САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО

Дисциплина: Бэк-энд разработка

Отчет

Домашняя работа 3

Выполнила:

Красюк Карина

Группа К3341

Проверил:

Добряков Д. И.

Санкт-Петербург

2025 г.

Задача

- реализовать автодокументирование средствами swagger;
- реализовать документацию API средствами Postman.

Ход работы

Реализация в коде:

```
import {
 Body,
 Controller,
 Delete,
 Get,
 Param,
 Post,
 Put,
 UsePipes,
 ValidationPipe,
} from '@nestjs/common';
import {
 ApiTags,
 ApiOperation,
 ApiResponse,
 ApiBearerAuth,
 ApiBody,
 ApiParam,
} from '@nestjs/swagger';
import { UsersService } from './users.service';
import { ParseIntPipe } from '../conception/pipe';
import { CreateUsersDto, TUpdateUsersDto } from './users.dto';
```

```
@ApiTags('Users')
@Controller('users_service')
export class UsersController {
 constructor(private readonly usersService: UsersService) {}

 @Get()
 @ApiOperation({ summary: 'Получить всех пользователей' })
 @ApiResponse({
   status: 200,
   description: 'Список пользователей успешно получен',
 })
 @ApiBearerAuth()
 findAll() {
   return this.usersService.userFindAll();
 }
```

```
 @Get(':id')
 @ApiOperation({ summary: 'Получить пользователя по ID' })
 @ApiResponse({ status: 200, description: 'Пользователь успешно получен' })
 @ApiResponse({ status: 404, description: 'Пользователь не найден' })
 @ApiParam({ name: 'id', type: 'number', description: 'ID пользователя' })
 getUser(@Param('id', ParseIntPipe) id: number) {
   return this.usersService.userGetById(id);
 }

 @Post()
 @UsePipes(new ValidationPipe())
 @ApiOperation({ summary: 'Создать нового пользователя' })
```

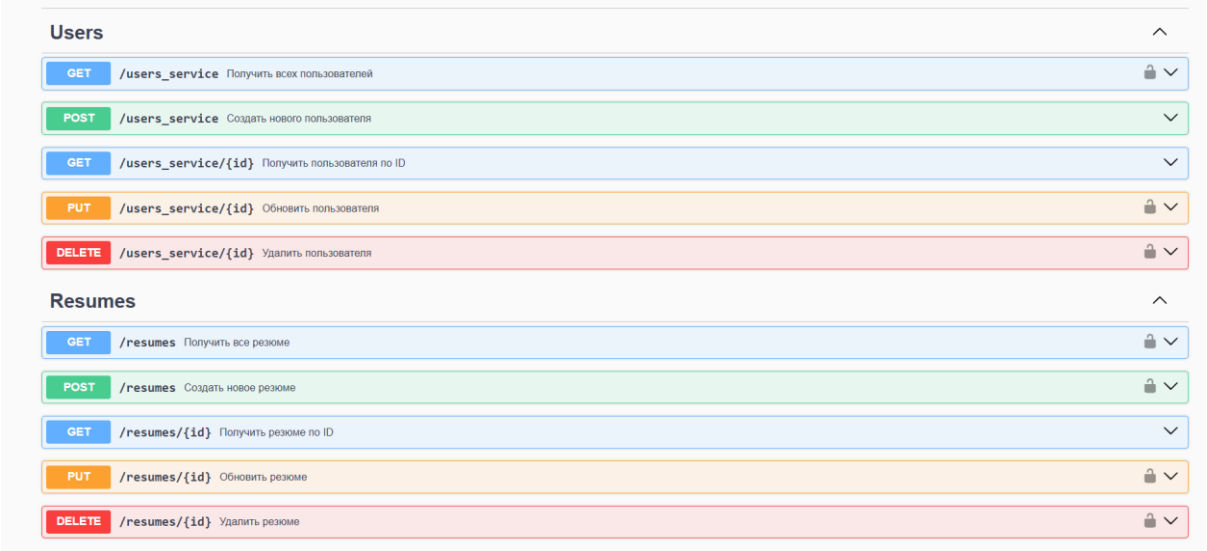
```
@ApiResponse({ status: 201, description: 'Пользователь успешно создан' })
@ApiResponse({ status: 400, description: 'Неверные данные' })
@ApiBody({ type: CreateUsersDto })
create(@Body() dto: CreateUsersDto) {
  return this.usersService.userCreate(dto);
}

@Put(':id')
@UsePipes(new ValidationPipe())
@ApiOperation({ summary: 'Обновить пользователя' })
@ApiResponse({ status: 200, description: 'Пользователь успешно обновлен' })
@ApiResponse({ status: 400, description: 'Неверные данные' })
@ApiResponse({ status: 404, description: 'Пользователь не найден' })
@ApiParam({ name: 'id', type: 'number', description: 'ID пользователя' })
@ApiBearerAuth()
update(@Param('id', ParseIntPipe) id: number, @Body() dto: TUpdateUsersDto) {
  return this.usersService.userUpdate(id, dto);
}

@Delete(':id')
@UsePipes(new ValidationPipe())
@ApiOperation({ summary: 'Удалить пользователя' })
@ApiResponse({ status: 200, description: 'Пользователь успешно удален' })
@ApiResponse({ status: 404, description: 'Пользователь не найден' })
@ApiParam({ name: 'id', type: 'number', description: 'ID пользователя' })
@ApiBearerAuth()
delete(@Param('id', ParseIntPipe) id: number) {
  return this.usersService.userDelete(id);
}
```

