Spring Core\_Maven

Exercise 9: Creating a Spring Boot Application

Out put :-

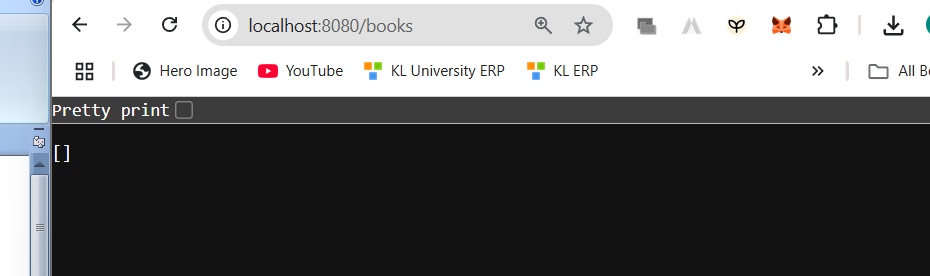
To run the application mvn spring-boot:run

we use GET POST to see the out put

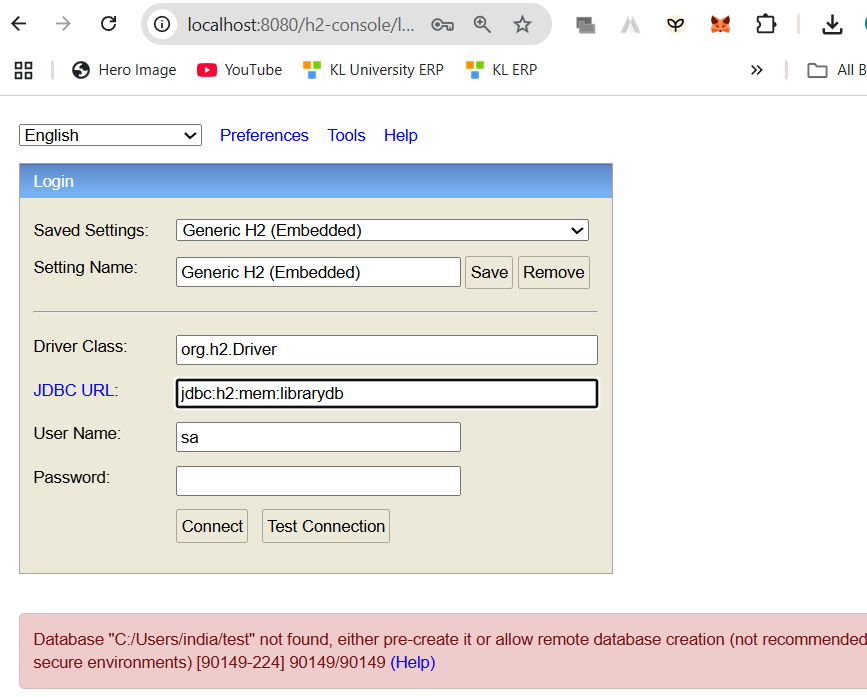
<http://localhost:8080/books>

you get this

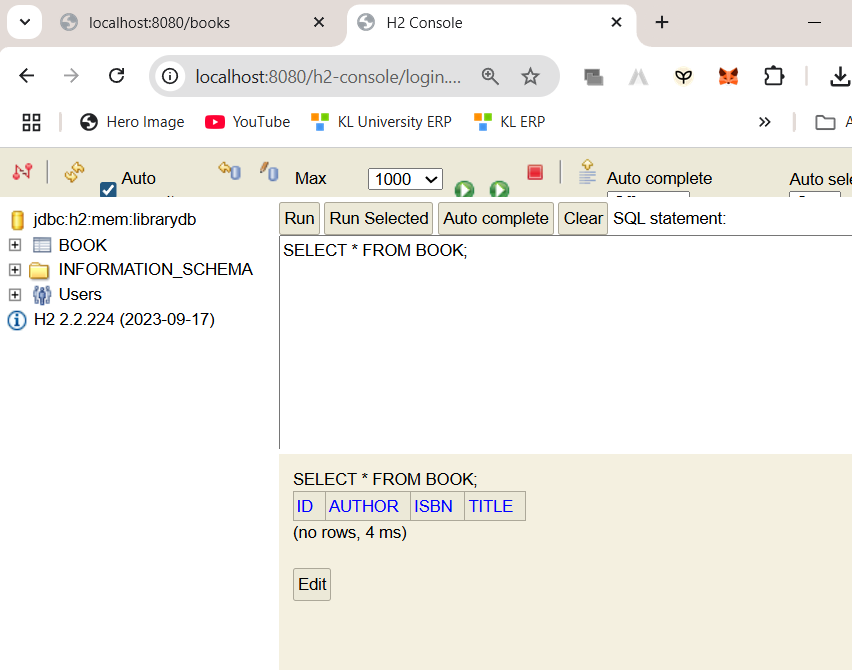
[ ] as output



If we do more changes



In the url use what you kept in application.properties.

