**Exercise 4: Integration Test with Spring Boot**

**Task:** Write an integration test that tests the full flow from controller to database.

src/main/java/

├── controller/UserController.java

├── model/User.java

├── repository/UserRepository.java

├── service/UserService.java

src/test/java/

└── UserIntegrationTest.java

**User.java :-**

@Entity

public class User {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private Long id;

private String name;

private String email;

}

**UserIntegrationTest.java :-**

package com.example.demo;

import com.example.demo.model.User;

import com.example.demo.repository.UserRepository;

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.AutoConfigureMockMvc;

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

import org.springframework.http.MediaType;

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

import java.util.Optional;

import static org.assertj.core.api.Assertions.assertThat;

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

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

@SpringBootTest

@AutoConfigureMockMvc

public class UserIntegrationTest {

@Autowired

private MockMvc mockMvc;

@Autowired

private UserRepository userRepository;

@Autowired

private ObjectMapper objectMapper;

@BeforeEach

void cleanDatabase() {

userRepository.deleteAll(); // Reset database before each test

}

@Test

public void fullFlow\_createUserAndRetrieveFromDatabase() throws Exception {

// 1. Prepare the user object

User user = new User();

user.setName("Naresh");

user.setEmail("naresh@example.com");

// 2. Create user via POST /users

String userJson = objectMapper.writeValueAsString(user);

String response = mockMvc.perform(post("/users")

.contentType(MediaType.APPLICATION\_JSON)

.content(userJson))

.andExpect(status().isCreated())

.andExpect(jsonPath("$.id").exists())

.andExpect(jsonPath("$.name").value("Naresh"))

.andExpect(jsonPath("$.email").value("naresh@example.com"))

.andReturn()

.getResponse()

.getContentAsString();

User createdUser = objectMapper.readValue(response, User.class);

// 3. Verify user exists in the database

Optional<User> dbUser = userRepository.findById(createdUser.getId());

assertThat(dbUser).isPresent();

assertThat(dbUser.get().getEmail()).isEqualTo("naresh@example.com");

// 4. Get user via GET /users/{id}

mockMvc.perform(get("/users/" + createdUser.getId()))

.andExpect(status().isOk())

.andExpect(jsonPath("$.name").value("Naresh"))

.andExpect(jsonPath("$.email").value("naresh@example.com"));

}

}

**Pom.xml :-**

<project xmlns="http://maven.apache.org/POM/4.0.0"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://maven.apache.org/POM/4.0.0

http://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>com.example</groupId>

<artifactId>user-demo</artifactId>

<version>1.0.0</version>

<packaging>jar</packaging>

<name>User Demo</name>

<description>Spring Boot Integration Test Example</description>

<properties>

<java.version>17</java.version>

<spring.boot.version>3.2.0</spring.boot.version>

</properties>

<dependencies>

<!-- Spring Boot Starter for Web -->

<dependency>

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

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

</dependency>

<!-- Spring Boot Starter for JPA -->

<dependency>

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

<artifactId>spring-boot-starter-data-jpa</artifactId>

</dependency>

<!-- H2 Database (In-memory DB for testing) -->

<dependency>

<groupId>com.h2database</groupId>

<artifactId>h2</artifactId>

<scope>runtime</scope>

</dependency>

<!-- JSON Handling -->

<dependency>

<groupId>com.fasterxml.jackson.core</groupId>

<artifactId>jackson-databind</artifactId>

</dependency>

<!-- Spring Boot Test (JUnit, MockMvc, etc.) -->

<dependency>

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

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

<scope>test</scope>

</dependency>

</dependencies>

<build>

<plugins>

<!-- Spring Boot Maven Plugin -->

<plugin>

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

<artifactId>spring-boot-maven-plugin</artifactId>

<version>${spring.boot.version}</version>

</plugin>

<!-- Compiler Plugin -->

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-compiler-plugin</artifactId>

<version>3.10.1</version>

<configuration>

<source>${java.version}</source>

<target>${java.version}</target>

<compilerArgs>

<arg>-parameters</arg> <!-- Helps with reflection-based param name resolution -->

</compilerArgs>

</configuration>

</plugin>

</plugins>

</build>

</project>

Output :-

