**SKG\_LAB\_BLOOD\_DONOR**

**Model**

**Donor.java**

package com.examly.springapp.model;

import javax.persistence.Entity;

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence.Id;

@Entity

public class Donor {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private int donorId;

private String name;

private int age;

private String address;

private String bloodGroup;

public Donor() {

}

public Donor(int donorId, String name, int age, String address, String bloodGroup) {

this.donorId = donorId;

this.name = name;

this.age = age;

this.address = address;

this.bloodGroup = bloodGroup;

}

public int getDonorId() {

return donorId;

}

public void setDonorId(int donorId) {

this.donorId = donorId;

}

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 getBloodGroup() {

return bloodGroup;

}

public void setBloodGroup(String bloodGroup) {

this.bloodGroup = bloodGroup;

}

}

**Repository**

**DonorRepo.java**

package com.examly.springapp.repository;

import java.util.List;

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

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

import com.examly.springapp.model.Donor;

public interface DonorRepo extends JpaRepository<Donor,Integer> {

Donor findByDonorId(int donorId);

@Query("SELECT d FROM Donor d ORDER BY d.age")

List<Donor> findByDonorId();

@Query("SELECT d FROM Donor d ORDER BY d.bloodGroup")

List<Donor> findByBloodGroup();

@Query("SELECT d FROM Donor d WHERE d.age BETWEEN :minAge AND :maxAge")

List<Donor> findByAgeRange(int minAge, int maxAge);

}

**Service**

**DonorService.java**

package com.examly.springapp.service;

import java.util.List;

import java.util.Optional;

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

import org.springframework.stereotype.Service;

import com.examly.springapp.model.Donor;

import com.examly.springapp.repository.DonorRepo;

@Service

public class DonorService {

@Autowired

private DonorRepo repo;

public Donor addDonor(Donor donor) {

return repo.save(donor);

}

public Donor getsDonor(int donorId) {

return repo.findByDonorId(donorId);

}

public List<Donor> getDonor() {

return repo.findAll();

}

public Donor updateDonor(int donorId,Donor donor) {

Optional <Donor> donorExist = repo.findById(donorId);

if(donorExist.isPresent()){

Donor existingDonor = donorExist.get();

existingDonor.setName(donor.getName());

existingDonor.setAge(donor.getAge());

existingDonor.setAddress(donor.getAddress());

return repo.save(donor);

}

return new Donor();

}

public boolean deleteDonor(int donorId) {

Optional<Donor> donorExist = repo.findById(donorId);

if (donorExist.isPresent()) {

repo.deleteById(donorId);

System.out.println("Donor deleted Successfully");

return true;

} else {

System.out.println("Donor not found");

return false;

}

}

public List<Donor> sortAge() {

return repo.findByDonorId();

}

public List<Donor> sortByBloodGroup() {

return repo.findByBloodGroup();

}

public List<Donor> sortByAgeRange(int minAge, int maxAge) {

return repo.findByAgeRange(minAge,maxAge);

}

}

**Controller**

**DonorController.java**

package com.examly.springapp.controller;

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.Donor;

import com.examly.springapp.service.DonorService;

@RestController

public class DonorController {

@Autowired

private DonorService service;

@PostMapping("/donor")

public ResponseEntity<?>addDonor(@RequestBody Donor donor)

{

try {

return new ResponseEntity<>(service.addDonor(donor),HttpStatus.CREATED);

} catch (Exception e) {

return new ResponseEntity<>(HttpStatus.INTERNAL\_SERVER\_ERROR);

}

}

@GetMapping("/donor/{donorId}")

public ResponseEntity<?>getsDonor(@PathVariable int donorId){

try {

return new ResponseEntity<>(service.getsDonor(donorId),HttpStatus.OK);

} catch (Exception e) {

return new ResponseEntity<>(HttpStatus.INTERNAL\_SERVER\_ERROR);

}

}

@GetMapping("/donor")

public ResponseEntity<?>getDonor(){

try {

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

} catch (Exception e) {

return new ResponseEntity<>(HttpStatus.INTERNAL\_SERVER\_ERROR);

}

}

@PutMapping("/donor/{donorId}")

public ResponseEntity<?>updateDonor(@PathVariable int donorId,@RequestBody Donor donor){

try {

return new ResponseEntity<>(service.updateDonor(donorId,donor),HttpStatus.OK);

} catch (Exception e) {

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

}

}

@DeleteMapping("/donor/{donorId}")

public ResponseEntity<?> deleteDonor(@PathVariable int donorId) {

boolean deletionStatus = service.deleteDonor(donorId);

if (deletionStatus) {

return new ResponseEntity<>("Donor deleted Successfully",HttpStatus.OK);

} else {

return new ResponseEntity<>("Donor not found",HttpStatus.NOT\_FOUND);

}

}

@GetMapping("/donor/age/{age}")

public ResponseEntity<?>sortByAge(){

try {

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

} catch (Exception e) {

return new ResponseEntity<>(HttpStatus.INTERNAL\_SERVER\_ERROR);

}

}

@GetMapping("/donor/bloodGroup/{bloodGroup}")

public ResponseEntity<?>sortByBloodGroup(){

try {

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

} catch (Exception e) {

return new ResponseEntity<>(HttpStatus.INTERNAL\_SERVER\_ERROR);

}

}

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

public ResponseEntity<?>sortByAgeRange(@PathVariable int minAge,@PathVariable int maxAge){

try {

return new ResponseEntity<>(service.sortByAgeRange(minAge,maxAge),HttpStatus.OK);

} catch (Exception e) {

return new ResponseEntity<>(HttpStatus.INTERNAL\_SERVER\_ERROR);

}

}

}

**OUTPUT**