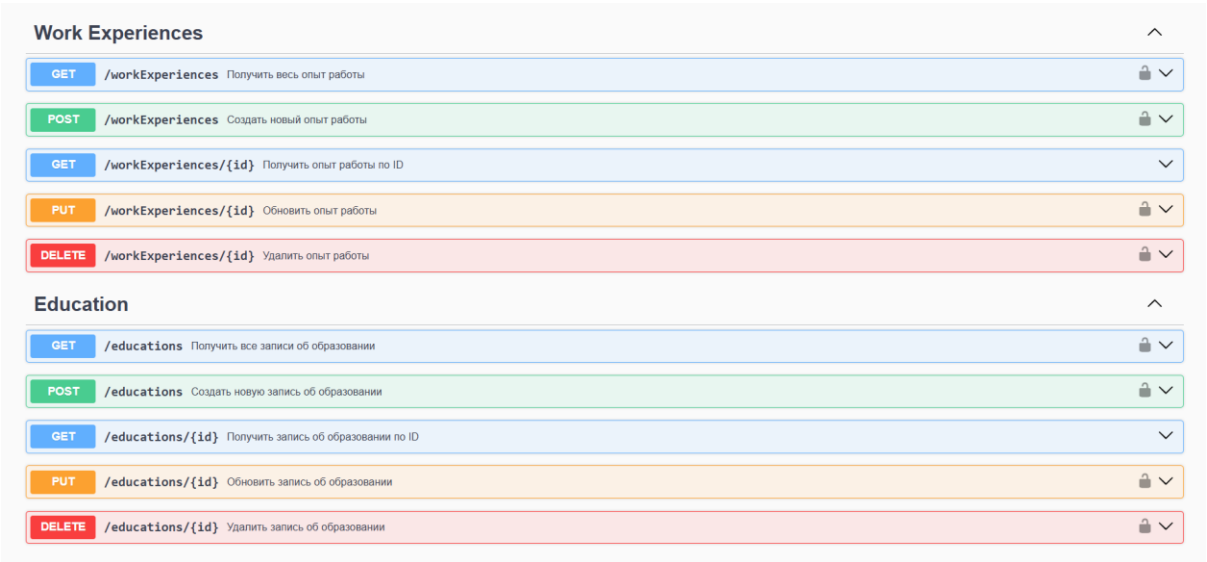
```
}

import { NestFactory } from '@nestjs/core';

import { AppModule } from './app.module';

import { ValidationPipe } from '@nestjs/common';

import { SwaggerModule, DocumentBuilder } from '@nestjs/swagger';



async function bootstrap() {

   const app = await NestFactory.create(AppModule);



   app.useGlobalPipes(new ValidationPipe());
```

