**SKG\_STUDENT\_PROJECT**

**Student.java:**

package com.examly.springapp.model;

import javax.persistence.Entity;

import javax.persistence.Id;

@Entity

public class Student {

@Id

int id;

String name;

int age;

String address,department;

public Student(){}

public Student(int id, String name, int age, String address, String department) {

this.id = id;

this.name = name;

this.age = age;

this.address = address;

this.department = department;

}

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public int getAge() {

return age;

}

public void setAge(int age) {

this.age = age;

}

public String getAddress() {

return address;

}

public void setAddress(String address) {

this.address = address;

}

public String getDepartment() {

return department;

}

public void setDepartment(String department) {

this.department = department;

}

}

**StudentRepository.java:**

package com.examly.springapp.repository;

import org.springframework.data.jpa.repository.JpaRepository;

import com.examly.springapp.model.Student;

public interface StudentRepository extends JpaRepository<Student,Integer>{}

**StudentService.java**

package com.examly.springapp.service;

import java.util.Comparator;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Service;

import com.examly.springapp.model.Student;

import com.examly.springapp.repository.StudentRepository;

@Service

public class StudentService {

@Autowired

StudentRepository sr;

public Student create(Student s)

{

return sr.save(s);

}

public List<Student> getall()

{

return sr.findAll();

}

public Student getById(int id)

{

return sr.findById(id).orElse(null);

}

public boolean update(int id,Student s)

{

if(this.getById(id)==null)

{

return false;

}

try{

sr.save(s);

}

catch(Exception e)

{

return false;

}

return true;

}

public List<Student> sortbyname()

{

List<Student> students = sr.findAll();

students.sort(Comparator.comparing(Student::getName));

return students;

}

public List<Student> getStudentsByAgeRange(int minAge, int maxAge) {

List<Student> students = sr.findAll();

students.removeIf(s -> s.getAge() < minAge || s.getAge() > maxAge);

students.sort(Comparator.comparing(Student::getAge));

return students;

}

public boolean delete(int id)

{

if(this.getById(id) == null)

{

return false;

}

sr.deleteById(id);

return true;

}

}

**Studentcontroller.java**

package com.examly.springapp.controller;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.http.HttpStatus;

import org.springframework.http.ResponseEntity;

import org.springframework.web.bind.annotation.DeleteMapping;

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.PathVariable;

import org.springframework.web.bind.annotation.PostMapping;

import org.springframework.web.bind.annotation.PutMapping;

import org.springframework.web.bind.annotation.RequestBody;

import org.springframework.web.bind.annotation.RestController;

import com.examly.springapp.model.Student;

import com.examly.springapp.service.StudentService;

@RestController

public class StudentController {

@Autowired

StudentService ss;

@PostMapping("/student")

public ResponseEntity<Student> post(@RequestBody Student s)

{

Student obj=ss.create(s);

return new ResponseEntity<>(obj,HttpStatus.CREATED);

}

@GetMapping("/student")

public ResponseEntity<List<Student>> get()

{

return new ResponseEntity<>(ss.getall(),HttpStatus.OK);

}

@GetMapping("/student/{id}")

public ResponseEntity<Student> getbyid(@PathVariable("id") int id)

{

return new ResponseEntity<>(ss.getById(id),HttpStatus.OK);

}

@PutMapping("/student/{id}")

public ResponseEntity<Student> putMethod(@PathVariable("id") int id,@RequestBody Student s)

{

if(ss.update(id,s) == true)

{

return new ResponseEntity<>(s,HttpStatus.OK);

}

return new ResponseEntity<>(null,HttpStatus.NOT\_FOUND);

}

@DeleteMapping("student/{id}")

public ResponseEntity<?> deleteBookById(@PathVariable int id)

{

boolean deleted = ss.delete(id);

if (deleted) {

return ResponseEntity.ok("Student deleted successfully");

} else {

return ResponseEntity.status(HttpStatus.NOT\_FOUND).body("Student not found with ID: " + id);

}

}

@GetMapping("/student/sortedByName")

public ResponseEntity<List<Student>> g()

{

List<Student> students = ss.sortbyname();

return ResponseEntity.ok(students);

}

@GetMapping("/student/ageRange/{minAge}/{maxAge}")

public ResponseEntity<List<Student>> getStudentsByAgeRange(@PathVariable int minAge, @PathVariable int maxAge) {

List<Student> students = ss.getStudentsByAgeRange(minAge, maxAge);

return ResponseEntity.ok(students);

}

}

**OUTPUT:**

