✅ Case Study Title: Hospital Management System using Spring Boot and Spring Data JPA

Appointment.java

```
package com.example.hospital.entity;
import java.time.*;
import com.fasterxml.jackson.annotation.JsonBackReference;
import jakarta.persistence.*;
@Entity
public class Appointment {
	@Id
	@GeneratedValue(strategy=GenerationType.IDENTITY)
	private Long id;

  private LocalDate date;
  private LocalTime time;
  private String notes;
  @ManyToOne
  @JoinColumn(name = "patient_id")
  @JsonBackReference(value = "patient-appointments")
  private Patient patient;
  @ManyToOne
  @JoinColumn(name = "doctor_id")
  @JsonBackReference(value = "doctor-appointments")
  private Doctor doctor;
	public Appointment() {}
	public Appointment(Long id, LocalDate date, LocalTime time, String notes, Patient patient,
Doctor doctor) {
		this.id = id;
		this.date = date;
		this.time = time;
		this.notes = notes;
		this.patient = patient;
		this.doctor = doctor;
	}
	public Long getId() {
		return id;
	}
	public void setId(Long id) {
		this.id = id;
	}
	public LocalDate getDate() {
		return date;
	}
	public void setDate(LocalDate date) {
		this.date = date;
```

```
	}
	public LocalTime getTime() {
		return time;
	}
	public void setTime(LocalTime time) {
		this.time = time;
	}
	public String getNotes() {
		return notes;
	}
	public void setNotes(String notes) {
		this.notes = notes;
	}
	public Patient getPatient() {
		return patient;
	}
	public void setPatient(Patient patient) {
		this.patient = patient;
	}
	public Doctor getDoctor() {
		return doctor;
	}
	public void setDoctor(Doctor doctor) {
		this.doctor = doctor;
	}
}
```

Doctor.java

```
package com.example.hospital.entity;
import java.util.*;
import com.fasterxml.jackson.annotation.JsonManagedReference;
import jakarta.persistence.*;
@Entity
public class Doctor {
	@Id
	@GeneratedValue(strategy=GenerationType.IDENTITY)
  private Long id;
  private String name;
  private String specialization;
  private String email;
  private String phone;
  @OneToMany(mappedBy = "doctor", cascade = CascadeType.ALL)
  @JsonManagedReference(value = "doctor-appointments")
  private List<Appointment> appointments = new ArrayList<>();
	public Doctor() {}
	public Doctor(Long id, String name, String specialization, String email, String phone,
List<Appointment> appointments) {
```

```
        this.id = id;
        this.name = name;
        this.specialization = specialization;
        this.email = email;
        this.phone = phone;
        this.appointments = appointments;
    }
    public Long getId() {
        return id;
    }
    public void setId(Long id) {
        this.id = id;
    }
    public String getName() {
        return name;
    }
    public void setName(String name) {
        this.name = name;
    }
    public String getSpecialization() {
        return specialization;
    }
    public void setSpecialization(String specialization) {
        this.specialization = specialization;
    }
    public String getEmail() {
        return email;
    }
    public void setEmail(String email) {
        this.email = email;
    }
    public String getPhone() {
        return phone;
    }
    public void setPhone(String phone) {
        this.phone = phone;
    }
    public List<Appointment> getAppointments() {
        return appointments;
    }
    public void setAppointments(List<Appointment> appointments) {
        this.appointments = appointments;
    }
}
```

MedicalRecord.java

```
package com.example.hospital.entity;
```

```
import java.time.LocalDate;
import com.fasterxml.jackson.annotation.JsonBackReference;
import jakarta.persistence.*;
@Entity
public class MedicalRecord {
	@Id
	@GeneratedValue(strategy=GenerationType.IDENTITY)
	private Long id;

  private String diagnosis;
  private String treatment;
  private LocalDate date;
  @ManyToOne
  @JoinColumn(name = "patient_id")
  @JsonBackReference(value = "patient-medicalRecords")
  private Patient patient;
	public MedicalRecord() {}
	public MedicalRecord(Long id, String diagnosis, String treatment, LocalDate date, Patient
patient) {
		this.id = id;
		this.diagnosis = diagnosis;
		this.treatment = treatment;
		this.date = date;
		this.patient = patient;
	}
	public Long getId() {
		return id;
	}
	public void setId(Long id) {
		this.id = id;
	}
	public String getDiagnosis() {
		return diagnosis;
	}
	public void setDiagnosis(String diagnosis) {
		this.diagnosis = diagnosis;
	}
	public String getTreatment() {
		return treatment;
	}
	public void setTreatment(String treatment) {
		this.treatment = treatment;
	}
	public LocalDate getDate() {
		return date;
	}
	public void setDate(LocalDate date) {
		this.date = date;
	}
```

```
	public Patient getPatient() {
		return patient;
	}
	public void setPatient(Patient patient) {
		this.patient = patient;
	}


}
```

