**Exercise 14: Online Bookstore - Integration Testing for REST Services**

**BookControllerIntegrationTest**

package com.bookstore.api.controllers;

import com.bookstore.api.dtos.BookDTO;

import com.bookstore.api.entities.Book;

import com.bookstore.api.mappers.BookMapper;

import com.bookstore.api.repositories.BookRepository;

import com.bookstore.api.services.BookService;

import org.junit.jupiter.api.Test;

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

import org.springframework.boot.test.autoconfigure.web.servlet.AutoConfigureMockMvc;

import org.springframework.boot.test.context.SpringBootTest;

import org.springframework.http.MediaType;

import org.springframework.test.web.servlet.MockMvc;

import static org.springframework.test.web.servlet.request.MockMvcRequestBuilders.\*;

import static org.springframework.test.web.servlet.result.MockMvcResultMatchers.\*;

@SpringBootTest

@AutoConfigureMockMvc

public class BookControllerIntegrationTest {

@Autowired

private MockMvc mockMvc;

@Autowired

private BookRepository bookRepository;

@Autowired

private BookMapper bookMapper;

@Autowired

private BookService bookService;

@Test

public void testGetAllBooks() throws Exception {

// Arrange

Book book = new Book();

bookRepository.save(book);

// Act and Assert

mockMvc.perform(get("/books"))

.andExpect(status().isOk())

.andExpect(jsonPath("$", hasSize(1)));

}

@Test

public void testGetBookById() throws Exception {

// Arrange

Book book = new Book();

bookRepository.save(book);

// Act and Assert

mockMvc.perform(get("/books/" + book.getId()))

.andExpect(status().isOk())

.andExpect(jsonPath("$.id").value(book.getId()));

}

@Test

public void testCreateBook() throws Exception {

// Arrange

BookDTO bookDTO = new BookDTO();

// Act and Assert

mockMvc.perform(post("/books")

.contentType(MediaType.APPLICATION\_JSON)

.content(objectMapper.writeValueAsString(bookDTO)))

.andExpect(status().isCreated())

.andExpect(jsonPath("$.id").isNotEmpty());

}

// ...

}

In the above code, i have written integration tests for the bookstore's RESTful services.

- I have set up Spring Test for integration testing.

- I have used MockMvc for end-to-end testing of the REST endpoints.

- I have included database integration in our tests using an in-memory database like H2.

Note that i have used @SpringBootTest to enable Spring Boot integration testing, and @AutoConfigureMockMvc to enable MockMvc. I have also injected the necessary dependencies, such as the book repository, book mapper, and book service.