

**САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ  
ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО**

**Дисциплина:** Бэк-энд разработка

Отчет

Домашняя работа №3

Выполнил:

Шурубова Прасковья

Группа: К3343

Проверил:

Добряков Д. И.

Санкт-Петербург

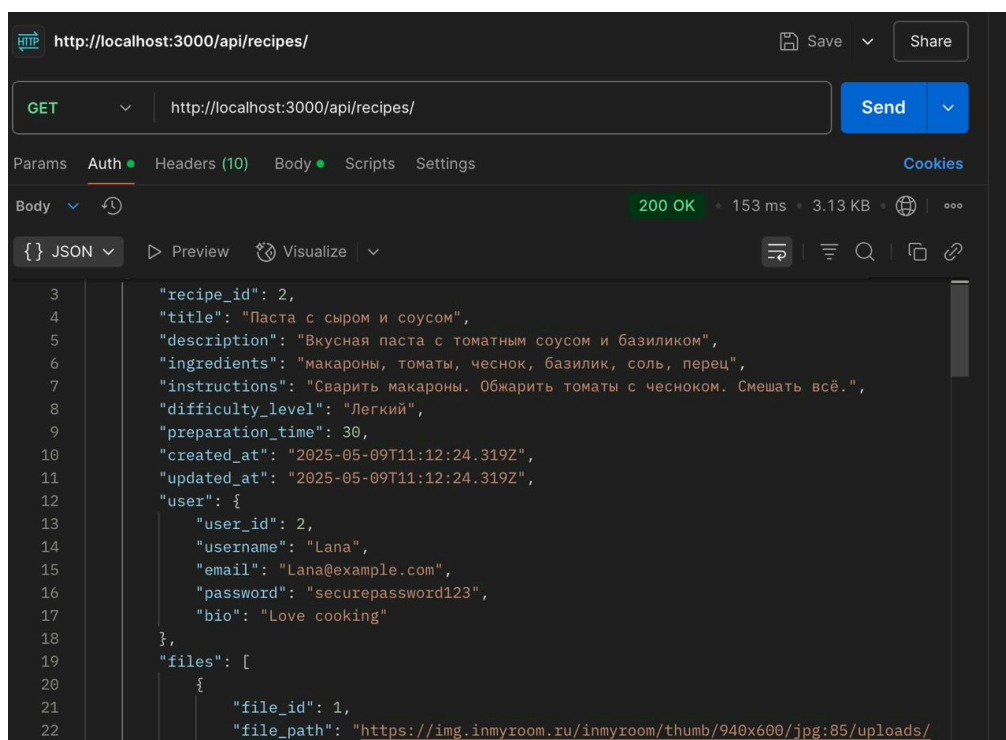
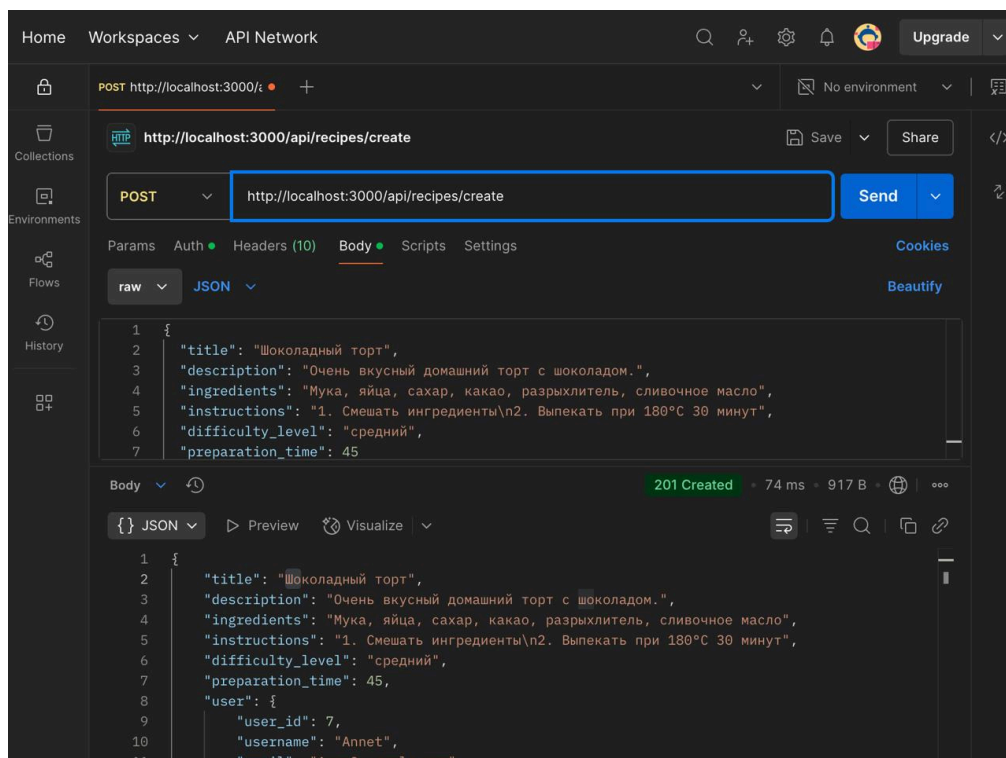
2025 г.