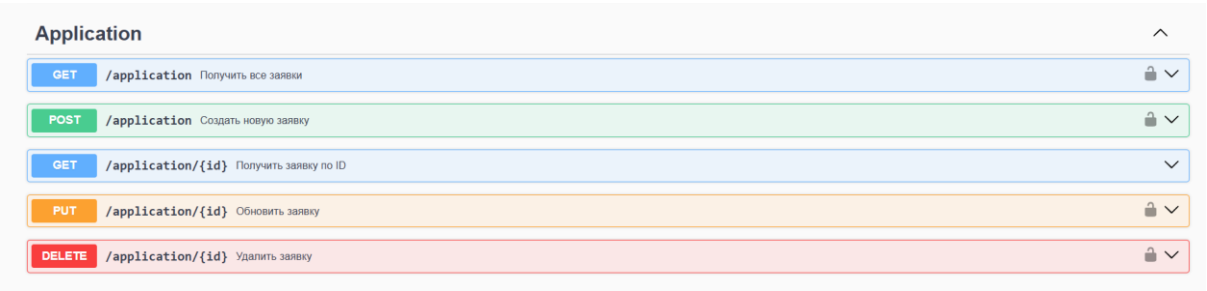
```
   const config = new DocumentBuilder()

       .setTitle('User and Resume Service')

       .setDescription('API documentation for managing users and resumes')

       .setVersion('1.0')

       .addTag('user', 'User Management')

       .addTag('resume', 'Resume Management')

       .addTag('application', 'Job Applications')

       .build();

   const document = SwaggerModule.createDocument(app, config);

   SwaggerModule.setup('api/docs', app, document);



   await app.listen(3001);

}

bootstrap();
```

