**1. NUnit-Handson**

**Calculator.cs**

namespace CalcLibrary

{

public class Calculator

{

public int Add(int a, int b)

{

return a + b;

}

}

}

**MathLibrary.cs**

using System;

namespace CalcLibrary

{

interface IMathLibrary

{

double Addition(double a, double b);

double Subtraction(double a, double b);

double Multiplication(double a, double b);

double Division(double a, double b);

}

public class SimpleCalculator : IMathLibrary

{

double result = 0;

public double Addition(double a, double b)

{

result = a + b;

return result;

}

public double Subtraction(double a, double b)

{

result = a - b;

return result;

}

public double Multiplication(double a, double b)

{

result = a \* b;

return result;

}

public double Division(double a, double b)

{

if (b == 0)

throw new ArgumentException("Second Parameter Can't be Zero");

result = a / b;

return result;

}

public void AllClear()

{

result = 0;

}

public double GetResult

{

get { return result; }

}

}

}

**CalculatorTests.cs**

using NUnit.Framework;

using CalcLibrary;

namespace CalcLibrary.Tests

{

[TestFixture]

public class CalculatorTests

{

private Calculator \_calculator;

[SetUp]

public void Setup()

{

\_calculator = new Calculator();

}

[TearDown]

public void Teardown()

{

\_calculator = null;

}

[Test]

[TestCase(2, 3, 5)]

[TestCase(-1, 4, 3)]

[TestCase(0, 0, 0)]

public void Add\_WhenCalled\_ReturnsCorrectSum(int a, int b, int expected)

{

int result = \_calculator.Add(a, b);

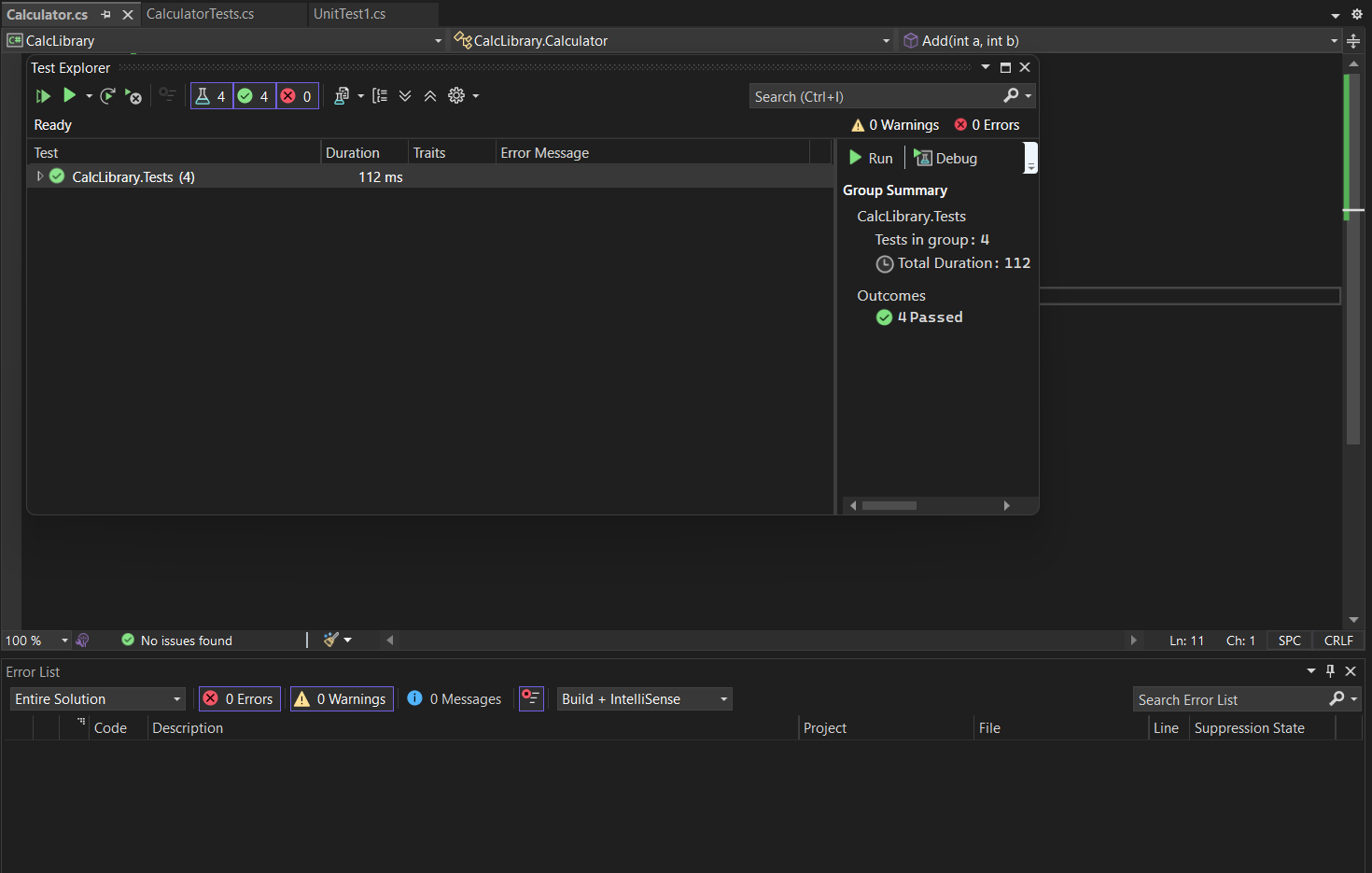
Assert.That(result, Is.EqualTo(expected));

}

}

}

**Output**

****

1. **Moq-Handson**

**MailSender.cs**

using System.Net;

using System.Net.Mail;

namespace CustomerCommLib

{

public interface IMailSender

{

bool SendMail(string toAddress, string message);

}

public class MailSender : IMailSender

{

public bool SendMail(string toAddress, string message)

{

MailMessage mail = new MailMessage();

SmtpClient smtpServer = new SmtpClient("smtp.gmail.com");

mail.From = new MailAddress("your\_email\_address@gmail.com");

mail.To.Add(toAddress);

mail.Subject = "Test Mail";

mail.Body = message;

smtpServer.Port = 587;

smtpServer.Credentials = new NetworkCredential("username", "password");

smtpServer.EnableSsl = true;

smtpServer.Send(mail);

return true; // add return to satisfy method signature

}

}

}

**CustomerComm.cs**

namespace CustomerCommLib

{

public class CustomerComm

{

private readonly IMailSender \_mailSender;

public CustomerComm(IMailSender mailSender)

{

\_mailSender = mailSender;

}

public bool SendMailToCustomer()

{

// simulate business logic

string toAddress = "cust123@abc.com";

string message = "Some Message";

\_mailSender.SendMail(toAddress, message);

return true;

}

}

}

**Output**