Patient.java

```
package com.example.hospital.entity;
import java.util.*;
import com.fasterxml.jackson.annotation.JsonManagedReference;
import jakarta.persistence.*;
@Entity
public class Patient {
	@Id
	@GeneratedValue(strategy=GenerationType.IDENTITY)
  private Long id;
  private String name;
  private int age;
  private String gender;
  private String address;
  @OneToMany(mappedBy = "patient", cascade = CascadeType.ALL)
  @JsonManagedReference(value = "patient-appointments")
  private List<Appointment> appointments = new ArrayList<>();
  @OneToMany(mappedBy = "patient", cascade = CascadeType.ALL)
  @JsonManagedReference(value = "patient-medicalRecords")
  private List<MedicalRecord> medicalRecords = new ArrayList<>();
	public Patient() {}
	public Patient(Long id, String name, int age, String gender, String address,
List<Appointment> appointments, List<MedicalRecord> medicalRecords) {
		this.id = id;
		this.name = name;
		this.age = age;
		this.gender = gender;
		this.address = address;
		this.appointments = appointments;
		this.medicalRecords = medicalRecords;
	}
	public Long getId() {
		return id;
	}
	public void setId(Long 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 getGender() {
        return gender;
    }
    public void setGender(String gender) {
        this.gender = gender;
    }
    public String getAddress() {
        return address;
    }
    public void setAddress(String address) {
        this.address = address;
    }
    public List<Appointment> getAppointments() {
        return appointments;
    }
    public void setAppointments(List<Appointment> appointments) {
        this.appointments = appointments;
    }
    public List<MedicalRecord> getMedicalRecords() {
        return medicalRecords;
    }
    public void setMedicalRecords(List<MedicalRecord> medicalRecords) {
        this.medicalRecords = medicalRecords;
    }
}
```

HospitalController.java

```
package com.example.hospital.controller;
import java.util.*;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.*;
import com.example.hospital.entity.*;
import com.example.hospital.repository.*;
@RestController
@RequestMapping("/api")
```

```
public class HospitalController {
  @Autowired
  private PatientRepository patientRepo;
  @Autowired
  private DoctorRepository doctorRepo;
  @Autowired
  private AppointmentRepository appointmentRepo;
  @Autowired
  private MedicalRecordRepository medicalRecordRepo;
//   PATIENT
  @PostMapping("/patients")
  public Patient createPatient(@RequestBody Patient patient) {
     return patientRepo.save(patient);
  }
  @GetMapping("/patients")
  public List<Patient> getAllPatients() {
     return patientRepo.findAll();
  }
  @PutMapping("/patients/{id}")
  public Patient updatePatient(@PathVariable Long id, @RequestBody Patient updated) {
     Patient p = patientRepo.findById(id).orElseThrow();
        p.setName(updated.getName());
        p.setAge(updated.getAge());
        p.setGender(updated.getGender());
        p.setAddress(updated.getAddress());
        return patientRepo.save(p);
  }
  @DeleteMapping("/patients/{id}")
  public String deletePatient(@PathVariable Long id) {
     patientRepo.deleteById(id);
     return "Deleted Successfully";
  }
  // DOCTOR
  @PostMapping("/doctors")
  public Doctor createDoctor(@RequestBody Doctor doctor) {
     return doctorRepo.save(doctor);
  }
  @GetMapping("/doctors")
  public List<Doctor> getAllDoctors() {
     return doctorRepo.findAll();
  }
  @PutMapping("/doctors/{id}")
  public Doctor updateDoctor(@PathVariable Long id, @RequestBody Doctor updated) {
     Doctor d = doctorRepo.findById(id).orElseThrow();
        d.setName(updated.getName());
        d.setSpecialization(updated.getSpecialization());
        d.setEmail(updated.getEmail());
        d.setPhone(updated.getPhone());
        return doctorRepo.save(d);
```

```
    }
    @DeleteMapping("/doctors/{id}")
    public String deleteDoctor(@PathVariable Long id) {
        doctorRepo.deleteById(id);
        return "Deleted Successfully";
    }
    // APPOINTMENT
    @PostMapping("/appointments")
    public Appointment createAppointment(@RequestBody Appointment appointment) {

appointment.setPatient(patientRepo.findById(appointment.getPatient().getId()).orElseThrow());
        appointment.setDoctor(doctorRepo.findById(appointment.getDoctor().getId()).orElseThrow());
        return appointmentRepo.save(appointment);
    }
    @GetMapping("/appointments")
    public List<Appointment> getAllAppointments() {
        return appointmentRepo.findAll();
    }
    @PutMapping("/appointments/{id}")
    public Appointment updateAppointment(@PathVariable Long id, @RequestBody Appointment
updated) {
        Appointment a = appointmentRepo.findById(id).orElseThrow();
        a.setDate(updated.getDate());
        a.setTime(updated.getTime());
        a.setNotes(updated.getNotes());
        a.setPatient(patientRepo.findById(updated.getPatient().getId()).orElseThrow());
        a.setDoctor(doctorRepo.findById(updated.getDoctor().getId()).orElseThrow());
        return appointmentRepo.save(a);
    }
    @DeleteMapping("/appointments/{id}")
    public String deleteAppointment(@PathVariable Long id) {
        appointmentRepo.deleteById(id);
        return "Deleted Successfully";
    }
    // MEDICAL RECORD
    @PostMapping("/medical-records")
    public MedicalRecord createMedicalRecord(@RequestBody MedicalRecord record) {
        record.setPatient(patientRepo.findById(record.getPatient().getId()).orElseThrow());
        return medicalRecordRepo.save(record);
    }
    @GetMapping("/medical-records")
    public List<MedicalRecord> getAllMedicalRecords() {
        return medicalRecordRepo.findAll();
    }
    @PutMapping("/medical-records/{id}")
    public MedicalRecord updateMedicalRecord(@PathVariable Long id, @RequestBody
MedicalRecord updated) {
        MedicalRecord m = medicalRecordRepo.findById(id).orElseThrow();
        m.setDiagnosis(updated.getDiagnosis());
```

```
        m.setTreatment(updated.getTreatment());
        m.setDate(updated.getDate());
        m.setPatient(patientRepo.findById(updated.getPatient().getId()).orElseThrow());
        return medicalRecordRepo.save(m);
    }
    @DeleteMapping("/medical-records/{id}")
    public String deleteMedicalRecord(@PathVariable Long id) {
        medicalRecordRepo.deleteById(id);
        return "Deleted Successfully";
    }
}
```