

**САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ  
ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО**

**Дисциплина:** Бэк-энд разработка

Отчет

Лабораторная работа 3: Микросервисы

Выполнил:

Конев Антон

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

Структура частей приложения

Auth

Gateway

Currency

```
|—configs
|—controllers
|   |—users
|—core
|—errors
|   |—users
|—middlewares
|—models
|   |—auth
|   |—users
|—providers
|—routes
|   |—v1
|       |—users
|—services
|   |—auth
|   |—users
|—utils
```

```
C:.\
|—dist
|—src
```

```
|—configs
|—controllers
|   |—chart
|   |—currency
|   |—portfolio
|—core
|—errors
|   |—chart
|   |—currency
|   |—portfolio
|—middlewares
|—models
|   |—currency
|   |—portfolio
|—providers
|—routes
|   |—v1
|       |—chart
|       |—currency
|       |—portfolio
|—services
|   |—chart
|   |—currency
|   |—portfolio
```

Для проверки авторизации был добавлен новый эндпоинт

```
router.route( prefix: '/validate')
    .post(controller.validateToken);
```

```
validateToken = async (request: any, response: any) : Promise<void> => {
    const {accessToken} = request.body;
    try {
        const payload : string | jwt.JwtPayload = jwt.verify(accessToken, jwtOptions.secretOrKey);
        // @ts-ignore
        const user : User = await this.userService.getById(payload.id)
        response.send({'valid': true, 'user': user})
    } catch (error: any) {
        response.status(401).send({'valid': false})
    }
}
```

Для проверки авторизации в приложении криптовалюты мы отправляем запрос в микросервис авторизации посредством нового Middleware

```
const customJwtStrategy : Strategy<VerifyFunctions> = new Strategy( verify: async (token: any, done: any) : Promise<void> => {
    axios.post(
        url: 'http://auth:8001/users/validate',
        data: {'accessToken': token}
    ).then((resp : AxiosResponse<any, any> ) : void => {
        if (resp.data.valid) {
            const user = resp.data.user
            done(null, user)
        } else {
            done(null, false)
        }
    }).catch((error) : void => {
        done(error)
    })
})
```

Также для проксирования запросов был реализован Gateway

```
const app = express();

app.use('/users', createProxyMiddleware({target: 'http://auth:8001', changeOrigin: true}))
app.use('/currency', createProxyMiddleware({target: 'http://currency:8002', changeOrigin: true}))

app.listen(8000, () : void => {
    console.log('Gateway is running on port 8000')
})
```

