<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-test</artifactId>

<scope>test</scope>

</dependency>

<dependency>

<groupId>org.mockito</groupId>

<artifactId>mockito-core</artifactId>

<scope>test</scope>

</dependency>

testImplementation 'org.springframework.boot:spring-boot-starter-test'

testImplementation 'org.mockito:mockito-core'

import static org.mockito.Mockito.when;

import static org.springframework.test.web.servlet.request.MockMvcRequestBuilders.get;

import static org.springframework.test.web.servlet.result.MockMvcResultMatchers.status;

import static org.springframework.test.web.servlet.result.MockMvcResultMatchers.jsonPath;

import com.example.bookstore.controller.BookController;

import com.example.bookstore.model.Book;

import com.example.bookstore.service.BookService;

import org.junit.jupiter.api.BeforeEach;

import org.junit.jupiter.api.Test;

import org.mockito.InjectMocks;

import org.mockito.Mock;

import org.mockito.MockitoAnnotations;

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

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

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

import org.springframework.test.web.servlet.setup.MockMvcBuilders;

import java.util.Arrays;

import java.util.List;

@WebMvcTest(BookController.class)

public class BookControllerTest {

@Autowired

private MockMvc mockMvc;

@Mock

private BookService bookService;

@InjectMocks

private BookController bookController;

@BeforeEach

public void setup() {

MockitoAnnotations.openMocks(this);

mockMvc = MockMvcBuilders.standaloneSetup(bookController).build();

}

@Test

public void testGetBooks() throws Exception {

List<Book> books = Arrays.asList(

new Book(1L, "Effective Java", "Joshua Bloch", "Programming", 45.00),

new Book(2L, "Clean Code", "Robert C. Martin", "Programming", 50.00)

);

when(bookService.getAllBooks()).thenReturn(books);

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

.andExpect(status().isOk())

.andExpect(jsonPath("$[0].title").value("Effective Java"))

.andExpect(jsonPath("$[1].title").value("Clean Code"));

}

}