## Задача

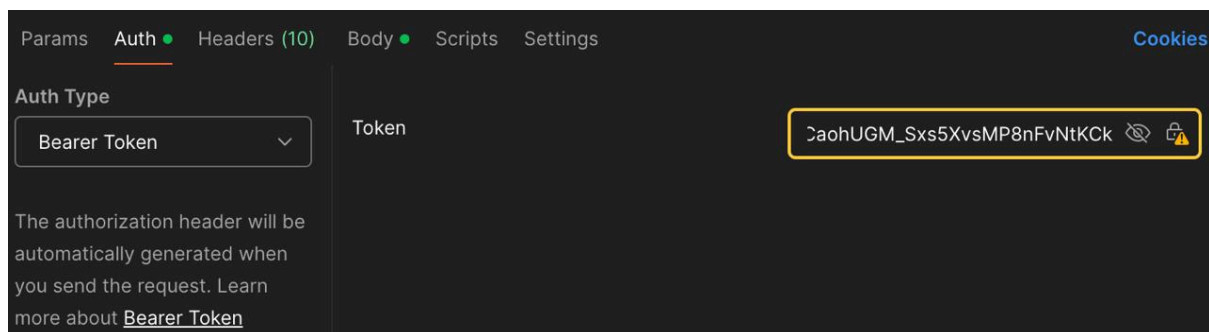
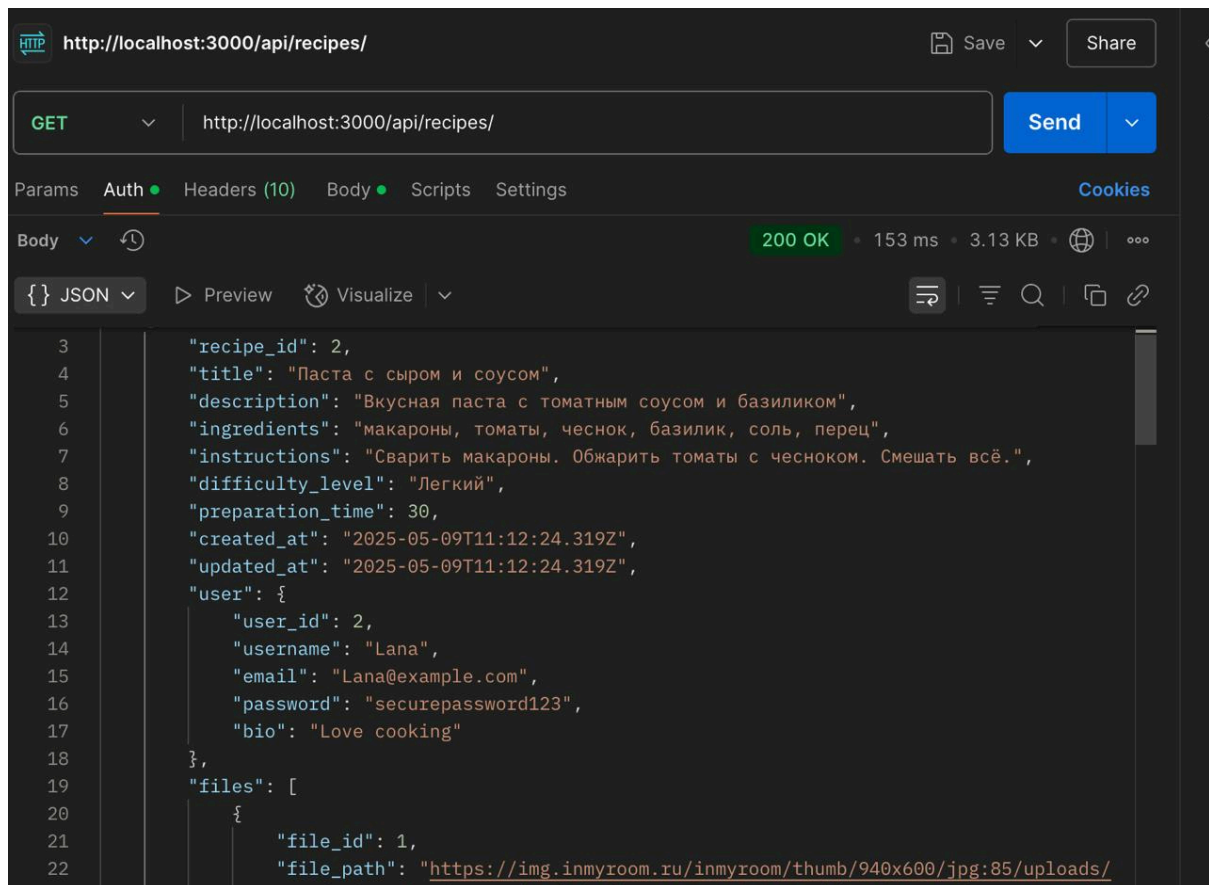
Реализовать автодокументирование средствами swagger, реализовать документацию API средствами Postman.