Эндпоинты сервиса криптовалюты теперь выглядят следующим образом

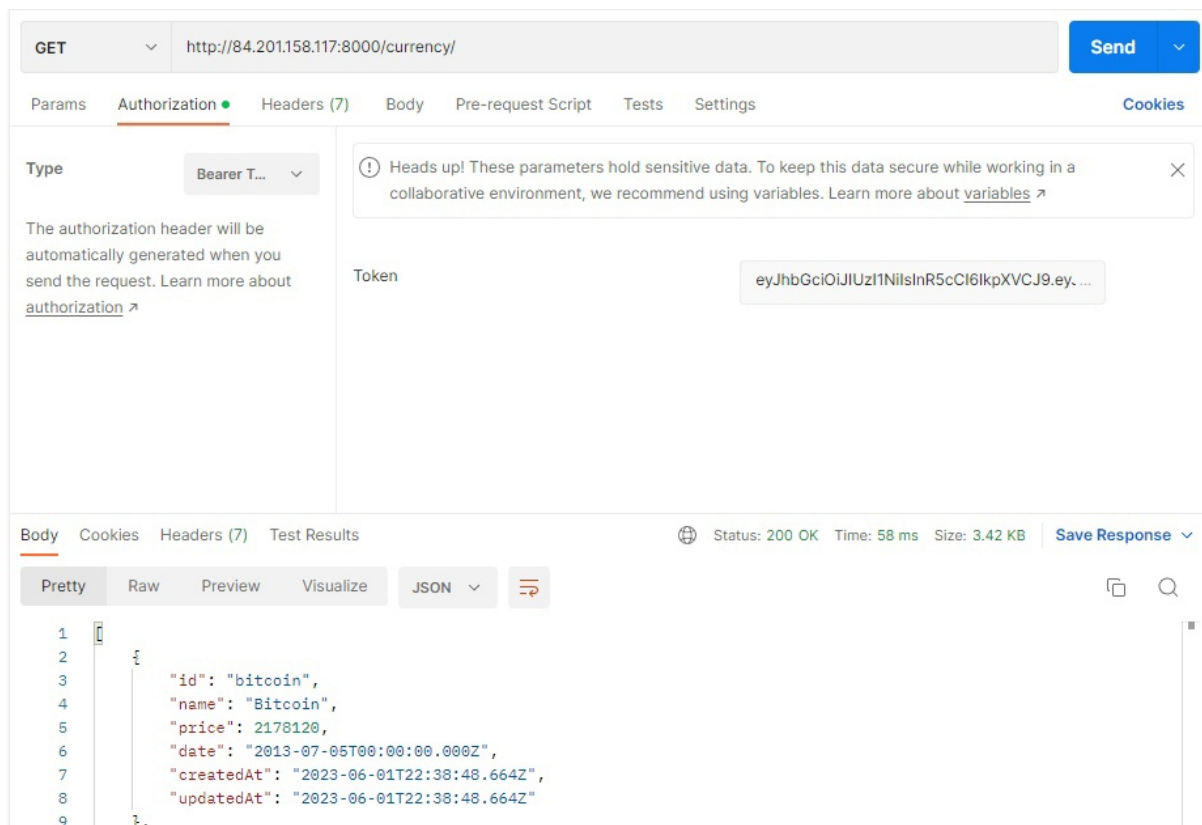
```
router.route( prefix:('/:id')
  .get(passport.authenticate( strategy: 'bearer', options: { session: false })),controller.findByUserId)

router.route( prefix: '/buy')
  .post(passport.authenticate( strategy: 'bearer', options: { session: false })),controller.buyCurrency)

router.route( prefix: '/sell')
  .post(passport.authenticate( strategy: 'bearer', options: { session: false })),controller.sellCurrency)
```

## Тестирование в Postman

### Валидный токен



## Протухший токен

**GET**  Send

Params Authorization Headers (7) Body Pre-request Script Tests Settings Cookies

Type Bearer T... ⌵

The authorization header will be automatically generated when you send the request. Learn more about [authorization](#).

Token  
nyJhbCciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZCI6ImSwiaWF0IjoxNjU5MDg2fQ.wWIDYmz84uq8It7WEv21cDwPxz1BU8SIS0rth85\_gTM

Body Cookies Headers (8) Test Results Status: 500 Internal Server Error Time: 103 ms Size: 943 B Save Response

Pretty Raw Preview Visualize HTML ⌵

```

1 <!DOCTYPE html>
2 <html lang="en">
3
4 <head>
5   <meta charset="utf-8">
6   <title>Error</title>
7 </head>
8
9 <body>
10  <pre>AxiosError: Request failed with status code 401<br>    at settle (/currency/node_modules/axios/lib/core/settle.js:19:12)<br>    at IncomingMessage.handleStreamEnd (/currency/node_modules/axios/lib/adapters/http.js:570:11)<br>    at IncomingMessage.emit (node:events:525:35)<br>    at IncomingMessage.emit (node:domain:489:12)<br>    at endReadableNT (node:internal/streams/readable:1359:12)<br>    at processTicksAndRejections (node:internal/process/task_queues:82:21)</pre>
```

## Отсутствие токена

The screenshot shows the Postman interface for an HTTP GET request to `http://84.201.158.117:8000/currency/`. The **Authorization** tab is active, displaying a **Bearer Token** type. A warning message is present: "Heads up! These parameters hold sensitive data. To keep this data secure while working in a collaborative environment, we recommend using variables. Learn more about [variables](#)". The **Token** field is empty. The bottom status bar indicates the response is **401 Unauthorized** with a time of 43 ms and size of 212 B. The response body is shown in the **Body** tab, displaying the text `Unauthorized`.

## **Вывод**

В ходе лабораторной работы монолитный проект из лабораторной работы 2 был разделен на микросервис авторизации, сервис криптовалюты и gateway их проксирующий.