Junit and Mockito Setup

Add Dependencies

<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>

Create Test Class for REST Controller

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 static org.hamcrest.Matchers.\*;

import com.example.bookstore.controller.BookController;

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.boot.test.mock.mockito.MockBean;

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

@WebMvcTest(BookController.class)

public class BookControllerTest {

@Autowired

private MockMvc mockMvc;

@MockBean

private BookService bookService;

@BeforeEach

void setUp() {

MockitoAnnotations.openMocks(this);

}

@Test

public void testGetBookById() throws Exception {

// Mock behavior

when(bookService.getBook(1L)).thenReturn(new BookDTO(1L, "Test Book", "Author", 100));

// Perform GET request

mockMvc.perform(get("/api/books/1"))

.andExpect(status().isOk())

.andExpect(jsonPath("$.title", is("Test Book")))

.andExpect(jsonPath("$.author", is("Author")))

.andExpect(jsonPath("$.price", is(100)));

}

}

Testing POST Requests

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

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

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

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

import com.fasterxml.jackson.databind.ObjectMapper;

@Test

public void testCreateBook() throws Exception {

BookDTO newBook = new BookDTO(null, "New Book", "New Author", 200);

// Mock behavior

when(bookService.createBook(any(BookDTO.class))).thenReturn(new BookDTO(2L, "New Book", "New Author", 200));

mockMvc.perform(post("/api/books")

.contentType("application/json")

.content(new ObjectMapper().writeValueAsString(newBook)))

.andExpect(status().isCreated())

.andExpect(jsonPath("$.id", is(2)))

.andExpect(jsonPath("$.title", is("New Book")))

.andExpect(jsonPath("$.author", is("New Author")))

.andExpect(jsonPath("$.price", is(200)));

}

Ensure Comprehensive Test Coverage

@Test

public void testCreateBook\_InvalidInput() throws Exception {

BookDTO newBook = new BookDTO(null, "", "", -50);

mockMvc.perform(post("/api/books")

.contentType("application/json")

.content(new ObjectMapper().writeValueAsString(newBook)))

.andExpect(status().isBadRequest())

.andExpect(jsonPath("$.errors", hasSize(2)));

}