## Ход работы

Были протестированы все crud-запросы с помощью Postman.



Все запросы кроме get доступны благодаря вход и получению токена. Все операции с favourite доступны только после входа.



Для реализации документирования средствами Swagger был создан файл для настройки и интеграции Swagger-документации в Express-приложение.

```
import swaggerJsdoc from 'swagger-jsdoc';
import swaggerUi from 'swagger-ui-express';
import e from 'express';

const options: swaggerJsdoc.Options = {
  definition: {
    openapi: '3.0.0',
    info: {
      version: '1.0.0',
      description: 'API documentation',
    },
  },
  components: {
    securitySchemes: {
      bearerAuth: {
        type: 'http',
        scheme: 'bearer',
      },
    },
  },
};

const swaggerSpec = swaggerJsdoc(options);
const swaggerUiExpress = swaggerUi(swaggerSpec);

app.use('/api-docs', swaggerUiExpress);
```

```

        bearerFormat: 'JWT',
      },
    },
    security: [{ bearerAuth: [] }],
  },
  apis: ['./src/routes/*.ts'],
};

const swaggerSpec = swaggerJsdoc(options);

export default function setupSwagger(app: e.Application): void {
  {
    app.use('/docs', swaggerUi.serve,
    swaggerUi.setup(swaggerSpec));
  }
}

```

В папке routes хранятся аннотации OpenAPI для каждой сущности. Над каждым маршрутом были прописаны JSDoc-комментарии с тегом `@openapi`, чтобы Swagger мог автоматически прочитать и построить документацию по этим маршрутам.

Пример с получением всех статей:

```

/**
 * @openapi
 * /api/articles:
 *   get:
 *     summary: Получить все статьи
 *     tags:
 *       - Articles
 *     responses:
 *       200:
 *         description: Список статей
 */
router.get("/articles", articleController.getAllArticles);

```

## CREATE

READ

UPDATEDELETE

DELETE

/api/recipes/delete/{id}

Удалить рецепт

🔒 ⬆

Parameters

Cancel

Name	Description
id <span>required</span>	
string	7
(path)	

ExecuteClear

Responses

Curl

```
curl -X 'DELETE' \
  'http://localhost:3000/api/recipes/delete/7' \
  -H 'accept: */*' \
  -H 'Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VzIjp7ImkiOiJ0SLCJlbnRpbGciOiJfswiawFBIjoxNzQ2OTkyOTgyLCJleHAiOiE3NDY5OTY1ODJ9.1p___aY1xMyQcybmCKaKDrqDKWZud3e67gzv' 
```

Request URL

```
http://localhost:3000/api/recipes/delete/7
```

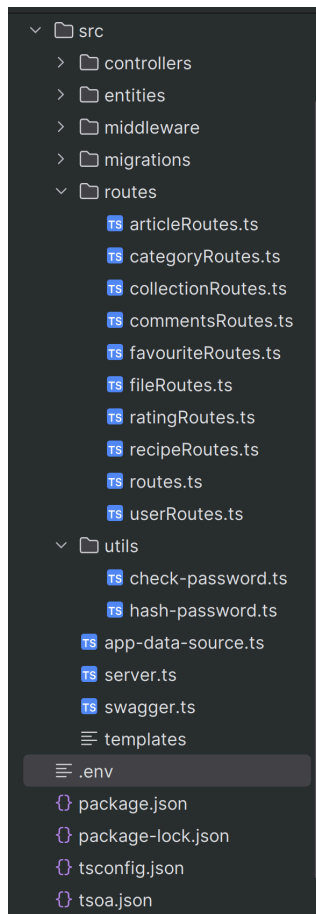
Server response

Code	Details
200	<div>Response body<pre>{   "row": [],   "affected": 1 }</pre></div> <div>Response headers<pre>connection: keep-alive content-length: 23 content-type: application/json; charset=utf-8 date: Sun, 11 May 2025 19:35:22 GMT etag: W/"17-PVQD/EOHAqMH9FRBYNXdiixftCA" keep-alive: timeout=5 x-powered-by: Express</pre></div>

Responses

Code	Description	Links
200	Рецепт удалён	No links
404	Рецепт не найден	No links

Как выглядит структура на данном этапе:



**Вывод:** в данной работе проверила все CRUDы в postman, была создана документация средствами swagger. Также разобралась, какие конфигурационные параметры требуются для автодокументации.