Case Study: Library Management System

Create MySQL Database

```
CREATE DATABASE library_db;
```

Configure application.properties

```
spring.datasource.url=jdbc:mysql://localhost:3306/library_db
spring.datasource.username=root
spring.datasource.password=yourpassword
spring.jpa.hibernate.ddl-auto=update
spring.jpa.show-sql=true
spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQL8Dialect
```

com.example.library.entity.Reader.java

```
package com.example.library.entity;

import jakarta.persistence.*;
import lombok.*;

import java.util.List;

@Entity
@Data
@NoArgsConstructor
@AllArgsConstructor
public class Reader {
   @Id
   @GeneratedValue(strategy = GenerationType.IDENTITY)
   private Long id;
```

```
    private String name;
    private String email;

    @OneToMany(mappedBy = "reader")
    private List<Book> books;
}
```

Book.java

```
package com.example.library.entity;

import jakarta.persistence.*;
import lombok.*;

import java.time.LocalDate;

@Entity
@Data
@NoArgsConstructor
@AllArgsConstructor
public class Book {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;

    private String title;
    private LocalDate publishDate;

    @ManyToOne
    @JoinColumn(name = "reader_id")
    private Reader reader;

    @ManyToOne
    @JoinColumn(name = "category_id")
    private Category category;

    @ManyToOne
    @JoinColumn(name = "author_id")
    private Author author;
}
```

Category.java

```
package com.example.library.entity;

import jakarta.persistence.*;
import lombok.*;

import java.util.List;

@Entity
@Data
@NoArgsConstructor
@AllArgsConstructor
public class Category {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;

    private String name;

    @OneToMany(mappedBy = "category")
    private List<Book> books;
}
```

Author.java

```
package com.example.library.entity;

import jakarta.persistence.*;
import lombok.*;

import java.util.List;

@Entity
@Data
@NoArgsConstructor
@AllArgsConstructor
public class Author {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;

    private String name;

    @OneToMany(mappedBy = "author")
    private List<Book> books;
}
```

repository/ReaderRepository.java

```
package com.example.library.repository;

import com.example.library.entity.Reader;
import org.springframework.data.jpa.repository.JpaRepository;

public interface ReaderRepository extends JpaRepository<Reader, Long> {}
```

BookRepository.java

```
package com.example.library.repository;

import com.example.library.entity.Book;
import org.springframework.data.jpa.repository.JpaRepository;

public interface BookRepository extends JpaRepository<Book, Long> {}
```

CategoryRepository.java

```
package com.example.library.repository;

import com.example.library.entity.Category;
import org.springframework.data.jpa.repository.JpaRepository;

public interface CategoryRepository extends JpaRepository<Category, Long> {}
```

AuthorRepository.java

```
package com.example.library.repository;

import com.example.library.entity.Category;
import org.springframework.data.jpa.repository.JpaRepository;

public interface CategoryRepository extends JpaRepository<Category, Long> {}
```

controller/LibraryController.java

```
package com.example.library.controller;

import com.example.library.entity.*;
import com.example.library.repository.*;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.*;
import java.util.List;

@RestController
@RequestMapping("/api")
public class LibraryController {

  @Autowired private ReaderRepository readerRepo;
  @Autowired private BookRepository bookRepo;
  @Autowired private CategoryRepository categoryRepo;
  @Autowired private AuthorRepository authorRepo;

  //  Add Category
  @PostMapping("/categories")
  public Category addCategory(@RequestBody Category category) {
    return categoryRepo.save(category);
  }

  //  Add Author
  @PostMapping("/authors")
  public Author addAuthor(@RequestBody Author author) {
    return authorRepo.save(author);
  }

  //  Add Reader
  @PostMapping("/readers")
  public Reader addReader(@RequestBody Reader reader) {
    return readerRepo.save(reader);
  }

  //  Add Book
  @PostMapping("/books")
```

```
    public Book addBook(@RequestBody Book book) {
        return bookRepo.save(book);
    }

    // GET all
    @GetMapping("/books")
    public List<Book> getAllBooks() {
        return bookRepo.findAll();
    }

    @GetMapping("/readers")
    public List<Reader> getAllReaders() {
        return readerRepo.findAll();
    }

    @GetMapping("/categories")
    public List<Category> getAllCategories() {
        return categoryRepo.findAll();
    }

    @GetMapping("/authors")
    public List<Author> getAllAuthors() {
        return authorRepo.findAll();
    }
}
```

