Exercise 2: Mocking a Repository in a Service Test

**User.java:**

package com.example.springtest;

public class User {

private Long id;

private String name;

public User() {}

public User(Long id, String name) {

this.id = id;

this.name = name;

}

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;

}

}

**UserRepository.java:**

package com.example.springtest;

import java.util.Optional;

public interface UserRepository {

Optional<User> findById(Long id);

}

**UserService.java:**

package com.example.springtest;

public class UserService {

private final UserRepository userRepository;

public UserService(UserRepository userRepository) {

this.userRepository = userRepository;

}

public User getUserById(Long id) {

return userRepository.findById(id).orElse(null);

}

}

**UserServiceTest.java:**

package com.example.springtest;

import org.junit.jupiter.api.Test;

import java.util.Optional;

import static org.junit.jupiter.api.Assertions.*assertEquals*;

import static org.mockito.Mockito.\*;

public class UserServiceTest {

@Test

public void testGetUserById() {

UserRepository mockRepo = *mock*(UserRepository.class);

User mockUser = new User(1L, "Aswin");

*when*(mockRepo.findById(1L)).thenReturn(Optional.*of*(mockUser));

UserService service = new UserService(mockRepo);

User result = service.getUserById(1L);

*assertEquals*("Aswin", result.getName());

}

}

**Output:**

