# Library API Project Setup and Files

## IDE Steps (IntelliJ IDEA)

1. Create New Spring Boot Project:

- File → New → Project…

- Select Spring Initializr, SDK Java 17, click Next.

- Group: com.example, Artifact: library-api, Name: library-api.

- Add Dependencies: Spring Web, Spring Data JPA, H2 Database, Spring Boot Validation.

- Click Finish.

2. Set Up Package Structure:

- Under src/main/java, create package com.example.library and sub-packages:

controller, dto, exception, model, repository, service.

- Mirror this under src/test/java.

3. Create Files:

- Right-click target package or src/main/resources → New → Java Class or File.

- Name it and paste contents as specified.

4. Run & Test:

- Run LibraryApiApplication.java as a Java application.

- Use Run → Edit Configurations… for VM options if needed.

- Run LibraryServiceTest as a JUnit test.

- Access API at http://localhost:8080/api.

## Project Files

### LibraryApiApplication.java

package com.example.library;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class LibraryApiApplication {

public static void main(String[] args) {

SpringApplication.run(LibraryApiApplication.class, args);

}

}

### Book.java

package com.example.library.model;

import jakarta.persistence.\*;

import jakarta.validation.constraints.NotBlank;

@Entity

@Table(name = "books")

public class Book {

@Id @GeneratedValue(strategy = GenerationType.IDENTITY)

private Long id;

@NotBlank private String isbn;

@NotBlank private String title;

@NotBlank private String author;

public Book() {}

public Book(String isbn, String title, String author) {

this.isbn = isbn; this.title = title; this.author = author;

}

// getters and setters

}

### application.yml

spring:

datasource:

url: jdbc:h2:mem:library;DB\_CLOSE\_DELAY=-1

driverClassName: org.h2.Driver

username: sa

password:

jpa:

hibernate:

ddl-auto: update

show-sql: true