САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО

Дисциплина: Фронд-энд разработка

Отчет

Лабораторная работа 3

Выполнил:
Комиссаров Александр
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

Вынесем аутентификацию в отдельный микросервис «auth». Остальное приложение останется в микросервисе «main».

Рассмотрим микросервис «auth». Приложение запускается на порту 8100

```
export default class App {
  public port: number
  public host: string

  private app: express.Application
  private server: Server

  constructor(port = 8100, host = "localhost") {
      this.port = port
      this.host = host

      this.app = this.createApp()
      this.server = this.createServer()
  }

  private createApp(): express.Application {
      const app = express()
      const bodyParser = require('body-parser')
      app.use(bodyParser.urlencoded({ extended: false }))
      app.use(bodyParser.json())
      app.use('/v1', routes)
      return app
    }
```

Контроллер в свою очередь для получения почты и пароля пользователя обращается к порту 8000

```
import axios from "axios"

export default class AuthController {

    post = async (request: any, response: any) => {
        const { email, password } = request.body
        const secretOrKey = 'test123'
        if (email && password) {
            let user = await axios.get("http://localhost:8000/v1/user", {
                params: {
                    email: email
                }
            })
            if (!user) {
                response.status(401).json({ msg: 'No such user found', user })
            }
            if (user!!.data.password === password) {
                let payload = { id: user!.data.id }
                const jwt = require('jsonwebtoken')
                let token = jwt.sign(payload, secretOrKey)
                response.json({ msg: 'ok', token: token })
            } else {
                response.status(401).json({ msg: 'Password is incorrect' })
            }
        }
    }
}
```

Роут для аутентификации также перенесен в новый микросервис

```
import express from "express"
import AuthController from "../controllers/auth"

const router: express.Router = express.Router()

const authController = new AuthController()

router
  .route('/auth')
  .post(authController.post)

export default router
```

Вывод

В результате проделанной работы функционал нашего приложения был разделен на два микросервиса. В отдельный микросервис был вынесен механизм аутентификации.