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.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.put;

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

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

@SpringBootTest

@AutoConfigureMockMvc

public class ProductRoutesTests {

@Autowired

private MockMvc mockMvc;

@Autowired

private ProductRepository productRepository;

private Product testProduct;

@BeforeEach

public void setUp() {

// Create and save a test product

testProduct = new Product("Old Product", 25.0, true);

productRepository.save(testProduct);

}

@Test

public void testUpdateProduct() throws Exception {

// Create a JSON representation of the updated product

String updatedProductJson = "{\"name\":\"Updated Product\", \"price\":30.0, \"availability\":false}";

// Perform a PUT request to update the product by ID

mockMvc.perform(put("/products/" + testProduct.getId())

.contentType(MediaType.APPLICATION\_JSON)

.content(updatedProductJson))

.andExpect(status().isOk())

.andExpect(jsonPath("$.name").value("Updated Product"))

.andExpect(jsonPath("$.price").value(30.0))

.andExpect(jsonPath("$.availability").value(false));

}

}