**Exercise 2: Mocking a Repository in a Service Test**

**Task:** Test a service that uses a repository to fetch data.

**User.java :-**

package org.example1.entity;

import jakarta.persistence.Entity;

import jakarta.persistence.Id;

@Entity

public class User {

@Id

private Long id;

private String name;

// Constructors

public User() {}

public User(Long id, String name) {

this.id = id;

this.name = name;

}

// Getters & Setters

public Long getId() { return id; }

public void setId(Long id) { this.id = id; }

public String getName() { return name; }

public void setName(String name) { this.name = name; }

}

package org.example1.repository;  
  
import org.example1.entity.User;  
import org.springframework.data.jpa.repository.JpaRepository;  
  
public interface UserRepository extends JpaRepository<User, Long> {  
}

**UserService.java :-**

package org.example1.service;  
  
import org.example1.entity.User;  
import org.example1.repository.UserRepository;  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.stereotype.Service;  
  
@Service  
public class UserService {  
  
 @Autowired  
 private UserRepository userRepository;  
  
 public User getUserById(Long id) {  
 return userRepository.findById(id).orElse(null);  
 }  
}

**UserServiceTest.java :-**

package org.example1.service;  
  
import org.example1.entity.User;  
import org.example1.repository.UserRepository;  
import org.junit.jupiter.api.Test;  
import org.mockito.InjectMocks;  
import org.mockito.Mock;  
import org.mockito.MockitoAnnotations;  
  
import java.util.Optional;  
  
import static org.junit.jupiter.api.Assertions.\*;  
import static org.mockito.Mockito.\*;  
  
public class UserServiceTest {  
  
 @Mock  
 private UserRepository userRepository;  
  
 @InjectMocks  
 private UserService userService;  
  
 public UserServiceTest() {  
 MockitoAnnotations.*openMocks*(this); // Initialize mocks  
 }  
  
 @Test  
 public void testGetUserById\_ReturnsUser() {  
 // Arrange  
 User mockUser = new User(1L, "John Doe");  
 *when*(userRepository.findById(1L)).thenReturn(Optional.*of*(mockUser));  
  
 // Act  
 User result = userService.getUserById(1L);  
  
 // Assert  
 *assertNotNull*(result);  
 *assertEquals*("John Doe", result.getName());  
 *verify*(userRepository, *times*(1)).findById(1L);  
 }  
  
 @Test  
 public void testGetUserById\_ReturnsNull() {  
 // Arrange  
 *when*(userRepository.findById(2L)).thenReturn(Optional.*empty*());  
  
 // Act  
 User result = userService.getUserById(2L);  
  
 // Assert  
 *assertNull*(result);  
 *verify*(userRepository, *times*(1)).findById(2L);  
 }  
}

**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>org.example1</groupId>  
 <artifactId>mocking-repository</artifactId>  
 <version>1.0-SNAPSHOT</version>  
  
 <properties>  
 <java.version>17</java.version>  
 </properties>  
  
 <dependencies>  
 <!-- Spring Data JPA -->  
 <dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-data-jpa</artifactId>  
 <version>3.2.5</version>  
 </dependency>  
  
 <!-- JUnit 5 -->  
 <dependency>  
 <groupId>org.junit.jupiter</groupId>  
 <artifactId>junit-jupiter</artifactId>  
 <version>5.10.0</version>  
 <scope>test</scope>  
 </dependency>  
  
 <!-- Mockito -->  
 <dependency>  
 <groupId>org.mockito</groupId>  
 <artifactId>mockito-core</artifactId>  
 <version>5.11.0</version>  
 <scope>test</scope>  
 </dependency>  
  
 <!-- Jakarta Persistence for @Entity and @Id -->  
 <dependency>  
 <groupId>jakarta.persistence</groupId>  
 <artifactId>jakarta.persistence-api</artifactId>  
 <version>3.1.0</version>  
 </dependency>  
 </dependencies>  
  
 <build>  
 <plugins>  
 <!-- Java Compiler -->  
 <plugin>  
 <groupId>org.apache.maven.plugins</groupId>  
 <artifactId>maven-compiler-plugin</artifactId>  
 <version>3.11.0</version>  
 <configuration>  
 <source>${java.version}</source>  
 <target>${java.version}</target>  
 </configuration>  
 </plugin>  
  
 <!-- Test Runner -->  
 <plugin>  
 <groupId>org.apache.maven.plugins</groupId>  
 <artifactId>maven-surefire-plugin</artifactId>  
 <version>3.1.2</version>  
 </plugin>  
 </plugins>  
 </build>  
</project>

**Output :-**

