createUser)
18 userRoutes.delete('/user/:id', UserController.deleteUser)
19
20 export default userRoutes
21
```

Токены выдаются с помощью JsonWebToken и находятся в заголовках всех последующих запросов

```
TS AuthRoutes.ts TS AuthController.ts TS AuthService.ts X
labs > K33402 > Мосин_Захар > lab1 > auth > src > services > TS AuthService.ts > AuthService > login
11 class AuthService {
35
36  static async login(email: string, password: string) {
37    try {
38      const user = await userRepository.findOne({ where: { email } })
39      if (!user) {
40        return serviceHandleError({ message: 'Некорретный адрес почты' })
41      }
42
43      const passwordMatch = user.checkPassword(password)
44      if (!passwordMatch) {
45        return serviceHandleError({ message: 'Некорректный пароль' })
46      }
47
48      const key = process.env.JWT_SECRET_KEY || 'testkey1'
49      const token = jwt.sign({ userId: user.id }, key, {
50        expiresIn: '1h',
51      })
52
53      return token
54    } catch (error) {
55      return serviceHandleError({
56        error,
57        message: 'Ошибка при авторизации пользователя',
58      })
59    }
60  }
61}
```

Также в директории микросервиса авторизации и регистрации содержатся свои переменные окружения, нужные для взаимодействия с другими частями приложения.



The screenshot shows a code editor with two tabs: 'TS AuthRoutes.ts' and 'TS AuthController.ts'. A third tab, '.env', is active and shows the following content:

```
labs > K33402 > Мосин_Захар > lab1 > auth > .env
1  PORT=8000
2  DB_NAME=postgres
3  DB_USERNAME=postgres
4  DB_PASSWORD=1234
5  DB_HOST=localhost
6  HOST=auth
```

Надо сказать, что данный момент наличие нескольких сепарированных частей приложения затрудняет его запуск. Этот аспект будет улучшен в следующей лабораторной работе с помощью Docker-compose.

ВЫВОД

В процессе работы вынесли часть функционала приложения в микросервис, тем самым разделив его логику, и научились работать с приложениями, состоящими из нескольких отдельных частей.