

unit tests using **JUnit** and **Mockito**, two widely-used testing frameworks

**JUnit** - popular Java framework for writing unit tests. It provides annotations and assertions to facilitate the writing and execution of tests.

**Mockito** - is a mocking framework that allows you to mock dependencies and interactions between objects in your tests. It helps isolate the code under test and makes it easier to focus on specific units of behavior.

**Testing Service Example**

**OrganizationService**

package com.example.junit\_testing.service;  
  
import com.example.junit\_testing.dto.OrganizationDto;  
import com.example.junit\_testing.entity.Organization;  
import com.example.junit\_testing.repository.OrganizationRepository;  
import lombok.RequiredArgsConstructor;  
import org.springframework.stereotype.Service;  
  
import java.util.List;  
  
@Service  
@RequiredArgsConstructor  
public class OrganizationService {  
 private final OrganizationRepository organizationRepository;

**01**

public List<Organization> getAllOrganizationDetails() {  
 return organizationRepository.findAll();  
}

**02**

public Organization getOrganizationDetailsById(Long id) {  
 return organizationRepository.findById(id).orElseThrow(() -> new NullPointerException("there is no records"));  
}

**03**

public Organization addOrganization(OrganizationDto organizationDto) {  
 Organization organization = Organization.*builder*()  
 .name(organizationDto.getName())  
 .location(organizationDto.getLocation())  
 .numberOfEmployees(organizationDto.getNumberOfEmployees())  
 .build();  
 return organizationRepository.save(organization);  
}

**04**

public Organization updateOrganizationDetails(Organization organization) {  
 Organization existingOrganization = organizationRepository.findById(organization.getId()).orElse(null);  
 if (existingOrganization == null){  
 throw new NullPointerException("there is no records in the given id");  
 }else{  
 existingOrganization.setName(organization.getName());  
 existingOrganization.setLocation(organization.getLocation());  
 existingOrganization.setNumberOfEmployees(organization.getNumberOfEmployees());  
 return organizationRepository.save(existingOrganization);  
 }  
}

**05**

public String deleteOrganization(Long id) throws NoSuchFieldException {  
 boolean isExist = organizationRepository.existsById(id);  
 if(isExist){  
 organizationRepository.deleteById(id);  
 return "deleted successfully";  
 }else{  
 throw new NoSuchFieldException("there is no records given id");  
 }  
  
 }  
}

**OrganizationServiceTest**

@ExtendWith(MockitoExtension.class)  
class OrganizationServiceTest {  
  
 @Mock  
 private OrganizationRepository organizationRepository;  
  
 @InjectMocks  
 private OrganizationService organizationService;  
  
 private OrganizationDto organizationDto;  
 private Organization organization;

@ExtendWith(MockitoExtension.class) tells JUnit to use the MockitoExtension to initialize mocks.

@Mock Creates mock instances of the dependencies, here it's UserRepository.

@InjectMocks injects the mocks into the userService instance.

**01**

@Test  
void getAllOrganizationDetails() {  
 Organization organization = Organization.*builder*()  
 .name("codelantic")  
 .location("jaffna")  
 .numberOfEmployees(10)  
 .build();  
 *when*(organizationRepository.findAll()).thenReturn(List.*of*(organization));  
 List<Organization> organizations = organizationService.getAllOrganizationDetails();  
 *assertNotNull*(organizations);  
 *assertEquals*("codelantic",organizations.get(0).getName());  
 *verify*(organizationRepository,*times*(1)).findAll();  
}

* The **when** method in Mockito is used to stub a method call. It defines what should be returned (or thrown) when a specific method is called on a mock object.

**02**

@Test  
void getOrganizationDetailsById\_Success() {  
 organization = Organization.*builder*()  
 .name("codelantic")  
 .location("jaffna")  
 .numberOfEmployees(10)  
 .build();  
  
 *when*(organizationRepository.findById(1L)).thenReturn(Optional.*ofNullable*(organization));  
 Organization orgResponse = organizationService.getOrganizationDetailsById(1L);  
 *assertNotNull*(orgResponse);  
 *assertEquals*("codelantic",orgResponse.getName());  
 *verify*(organizationRepository,*times*(1)).findById(1L);  
}

**03**

@Test  
void getOrganizationDetailsById\_NotFound(){  
 Organization organization1;  
 *when*(organizationRepository.findById(2L)).thenReturn(Optional.*empty*());  
  
 Exception exception = *assertThrows*(NullPointerException.class, () -> organizationService.getOrganizationDetailsById(2L));  
 *assertEquals*("there is no records",exception.getMessage());  
 *verify*(organizationRepository,*times*(1)).findById(2L);  
}

**04**

@Test  
void addOrganization() {  
  
 Organization organization = Organization.*builder*()  
 .name("codelantic")  
 .location("jaffna")  
 .numberOfEmployees(10)  
 .build();  
 organizationDto = OrganizationDto.*builder*()  
 .name("codelantic")  
 .location("jaffna")  
 .numberOfEmployees(10)  
 .build();  
 *when*(organizationRepository.save(organization)).thenReturn(organization);  
 Organization organization1 = organizationService.addOrganization(organizationDto);  
  
 *assertNotNull*(organization1);  
 *assertEquals*("codelantic",organization1.getName());  
 *verify*(organizationRepository,*times*(1)).save(organization);  
}

**05**

@Test  
void updateOrganizationDetails() {  
 Organization organization = Organization.*builder*()  
 .id(1L)  
 .name("codelantic")  
 .location("jaffna")  
 .numberOfEmployees(10)  
 .build();  
 *when*(organizationRepository.findById(organization.getId())).thenReturn(Optional.*of*(organization));  
 *when*(organizationRepository.save(organization)).thenReturn(organization);  
  
 Organization organization1 = organizationService.updateOrganizationDetails(organization);  
 *assertNotNull*(organization1);  
}

**06**

@Test  
void deleteOrganization() throws NoSuchFieldException {  
 Organization organization = Organization.*builder*()  
 .id(1L)  
 .name("codelantic")  
 .location("jaffna")  
 .numberOfEmployees(10)  
 .build();  
 *when*(organizationRepository.existsById(organization.getId())).thenReturn(true);  
 String response = organizationService.deleteOrganization(organization.getId());  
  
}