<dependency>

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

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

<scope>test</scope>

</dependency>

<dependency>

<groupId>com.h2</groupId>

<artifactId>h2</artifactId>

<scope>test</scope>

</dependency>

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

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

import com.fasterxml.jackson.databind.ObjectMapper;

import org.junit.jupiter.api.BeforeEach;

import org.junit.jupiter.api.Test;

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

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

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

import org.springframework.context.annotation.Import;

import org.springframework.test.context.junit.jupiter.SpringJUnitConfig;

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

import org.springframework.transaction.annotation.Transactional;

@SpringBootTest

@Import({BookController.class, BookService.class})

@Transactional

public class BookControllerIntegrationTest {

@Autowired

private MockMvc mockMvc;

@Autowired

private ObjectMapper objectMapper;

@Test

public void testGetAllBooks() throws Exception {

// Act and Assert

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

.andExpect(status().isOk())

.andExpect(content().contentType("application/json"))

.andExpect(jsonPath("$[0].id").value(1))

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

.andExpect(jsonPath("$[0].author").value("Author"));

}

@Test

public void testAddBook() throws Exception {

// Arrange

Book book = new Book(1L, "New Title", "New Author");

// Act and Assert

mockMvc.perform(post("/books")

.contentType("application/json")

.content(objectMapper.writeValueAsString(book)))

.andExpect(status().isCreated())

.andExpect(jsonPath("$.id").value(1))

.andExpect(jsonPath("$.title").value("New Title"))

.andExpect(jsonPath("$.author").value("New Author"));

}

@Test

public void testGetBookById() throws Exception {

// Act and Assert

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

.andExpect(status().isOk())

.andExpect(content().contentType("application/json"))

.andExpect(jsonPath("$.id").value(1))

.andExpect(jsonPath("$.title").value("Title"))

.andExpect(jsonPath("$.author").value("Author"));

}

}

# Use H2 in-memory database for testing

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

spring.datasource.driver-class-name=org.h2.Driver

spring.datasource.username=sa

spring.datasource.password=password

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

# Enable schema creation on startup

spring.jpa.hibernate.ddl-auto=create-drop

import org.springframework.test.context.ActiveProfiles;

@SpringBootTest

@Import({BookController.class, BookService.class})

@Transactional

@ActiveProfiles("test") // Load application-test.properties

public class BookControllerIntegrationTest {

// Test methods as before

}