**Exercise 3.4: Handling Void Methods**

**CODE:**

**Notifier Interface: -**

package org.example;  
public interface Notifier{  
 void send(String message);  
}

**MyService Class: -**

package org.example;  
  
public class MyService{  
 private Notifier notifier;  
 public MyService(Notifier notifier){  
 this.notifier=notifier;  
 }  
  
 public void checkCondition(boolean condition){  
 if(condition){  
 System.*out*.println("Condition met, raising alert.");  
 notifier.send("ALERT!");  
 }else{  
 System.*out*.println("Condition not met, no alert.");  
 }  
 }  
}

**MyServiceTest Class: -**

package org.example;  
import org.junit.jupiter.api.Test;  
import static org.mockito.ArgumentMatchers.*anyString*;  
import static org.mockito.Mockito.\*;  
  
class MyServiceTest {  
 @Test  
 public void testVoidMethodCalled(){  
 Notifier mockNotifier = *mock*(Notifier.class);  
 *doNothing*().when(mockNotifier).send(*anyString*());  
 MyService service = new MyService(mockNotifier);  
 service.checkCondition(true);  
 *verify*(mockNotifier).send("ALERT!");  
 System.*out*.println("Verified: send() was called with 'ALERT!'");  
 }  
}

**OUTPUT:**

