

**САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ  
ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО**

**Дисциплина:** Бэк-энд разработка

Отчет

Практическая работа №4: Документирование API

Выполнил:  
Жаров Александр  
К33402

Проверил:  
Добряков Д. И.

Санкт-Петербург

2024 г.

## Задача

- 1) Сделать документацию API с помощью Swagger

## Ход работы

Для создания документации я решил использовать swagger, который внедрял в проект на nestjs.

- 1) Добавил swagger в зависимости

```
    },
    "devDependencies": {
      "@nestjs/cli": "^9.0.0",
      "@nestjs/schematics": "^9.0.0",
      "@nestjs/swagger": "^7.3.1",
      "@nestjs/testing": "^9.0.0",
      "@types/bcryptjs": "^2.4.6",
      "@types/express": "^4.17.13",
      "@types/jest": "29.2.4",
      "@types/node": "18.11.18",
      "@types/passport-jwt": "^4.0.1",
    },
  },
}
```

- 2) В главном файле подключил его

```
 8  async function bootstrap() {
 9    const config = app.get(ConfigService)
10    const globalPrefix = 'api'
11
12
13    app.useGlobalPipes(new ValidationPipe())
14    app.setGlobalPrefix(globalPrefix)
15
16    const swaggerConfig = new DocumentBuilder()
17      .setTitle('nestjs-prisma-boilerplate')
18      .setDescription('Store application')
19      .setVersion('1.0')
20      .addTag('auth')
21      .addTag('user')
22      .addBearerAuth()
23      .build()
24    const document = SwaggerModule.createDocument(app, swaggerConfig, {
25      deepScanRoutes: true,
26    })
27    SwaggerModule.setup('api', app, document)
28
29    await app.listen(config.PORT)
30    Logger.log(
31      `🚀 Application is running on: http://localhost:\${config.PORT}/\${globalPrefix}`
32    )
33  }
```

- 3) Добавил описание каждого эндпоинта

```

12 @ApiTags('products')
13 @Controller('products')
14 export class ProductController {
15     constructor(private readonly productService: ProductService) {}
16
17     @ApiQuery({ name: 'title', required: false })
18     @ApiQuery({ name: 'minPrice', required: false, type: 'number' })
19     @ApiQuery({ name: 'maxPrice', required: false, type: 'number' })
20     @ApiQuery({
21         name: 'categoryNames',
22         required: false,
23         type: String,
24         isArray: true,
25         description: 'Array of category names separated by commas',
26     })
27     @Get()
28     async getAllProducts(
29         @Query('title') title?: string,
30         @Query('minPrice') minPrice?: string,
31         @Query('maxPrice') maxPrice?: string,
32         @Query('categoryNames') categoryNames?: string[]
33     ): Promise<Product[]> {
34         console.log(categoryNames)

```

#### 4) Проверил получившуюся документацию

The screenshot displays the Swagger UI for an API. At the top, there are two endpoints: a DELETE endpoint for removing a product and a PATCH endpoint for updating a product. Below these, the 'products' section is expanded, showing a GET endpoint at /api/products. This endpoint has four query parameters: 'title' (string), 'minPrice' (number), 'maxPrice' (number), and 'categoryNames' (array of strings). The 'categoryNames' parameter has a description: 'Array of category names separated by commas'. The response is listed as 200 OK. There is a 'Try it out' button and a 'Links' section at the bottom right.

## Вывод

При выполнении работы я подключил к проекту swagger и научился внедрять документацию в бекенд, написанный на nestjs.