**TALLER 1 Dia 2**

**OBJETIVO: Implementar desde 0 un proyecto**

* **CONFIGURAR POM.XML**

<dependencies>

<!-- JUnit 5 -->

<dependency>

<groupId>org.junit.jupiter</groupId>

<artifactId>junit-jupiter-api</artifactId>

<version>5.8.1</version>

<scope>test</scope>

</dependency>

<dependency>

<groupId>org.junit.jupiter</groupId>

<artifactId>junit-jupiter-engine</artifactId>

<version>5.8.1</version>

<scope>test</scope>

</dependency>

<!-- Mockito -->

<dependency>

<groupId>org.mockito</groupId>

<artifactId>mockito-core</artifactId>

<version>4.0.0</version>

<scope>test</scope>

</dependency>

<dependency>

<groupId>org.mockito</groupId>

<artifactId>mockito-junit-jupiter</artifactId>

<version>4.0.0</version>

<scope>test</scope>

</dependency>

</dependencies>

* **IMPLEMENTAR LAS CLASES**

**UserRepository.java** (Interfaz del repositorio)

package com.example.repository;

import com.example.model.User;

public interface UserRepository {

User findById(Long id);

User save(User user);

void delete(Long id);

}

**User.java** (Modelo de usuario):

package com.example.model;

public class User {

private Long id;

private String name;

// Getters y 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;

}

}

**UserService.java** (Clase de servicio)

package com.example.service;

import com.example.model.User;

import com.example.repository.UserRepository;

public class UserService {

private final UserRepository userRepository;

public UserService(UserRepository userRepository) {

this.userRepository = userRepository;

}

public User getUserById(Long id) {

return userRepository.findById(id);

}

public User saveUser(User user) {

return userRepository.save(user);

}

public void deleteUser(Long id) {

userRepository.delete(id);

}

}

**ESCRIBIR PRUEBAS CON MOCKITO**

package com.example.service;

import com.example.model.User;

import com.example.repository.UserRepository;

import org.junit.jupiter.api.BeforeEach;

import org.junit.jupiter.api.Test;

import org.mockito.InjectMocks;

import org.mockito.Mock;

import org.mockito.MockitoAnnotations;

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

import static org.mockito.Mockito.\*;

class UserServiceTest {

@Mock

private UserRepository userRepository;

@InjectMocks

private UserService userService;

@BeforeEach

void setUp() {

MockitoAnnotations.openMocks(this);

}

@Test

void testGetUserById() {

User user = new User();

user.setId(1L);

user.setName("John Doe");

when(userRepository.findById(1L)).thenReturn(user);

User result = userService.getUserById(1L);

assertEquals("John Doe", result.getName());

verify(userRepository, times(1)).findById(1L);

}

@Test

void testSaveUser() {

User user = new User();

user.setName("Jane Doe");

when(userRepository.save(user)).thenReturn(user);

User result = userService.saveUser(user);

assertEquals("Jane Doe", result.getName());

verify(userRepository, times(1)).save(user);

}

@Test

void testDeleteUser() {

doNothing().when(userRepository).delete(1L);

userService.deleteUser(1L);

verify(userRepository, times(1)).delete(1L);

}

}