САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО

Дисциплина: Бэк-энд разработка

Отчет

Лабораторная работа 3: Микросервисы

Выполнила:

Никифорова Кюннэй
Группа К33402

Проверил:
Добряков Д. И.

Санкт-Петербург

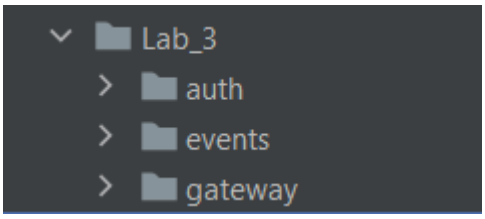
2023 г.

Задача

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

Ход работы

1) Структура проекта:



Я разделила приложение на 2 микросервиса *(auth* для работы с пользователями и *events* для работы с мероприятиями) и добавила общую точку входа *gateway*, перенаправляющую запросы по нашим микросервисам.

2) gateway:

```
import express from "express"
import proxy from "express-http-proxy"

const app = express();

app.use('/auth', proxy('http://localhost:8001'))
app.use('/events', proxy('http://localhost:8002'))

app.listen(8000, () => {
  console.log(`Running server on port 8000`)
})
```

3) auth:

```
import express from "express"
import { UserController } from "../../controllers/auth"
import passport from "../../middlewares/passport"

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

const controller: UserController = new UserController()

userRoutes.route('/')
    .post(controller.post)
```

```
userRoutes.route('/profile')
    .get(passport.authenticate('jwt', { session: false }), controller.me)

userRoutes.route('/profile/:id')
    .get(controller.get)

userRoutes.route('/login')
    .post(controller.auth)

userRoutes.route('/delete/:id')
    .delete(controller.delete)

userRoutes.route('/accessToken')
    .post(controller.accessToken)

export default userRoutes
```

4) events:

```
import express from "express"
import { EventController } from "../controllers/Event"

const routes: express.Router = express.Router()
const controller: EventController = new EventController()

routes.route('/filter')
    .get(controller.filter)

routes.route('/')
    .get(controller.calendar)

routes.route('/author/:id')
    .get(controller.author)

routes.route('/:id')
    .get(controller.get)

export default routes
```

5) EventSettings:

```
import express from "express"
import { EventSettingsController } from "../controllers/EventSettings"
import passport from "../middlewares/passport";

const routes: express.Router = express.Router()
const controller: EventSettingsController = new EventSettingsController()

routes.route('/create')
    .post(passport.authenticate('bearer', { session: false }), controller.post)
```

```
routes.route('/update/:id')
    .patch(passport.authenticate('bearer', { session: false }), controller.update)

routes.route('/delete/:id')
    .delete(passport.authenticate('bearer', { session: false }),
controller.delete)

routes.route('/join')
    .post(passport.authenticate('bearer', { session: false }), controller.join)

routes.route('/my')
    .get(passport.authenticate('bearer', { session: false }), controller.get)

export default routes
```

Вывод

В ходе данной лабораторной работы был реализован отдельный микросервис, выполняющий содержательную функцию из всего арсенала функций приложения.