Library Management system

1.Author.java

```
package com.example.library;


import java.util.List;


import jakarta.persistence.CascadeType;
import jakarta.persistence.Entity;
import jakarta.persistence.GeneratedValue;
import jakarta.persistence.GenerationType;
import jakarta.persistence.Id;
import jakarta.persistence.OneToMany;

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

    private String name;
    private String biography;

    @OneToMany(mappedBy = "author", cascade = CascadeType.ALL)
    private List<Book> books;

    // Getters and Setters
    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 getBiography() { return biography; }
    public void setBiography(String biography) { this.biography = biography; }

    public List<Book> getBooks() { return books; }

    public void setBooks(List<Book> books) { this.books = books; }

}
```

2.AuthorRrepository

```
package com.example.library;

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

public interface AuthorRepository extends JpaRepository<Author, Long> {
}
```

3.AuthorService

```
package com.example.library;



import java.util.List;


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

import org.springframework.stereotype.Service;

import org.springframework.transaction.annotation.Transactional;


@Service

@Transactional

public class AuthorService {

    @Autowired

    private AuthorRepository authorRepository;


    public Author addAuthor(Author author) {

        return authorRepository.save(author);

    }


    public List<Author> getAllAuthors() {
        return authorRepository.findAll();
    }

    public Author getAuthorById(Long id) {
        return authorRepository.findById(id)
                .orElseThrow(() -> new RuntimeException("Author not found"));
    }
}
```

4.Book.java

```
package com.example.library;


import java.util.Date;

import jakarta.persistence.Entity;
import jakarta.persistence.GeneratedValue;
import jakarta.persistence.GenerationType;
import jakarta.persistence.Id;
import jakarta.persistence.JoinColumn;
import jakarta.persistence.ManyToOne;
import jakarta.persistence.Temporal;
import jakarta.persistence.TemporalType;

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

    private String title;
    private String isbn;
    @Temporal(TemporalType.DATE)
    private Date publishedDate;

    private int numberOfCopies;

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

    // Getters and Setters
    public Long getId() { return id; }
    public void setId(Long id) { this.id = id; }
    public String getTitle() { return title; }
    public void setTitle(String title) { this.title = title; }
    public String getIsbn() { return isbn; }
    public void setIsbn(String isbn) { this.isbn = isbn; }
    public Date getPublishedDate() { return publishedDate; }
    public void setPublishedDate(Date publishedDate) { this.publishedDate = publishedDate; }
    public int getNumberOfCopies() { return numberOfCopies; }
    public void setNumberOfCopies(int numberOfCopies) { this.numberOfCopies = numberOfCopies; }
    public Author getAuthor() { return author; }
    public void setAuthor(Author author) { this.author = author; }
}
```

5.bookRepository.java

```
package com.example.library;

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

public interface BookRepository extends JpaRepository<Book, Long> {
    Book findByIsbn(String isbn);
}
```

6.booksServices.java

```
package com.example.library;



import java.util.List;


import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
import org.springframework.transaction.annotation.Transactional;


@Service
@Transactional
public class BookService {
   @Autowired
   private BookRepository bookRepository;


   public Book addBook(Book book) {
      return bookRepository.save(book);
   }


   public Book updateBook(Long id, Book bookDetails) {
      Book book = bookRepository.findById(id)
            .orElseThrow(() -> new RuntimeException("Book not found"));
      book.setTitle(bookDetails.getTitle());
      book.setIsbn(bookDetails.getIsbn());
      book.setPublishedDate(bookDetails.getPublishedDate());
      book.setNumberOfCopies(bookDetails.getNumberOfCopies());
      return bookRepository.save(book);
   }
```

```
    public List<Book> getAllBooks() {
        return bookRepository.findAll();
    }

    public Book getBookById(Long id) {
        return bookRepository.findById(id)
                .orElseThrow(() -> new RuntimeException("Book not found"));
    }

    public void deleteBook(Long id) {
        bookRepository.deleteById(id);
    }
}
```