Документация в Swager

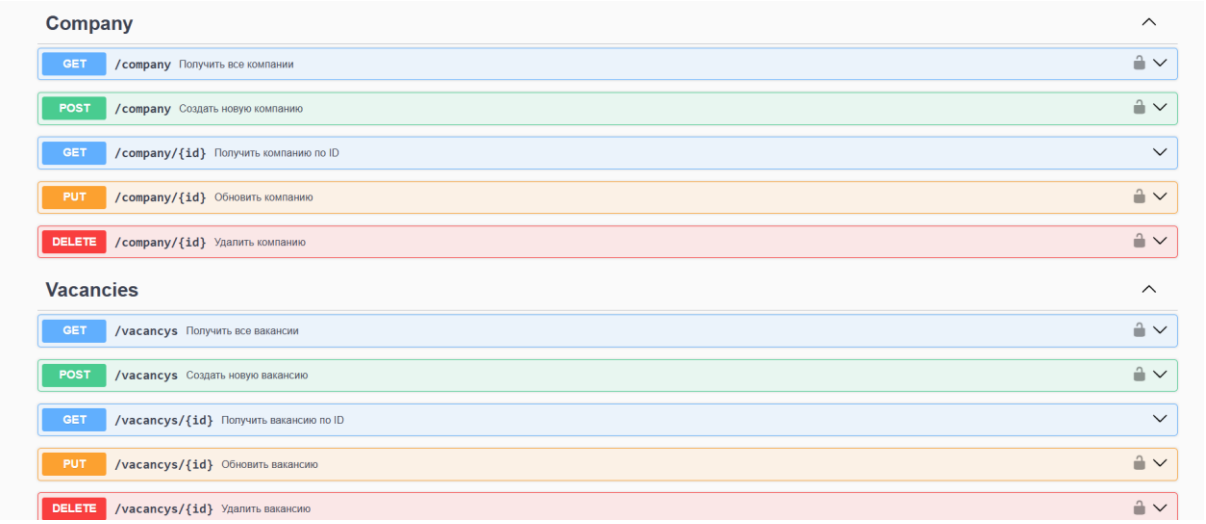
User сервис



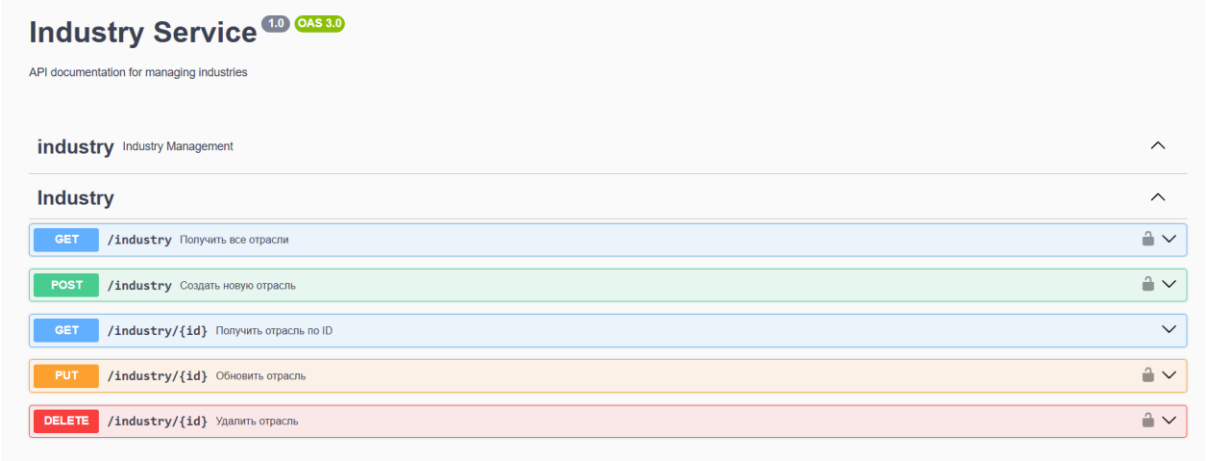




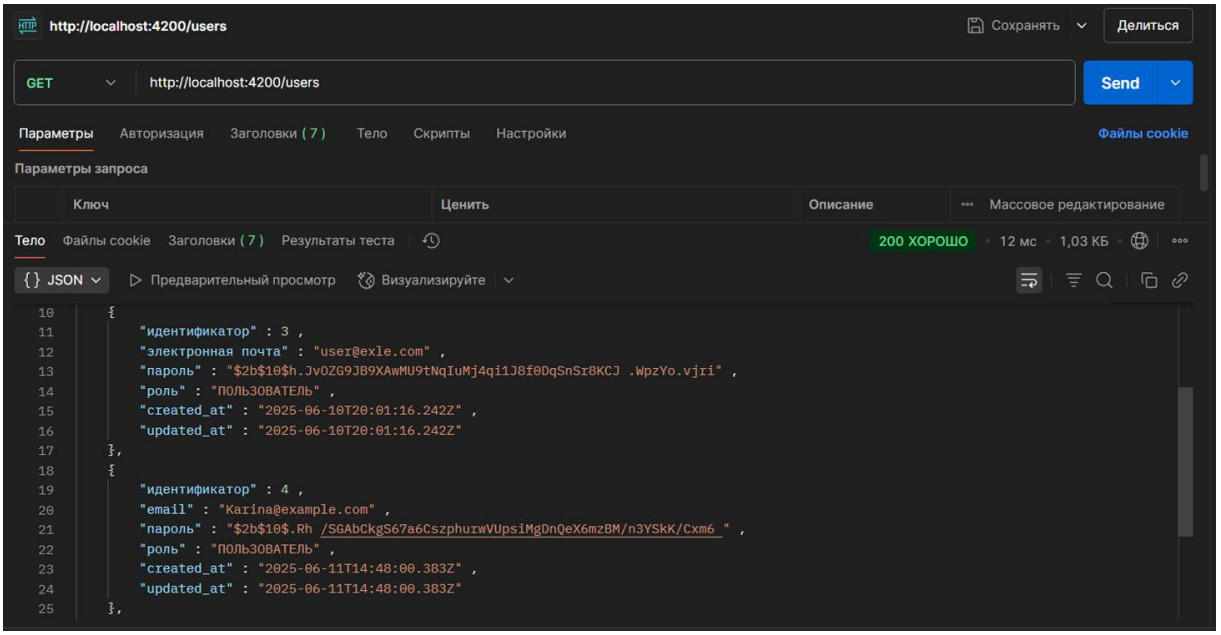
Company сервис

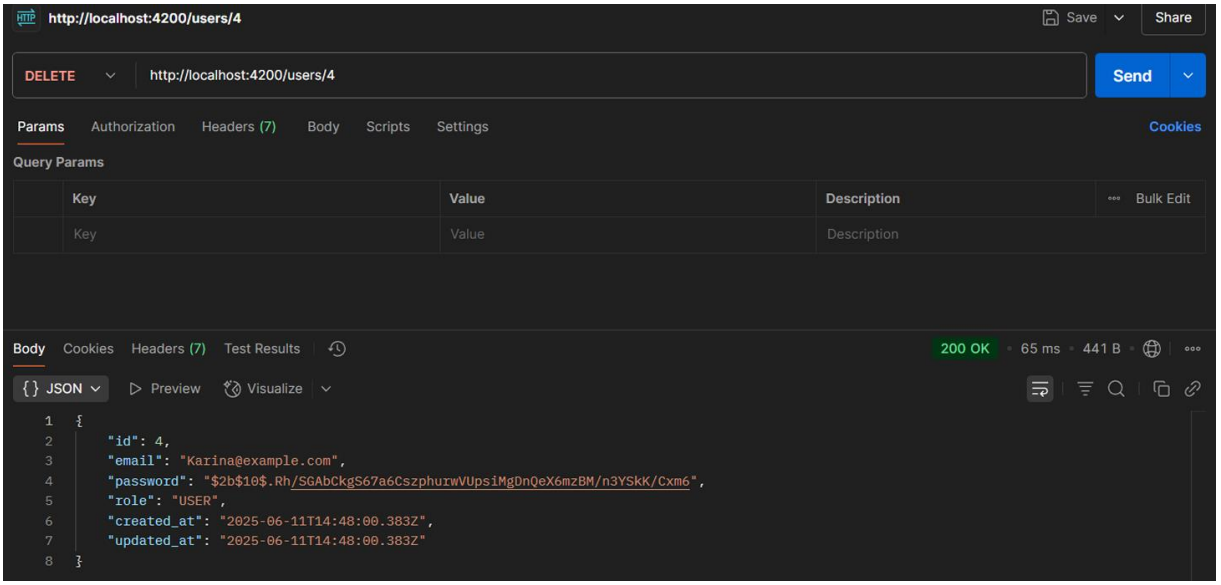


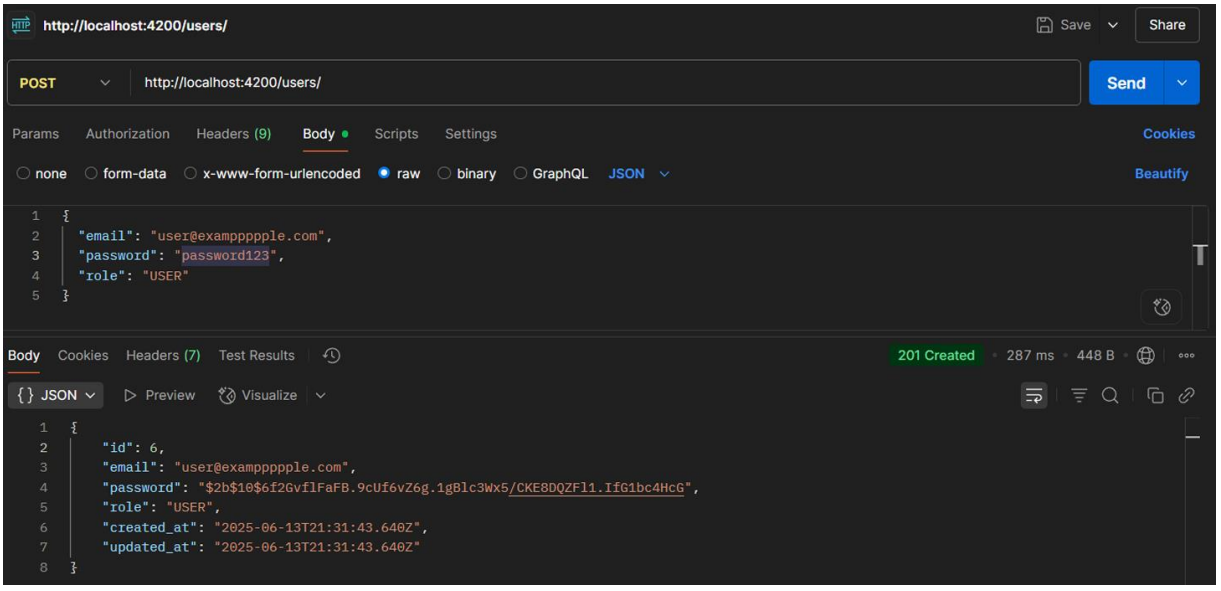
Industry сервис

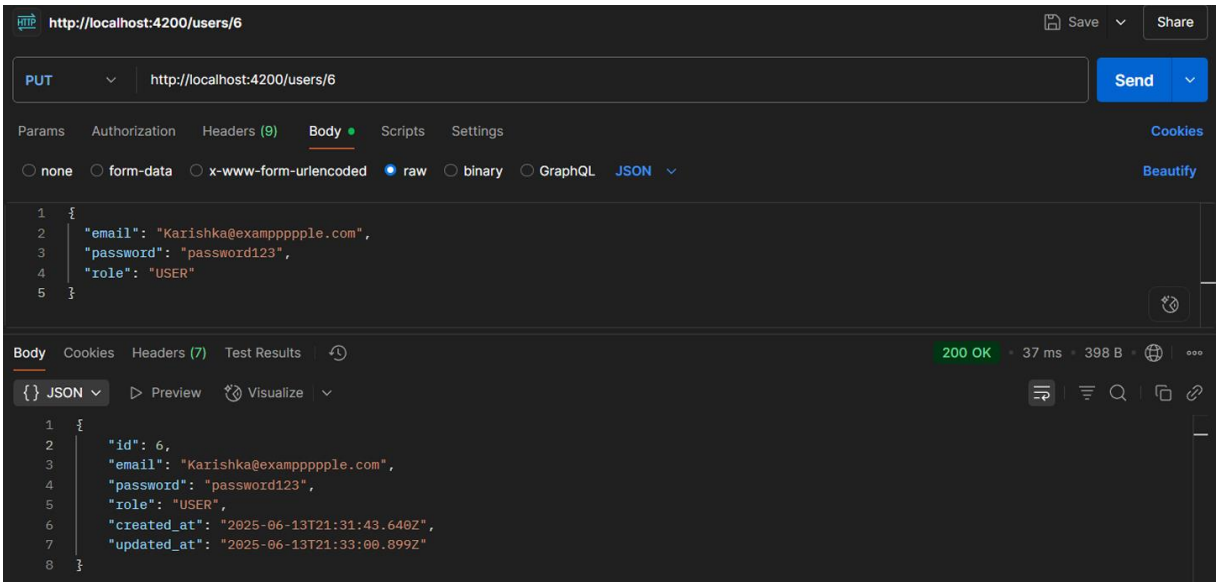


Документация API средствами Postman









Вывод

В данной домашней работе я реализовала автодокументирование средствами swagger и документацию API средствами Postman.