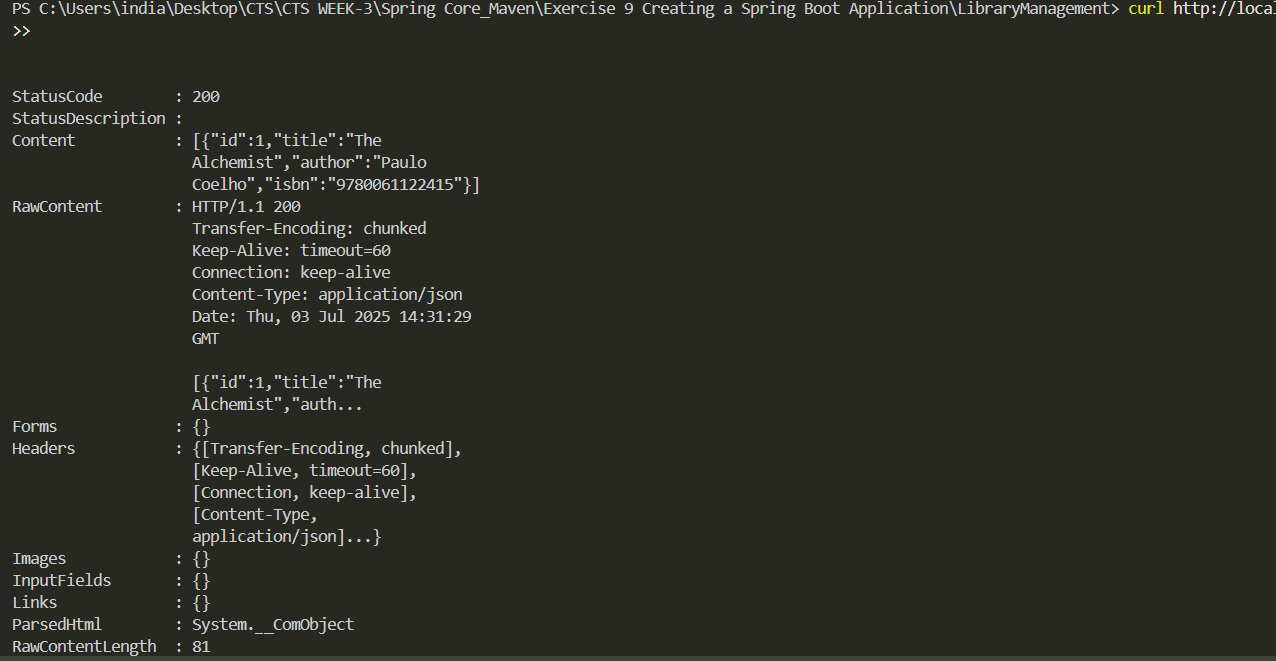


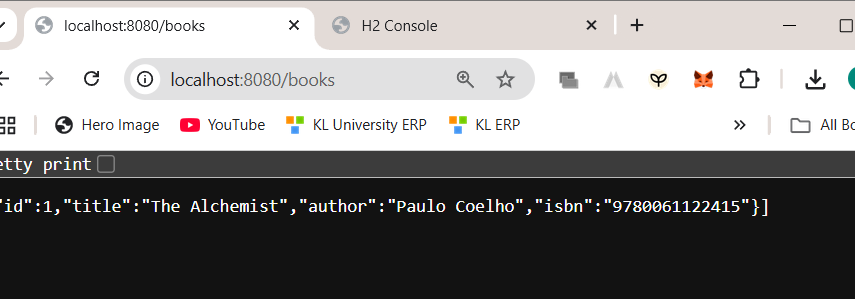
You will get to see this

curl -X POST http://localhost:8080/books -H "Content-Type: application/json" -d "{\"title\":\"The Alchemist\", \"author\":\"Paulo Coelho\", \"isbn\":\"9780061122415\"}"

try this on your cmd

you will see output in your terminal instead on POSTMAN

see the content you get the book 1 data



Dependencies used

**Spring Web, Spring Data JPA, and H2 Database**

**Create Application Properties**

# H2 in-memory DB

spring.datasource.url=jdbc:h2:mem:librarydb

spring.datasource.driverClassName=org.h2.Driver

spring.datasource.username=sa

spring.datasource.password=

spring.jpa.database-platform=org.hibernate.dialect.H2Dialect

spring.jpa.hibernate.ddl-auto=update

spring.h2.console.enabled=true

spring.h2.console.path=/h2-console

**Define Entities and Repositories:**

package com.example.librarymanagement.model;

import jakarta.persistence.Entity;

import jakarta.persistence.GeneratedValue;

import jakarta.persistence.GenerationType;

import jakarta.persistence.Id;

@*Entity*

public class Book {

    @*Id*

    @*GeneratedValue*(strategy = GenerationType.IDENTITY)

    private *Long* id;

    private *String* title;

    private *String* author;

    private *String* isbn;

    // Getters & 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* getAuthor() { return author; }

    public *void* setAuthor(*String* *author*) { this.author = author; }

    public *String* getIsbn() { return isbn; }

    public *void* setIsbn(*String* *isbn*) { this.isbn = isbn; }

}

package com.example.librarymanagement.repository;