Main Application Class

```
package com.example.library;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
```

```
@SpringBootApplication
public class LibraryManagementApplication {
    public static void main(String[] args) {
        SpringApplication.run(LibraryManagementApplication.class, args);
    }
}
```

Test Using Postman

Add Category

http

Copy code

POST http://localhost:8080/api/categories

Body:

```
{
 "name": "Fiction"
}
```

Add Author

http

Copy code

POST http://localhost:8080/api/authors

```
{
 "name": "George Orwell"
}
```

Add Reader

http

Copy code

POST http://localhost:8080/api/readers

```
{
 "name": "Alice",
 "email": "alice@gmail.com"
}
```

Add Book

http

Copy code

```
POST http://localhost:8080/api/books
{
 "title": "1984",
 "publishDate": "1949-06-08",
 "reader": { "id": 1 },
 "category": { "id": 1 },
 "author": { "id": 1 }
}
```

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

Create Database

```
CREATE DATABASE hospitaldb;
```

application.properties:

```
spring.datasource.url=jdbc:mysql://localhost:3306/hospitaldb
spring.datasource.username=root
spring.datasource.password=yourpassword
spring.jpa.hibernate.ddl-auto=update
spring.jpa.show-sql=true
spring.jpa.properties.hibernate.format_sql=true
```

Patient.java

```
@Entity
```

```
@Data
@NoArgsConstructor
@AllArgsConstructor
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)
    private List<Appointment> appointments;

    @OneToMany(mappedBy = "patient", cascade = CascadeType.ALL)
    private List<MedicalRecord> medicalRecords;
}
```

Doctor.java

```
@Entity
@Data
@NoArgsConstructor
@AllArgsConstructor
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)
    private List<Appointment> appointments;
}
```

Appointment.java

```
@Entity
@Data
@NoArgsConstructor
@AllArgsConstructor
public class Appointment {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;

    private LocalDate date;
    private LocalTime time;
    private String notes;

    @ManyToOne
    private Patient patient;

    @ManyToOne
    private Doctor doctor;
}
```

MedicalRecord.java

```
@Entity
@Data
@NoArgsConstructor
@AllArgsConstructor
public class MedicalRecord {
   @Id
   @GeneratedValue(strategy = GenerationType.IDENTITY)
   private Long id;

   private String diagnosis;
   private String treatment;
   private LocalDate date;

   @ManyToOne
   private Patient patient;
}
```

Repository

```
public interface PatientRepository extends JpaRepository<Patient, Long> {}
public interface DoctorRepository extends JpaRepository<Doctor, Long> {}
public interface AppointmentRepository extends JpaRepository<Appointment, Long> {}
public interface MedicalRecordRepository extends JpaRepository<MedicalRecord, Long> {}
```

controller

```
@RestController
@RequestMapping("/api/patients")
@RequiredArgsConstructor
public class PatientController {
   private final PatientRepository patientRepo;
```

```
    @PostMapping
    public Patient create(@RequestBody Patient patient) {
        return patientRepo.save(patient);
    }

    @GetMapping
    public List<Patient> getAll() {
        return patientRepo.findAll();
    }
}
```

Application Class

```
@SpringBootApplication
public class HospitalManagementApplication {
    public static void main(String[] args) {
        SpringApplication.run(HospitalManagementApplication.class, args);
    }
}
```

Testing in Postman

POST /api/patients

```
{
  "name": "John Doe",
  "age": 35,
  "gender": "Male",
  "address": "123 Main Street"
}
```