**NUnit - Hands\_on**

**Code :**

**Calculator.cs**

namespace CalcLibrary {

    public class Calculator

    {

        public int Div(int a, int b)

        {

            return a /b;

        }

       public int Mul(int a, int b){

    return a\*b;

} } }

**CalculatorTests.cs**

using NUnit.Framework;

using CalcLibrary;

namespace CalcLibrary.Tests {

    [TestFixture]

    public class CalculatorTests {

       private Calculator calc;

        [SetUp]

        public void Setup()

        {

            calc = new Calculator();

        }

        [TearDown]

        public void Teardown() {

            calc = null;

        }

        [Test]

        [TestCase(2, 3, 6)]

        [TestCase(0, 5, 0)]

        [TestCase(-2, 4, -8)]

        public void Multiply\_WhenCalled\_ReturnsExpectedResult(int a, int b, int expected)

        {

            int result = calc.Mul(a, b);

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

        }

        [Test]

        [TestCase(6, 3, 2)]

        [TestCase(10, 5, 2)]

        [TestCase(-8, 2, -4)]

        public void Divide\_WhenCalled\_ReturnsExpectedResult(int a, int b, int expected)

        {

            int result = calc.Div(a, b);

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

        }

        [Test]

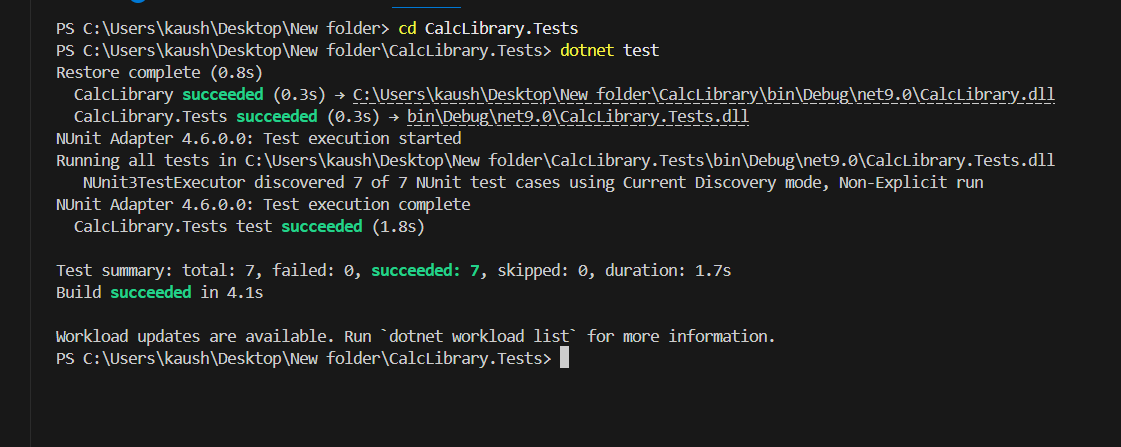
        public void Divide\_ByZero\_ThrowsDivideByZeroException()

        {

            Assert.Throws<System.DivideByZeroException>(() => calc.Div(5, 0));

        } } }

**Output:**



Moq-Handson

1.Write Testable Code with Moq

**Code Implementation:**

IMailSender.cs

namespace CustomerCommLib

{

    public interface IMailSender

    {

        bool SendMail(string toAddress, string message);

    }

}

MailSender.cs

using System.Net;

using System.Net.Mail;

namespace CustomerCommLib {

    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;

        } } }

CustomerComm.cs

namespace CustomerCommLib

{

    public class CustomerComm

    {

        IMailSender \_mailSender;

        public CustomerComm(IMailSender mailSender)

        {

            \_mailSender = mailSender;

        }

        public bool SendMailToCustomer()

        {

            return \_mailSender.SendMail("cust123@abc.com", "Some Message");

        } } }

1. Write Test in CustomerComm.Tests

CustomerCommTests.cs

using NUnit.Framework;

using Moq;

using CustomerCommLib;

namespace CustomerCommTests {

    [TestFixture]

    public class CustomerCommTests

    {

        private Mock<IMailSender> mockMailSender;

        private CustomerCommLib.CustomerComm customerComm;

        [OneTimeSetUp]

        public void Setup()

        {

            mockMailSender = new Mock<IMailSender>();

            mockMailSender

                .Setup(m => m.SendMail(It.IsAny<string>(), It.IsAny<string>()))

                .Returns(true);

customerComm=new CustomerCommLib.CustomerComm(mockMailSender.Object);

        }

        [Test]

        [TestCase("cust123@abc.com", "Some Message")]

        public void SendMailToCustomer\_ShouldReturnTrue(string email, string message); {

            bool result = customerComm.SendMailToCustomer();

            Assert.That(result, Is.True);

        } } }

Output:

