

Министерство науки и высшего образования Российской Федерации
Муромский институт (филиал)
федерального государственного бюджетного образовательного учреждения высшего образования
«Владимирский государственный университет
Имени Александра Григорьевича и Николая Григорьевича Столетовых»
(МИВлГУ)

Факультет _____ ИТР
Кафедра _____ ПИН

ЛАБОРАТОРНАЯ РАБОТА №4

по _____ Разработка корпоративных приложений
Тема Построение REST API

Руководитель

Кульков Я.Ю.

(фамилия, инициалы)

(подпись)

(дата)

Студент ПИН-122

(группа)

Загребин Д. А.

(фамилия, инициалы)

(подпись)

(дата)

Муром 2025

Лабораторная работа №4

Тема: Построение REST API

Цель: Создание Spring-приложение с несколькими таблицами с использованием REST API

Задание:

1. Создать набор сущностей в зависимости от варианта
2. Реализовать REST API с основными CRUD операциями.
3. Протестировать разработанные методы.
4. Реализовать методы с расширенной бизнес-логикой и сложными операциями.
5. Протестировать работу методов (curl, Postmant, HTTPie)

3 «Студент»	<i>фамилия; имя; отчество; пол; национальность; рост; вес; дата рождения (год, месяц число); номер телефона; домашний адрес (почтовый индекс, страна, область, район, город, улица, дом, квартира); ВУЗ; курс; группа; средний бал; специальность.</i>
----------------	--

					МИВУ 09.03.04-03.004												
Изм	Лист	№ докум.	Подп.	Дата													
Разраб.		Загребин Д. А.			Построение REST API												
Провер.		Кульков Я.Ю.										Лит.		Лист		Листов	
														2		11	
												МИВУ ПИН - 122					
Н.контр.																	
Утв.																	

Ход работы:

Листинг 1 - StudentRestController.java:

```
package ru.zagrebin.laball;

import java.util.NoSuchElementException;
import java.util.Optional;
import java.util.stream.Collectors;

import jakarta.persistence.EntityNotFoundException;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.HttpStatus;
import org.springframework.http.ResponseEntity;
import org.springframework.ui.Model;
import org.springframework.validation.BindingResult;
import org.springframework.web.bind.annotation.*;

import jakarta.validation.Valid;
import ru.zagrebin.laball.dto.*;
import ru.zagrebin.laball.entity.Address;
import ru.zagrebin.laball.entity.Student;
import ru.zagrebin.laball.entity.StudentGroup;
import ru.zagrebin.laball.form.StudentForm;
import ru.zagrebin.laball.service.ServiceAddress;
import ru.zagrebin.laball.service.ServiceCourse;
import ru.zagrebin.laball.service.ServiceStudent;
import ru.zagrebin.laball.service.ServiceStudentGroup;
import ru.zagrebin.laball.service.ServiceUniversity;

@RequestMapping("/api/student")
@RestController
public class StudentRestController {

    @Autowired
    private final ServiceStudent studentService;

    public StudentRestController(ServiceStudent studentService) {
        this.studentService = studentService;
    }

    @GetMapping("/statistics")
    public ResponseEntity<StudentStatistics> getStudentStatistics() {
        StudentStatistics statistics = studentService.getStudentStatistics();
        return ResponseEntity.ok(statistics);
    }
}
```

					МИВУ 09.03.04-03.004	Лист
						3
Изм.	Лист	№ докум.	Подп.	Дата		

```

    }

    @GetMapping("/{id}")
    public ResponseEntity<StudentDTO> getStudentById(@PathVariable Long id) {
        Student student = studentService.getStudentById(id).orElseThrow(() -
>new EntityNotFoundException("Student not found with id " + id));
        return new ResponseEntity<>(convertToDTO(student), HttpStatus.OK);
    }

    @PostMapping("/create")
    @ResponseStatus(HttpStatus.CREATED)
    public ResponseEntity<StudentDTO> createStudent(@Valid @RequestBody
StudentCreatedDTO studentCreatedDTO) {
        Student student =
studentService.createStudentFromDTO(studentCreatedDTO);
        return
ResponseEntity.status(HttpStatus.CREATED).body(convertToDTO(student));
    }

    @PutMapping("/{id}")
    public ResponseEntity<StudentDTO> updateStudent(@PathVariable Long id,
                                                    @Valid @RequestBody
StudentUpdatedDTO studentUpdatedDTO) {
        Student student = studentService.updateStudentFromDTO(id,
studentUpdatedDTO);
        return ResponseEntity.ok(convertToDTO(student));
    }

    @DeleteMapping("/{id}")
    @ResponseStatus(HttpStatus.NO_CONTENT)
    public void deleteStudent(@PathVariable Long id) {
        studentService.deleteStudentById(id);
    }

    private StudentDTO convertToDTO(Student student) {
        StudentDTO dto = new StudentDTO();
        dto.setId(student.getId());
        dto.setLastName(student.getLastName());
        dto.setFirstName(student.getFirstName());
        dto.setPatronymic(student.getPatronymic());
        dto.setGender(student.getGender());
        dto.setNationality(student.getNationality());
        dto.setHeight(student.getHeight());
        dto.setWeight(student.getWeight());
        dto.setBirthDate(student.getBirthDate());
    }

```

					МИВУ 09.03.04-03.004	Лист
						4
Изм.	Лист	№ докум.	Подп.	Дата		

```

        dto.setPhoneNumber(student.getPhoneNumber());
        dto.setGpa(student.getGpa());
        dto.setSpeciality(student.getSpeciality());

        if (student.getAddress() != null) {
            dto.setAddressId(student.getAddress().getId());
            dto.setCity(student.getAddress().getCity());
            dto.setStreet(student.getAddress().getStreet());
            dto.setHouse(student.getAddress().getHouse());
            dto.setApartment(student.getAddress().getApartment());
        }

        if (student.getStudentGroup() != null) {
            dto.setStudentGroupId(student.getStudentGroup().getId());
            dto.setGroupName(student.getStudentGroup().getName());
            if (student.getStudentGroup().getUniversity() != null) {
                dto.setUniversityName(student.getStudentGroup().getUniversity().getName());
            }
        }

        if (student.getCourses() != null) {
            dto.setCourseIds(student.getCourses().stream()
                .map(course -> course.getId())
                .collect(Collectors.toList()));
            dto.setCourseNames(student.getCourses().stream()
                .map(course -> course.getName())
                .collect(Collectors.toList()));
        }

        return dto;
    }
}

```

Листинг 2 - StudentCreatedDTO.java:

```

package ru.zagrebin.lab11.dto;

import lombok.Data;
import jakarta.validation.constraints.*;
import java.time.LocalDate;
import java.util.List;

@Data
public class StudentCreatedDTO {

```

Изм.	Лист	№ докум.	Подп.	Дата

```

@NotBlank(message = "Фамилия обязательна")
private String lastName;

@NotBlank(message = "Имя обязательно")
private String firstName;

private String patronymic;

@NotBlank(message = "Пол обязателен")
private String gender;

private String nationality;

@NotNull(message = "Рост обязателен")
@Min(value = 100, message = "Рост должен быть не менее 100 см")
private Double height;

@NotNull(message = "Вес обязателен")
@Min(value = 30, message = "Вес должен быть не менее 30 кг")
private Double weight;

@NotNull(message = "Дата рождения обязательна")
@Past(message = "Дата рождения должна быть в прошлом")
private LocalDate birthDate;

@NotBlank(message = "Номер телефона обязателен")
private String phoneNumber;

@NotNull(message = "GPA обязателен")
@DecimalMin(value = "0.0", message = "GPA не может быть меньше 0")
@DecimalMax(value = "5.0", message = "GPA не может быть больше 5")
private Double gpa;

private String speciality;

private String city;
private String street;
private String house;
private String apartment;

@NotNull(message = "ID группы обязателен")
private Long studentGroupId;

```

					МИВУ 09.03.04-03.004	Лист
						6
Изм.	Лист	№ докум.	Подп.	Дата		

```

        private List<Long> courseIds;
    }

```

Листинг 3 - StudentDTO.java:

```

package ru.zagrebin.laball.dto;

import lombok.Data;
import java.time.LocalDate;
import java.util.List;

@Data
public class StudentDTO {
    private Long id;
    private String lastName;
    private String firstName;
    private String patronymic;
    private String gender;
    private String nationality;
    private Double height;
    private Double weight;
    private LocalDate birthDate;
    private String phoneNumber;
    private Double gpa;
    private String speciality;

    private Long addressId;
    private String city;
    private String street;
    private String house;
    private String apartment;

    private Long studentGroupId;
    private String groupName;
    private String universityName;

    private List<Long> courseIds;
    private List<String> courseNames;
}

```

Листинг 4 - StudentStatistics.java:

```

package ru.zagrebin.laball.dto;

import lombok.Data;

```

					МИВУ 09.03.04-03.004	Лист
						7
Изм.	Лист	№ докум.	Подп.	Дата		

```

@Data
public class StudentStatistics {
    private Long totalStudents;
    private Double averageGPA;
    private Double maxGPA;
    private Double minGPA;
    private Long studentsWithHighGPA;
    private String mostPopularSpeciality;
    private String cityWithMostStudents;

    private Long maleCount;
    private Long femaleCount;

    private String largestGroup;
    private Long studentsInLargestGroup;
}

```

Листинг 5 - StudentUpdateDTO.java:

```

package ru.zagrebin.laball.dto;

import lombok.Data;
import jakarta.validation.constraints.*;
import java.time.LocalDate;
import java.util.List;

@Data
public class StudentUpdateDTO {
    private String lastName;

    private String firstName;

    private String patronymic;

    private String gender;

    private String nationality;

    @Min(value = 100, message = "Рост должен быть не менее 100 см")
    private Double height;

    @Min(value = 30, message = "Вес должен быть не менее 30 кг")
    private Double weight;

    @Past(message = "Дата рождения должна быть в прошлом")

```

					МИВУ 09.03.04-03.004	Лист
						8
Изм.	Лист	№ докум.	Подп.	Дата		


```

private LocalDate birthDate;

private String phoneNumber;

@DecimalMin(value = "0.0", message = "GPA не может быть меньше 0")
@DecimalMax(value = "5.0", message = "GPA не может быть больше 5")
private Double gpa;

private String speciality;

private String city;
private String street;
private String house;
private String apartment;

private Long studentGroupId;

private List<Long> courseIds;
}

```

Демонстрация работы программы:

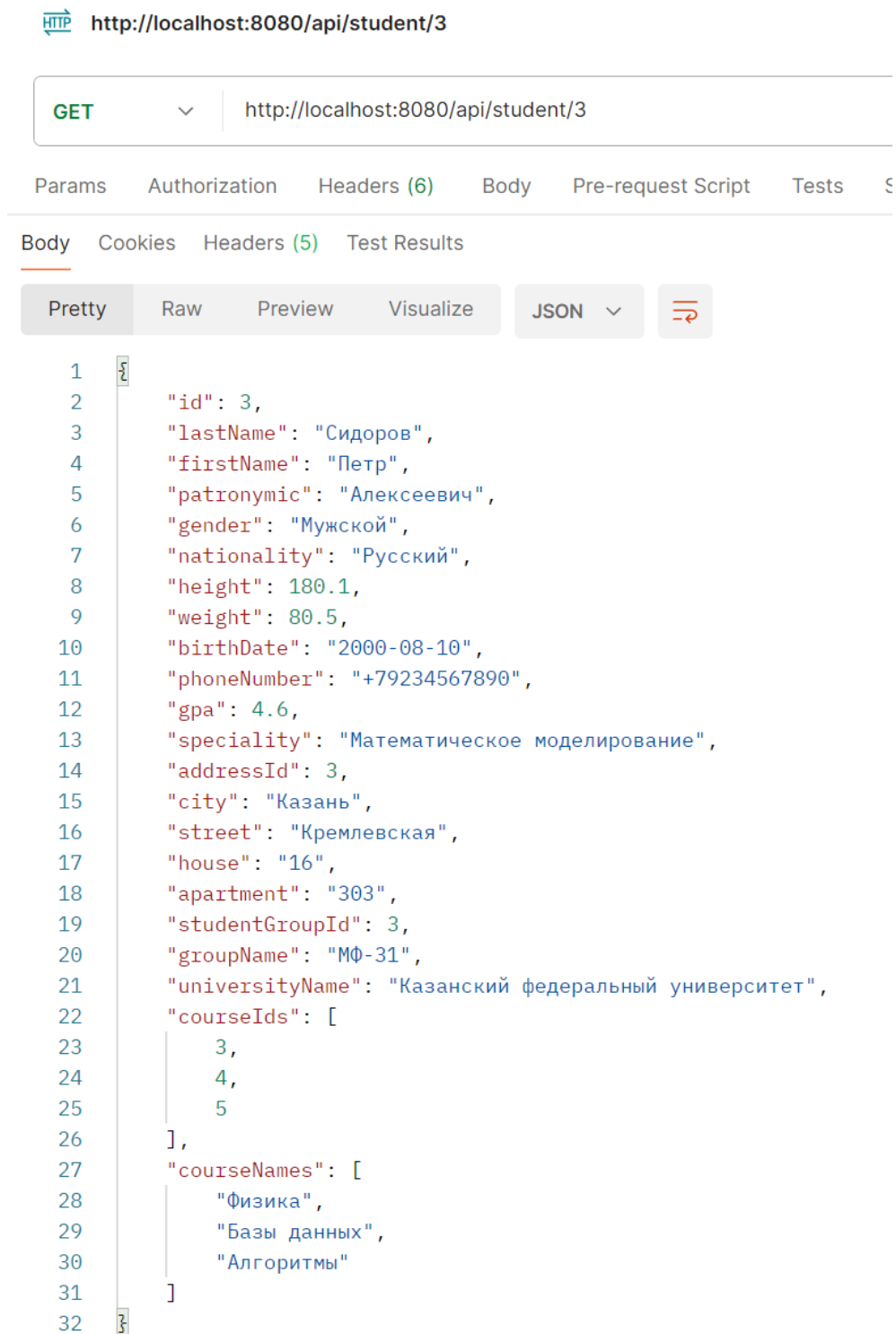


Рисунок 1 – Получение студента

GET
http://localhost:8080/api/student/statistics

Params

Authorization

Headers (6)

Body

Pre-request Script

Tests

Settings

Query Params

	Key
	Key

Body

Cookies

Headers (5)

Test Results

Pretty

Raw

Preview

Visualize

JSON

```

1  {
2      "totalStudents": 3,
3      "averageGPA": 4.633333333333333,
4      "maxGPA": 4.8,
5      "minGPA": 4.5,
6      "studentsWithHighGPA": 3,
7      "mostPopularSpeciality": "Математическое моделирование",
8      "cityWithMostStudents": "Казань",
9      "maleCount": 2,
10     "femaleCount": 1,
11     "largestGroup": "ИТ-21",
12     "studentsInLargestGroup": 1
13 }

```

Рисунок 2 – Получение статистики

Вывод: Входе лабораторной работы было изучено создание Spring-приложения с несколькими таблицами с использованием REST API