7.LibraryController.java

```
package com.example.library;


import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
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.RequestMapping;
import org.springframework.web.bind.annotation.RestController;
```

```java
@RestController
@RequestMapping("/api")
public class LibraryController {
    @Autowired
    private BookService bookService;

    @Autowired
    private AuthorService authorService;

    // Book endpoints
    @PostMapping("/books")
    public Book addBook(@RequestBody Book book) {
        return bookService.addBook(book);
    }

    @GetMapping("/books")
    public List<Book> getAllBooks() {
        return bookService.getAllBooks();
    }

    @GetMapping("/books/{id}")
    public ResponseEntity<Book> getBookById(@PathVariable Long id) {
        return ResponseEntity.ok(bookService.getBookById(id));
    }

    @PutMapping("/books/{id}")
    public Book updateBook(@PathVariable Long id, @RequestBody Book bookDetails) {
        return bookService.updateBook(id, bookDetails);
    }

    @DeleteMapping("/books/{id}")
    public ResponseEntity<Void> deleteBook(@PathVariable Long id) {
        bookService.deleteBook(id);
        return ResponseEntity.noContent().build();
    }

    // Author endpoints
    @PostMapping("/authors")
    public Author addAuthor(@RequestBody Author author) {
        return authorService.addAuthor(author);
    }

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

    @GetMapping("/authors/{id}")
    public ResponseEntity<Author> getAuthorById(@PathVariable Long id) {
        return ResponseEntity.ok(authorService.getAuthorById(id));
    }
}
```

8.application.propreties

```
spring.application.name=library-managemnet-system
spring.datasource.url=jdbc:h2:mem:testdb
spring.datasource.driverClassName=org.h2.Driver
spring.datasource.username=sa
spring.datasource.password=password
spring.h2.console.enabled=true
spring.jpa.hibernate.ddl-auto=update
```

10.pom.xml

```
<?xml version="1.0" encoding="UTF-8"?>
```

```xml
<project xmlns="http://maven.apache.org/POM/4.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
	xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
https://maven.apache.org/xsd/maven-4.0.0.xsd">
	<modelVersion>4.0.0</modelVersion>
	<parent>
		<groupId>org.springframework.boot</groupId>
		<artifactId>spring-boot-starter-parent</artifactId>
		<version>3.3.4</version>
		<relativePath/> <!-- lookup parent from repository -->
	</parent>
	<groupId>com.example</groupId>
	<artifactId>library</artifactId>
	<version>0.0.1-SNAPSHOT</version>
	<name>library-managemnet-system</name>
	<description>Demo project for Spring Boot</description>
	<url/>
	<licenses>
		<license/>
	</licenses>
	<developers>
		<developer/>
	</developers>
	<scm>
		<connection/>
		<developerConnection/>
		<tag/>
		<url/>
	</scm>
	<properties>
		<java.version>17</java.version>
	</properties>
	<dependencies>
		<dependency>
			<groupId>org.springframework.boot</groupId>
			<artifactId>spring-boot-starter-data-jpa</artifactId>
		</dependency>
		<dependency>
			<groupId>org.springframework.boot</groupId>
			<artifactId>spring-boot-starter-web</artifactId>
		</dependency>

    <!-- Spring Boot Starter Test -->
    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-test</artifactId>
        <scope>test</scope>
    </dependency>


    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter</artifactId>
    </dependency>
    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-web</artifactId>
    </dependency>
```

```
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-devtools</artifactId>
            <scope>runtime</scope>
            <optional>true</optional>
        </dependency>
        <dependency>
            <groupId>com.h2database</groupId>
            <artifactId>h2</artifactId>
            <scope>runtime</scope>
        </dependency>
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-test</artifactId>
            <scope>test</scope>
        </dependency>
    </dependencies>

    <build>
        <plugins>
            <plugin>
                <groupId>org.springframework.boot</groupId>
                <artifactId>spring-boot-maven-plugin</artifactId>
            </plugin>
        </plugins>
    </build>

</project>
```