import com.example.librarymanagement.model.Book;

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

public interface BookRepository extends *JpaRepository*<*Book*, *Long*> {

}

**Create a REST Controller:**

package com.example.librarymanagement.controller;

import com.example.librarymanagement.model.Book;

import com.example.librarymanagement.repository.BookRepository;

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

import org.springframework.web.bind.annotation.\*;

import java.util.List;

@*RestController*

@*RequestMapping*("/books")

public class BookController {

    @*Autowired*

    private *BookRepository* bookRepository;

    @*GetMapping*

    public *List*<*Book*> getAllBooks() {

        return bookRepository.findAll();

    }

    @*GetMapping*("/{id}")

    public *Book* getBookById(@*PathVariable* *Long* *id*) {

        return bookRepository.findById(id).orElse(null);

    }

    @*PostMapping*

    public *Book* createBook(@*RequestBody* *Book* *book*) {

        return bookRepository.save(book);

    }

    @*PutMapping*("/{id}")

    public *Book* updateBook(@*PathVariable* *Long* *id*, @*RequestBody* *Book* *bookDetails*) {

*Book* book = bookRepository.findById(id).orElse(null);

        if (book != null) {

            book.setTitle(bookDetails.getTitle());

            book.setAuthor(bookDetails.getAuthor());

            book.setIsbn(bookDetails.getIsbn());

            return bookRepository.save(book);

        }

        return null;

    }

    @*DeleteMapping*("/{id}")

    public *void* deleteBook(@*PathVariable* *Long* *id*) {

        bookRepository.deleteById(id);

    }

}