**1. NUnit-Handson**

**Code:**

using NUnit.Framework;

using CalcLibrary;

namespace CalcLibrary.Tests

{

[TestFixture]

public class CalculatorTests

{

private SimpleCalculator calc;

[SetUp]

public void Setup()

{

calc = new SimpleCalculator();

}

[TearDown]

public void Cleanup()

{

calc = null;

}

[Test]

public void Addition\_TwoPositiveNumbers\_ReturnsCorrectSum()

{

var result = calc.Addition(5, 3);

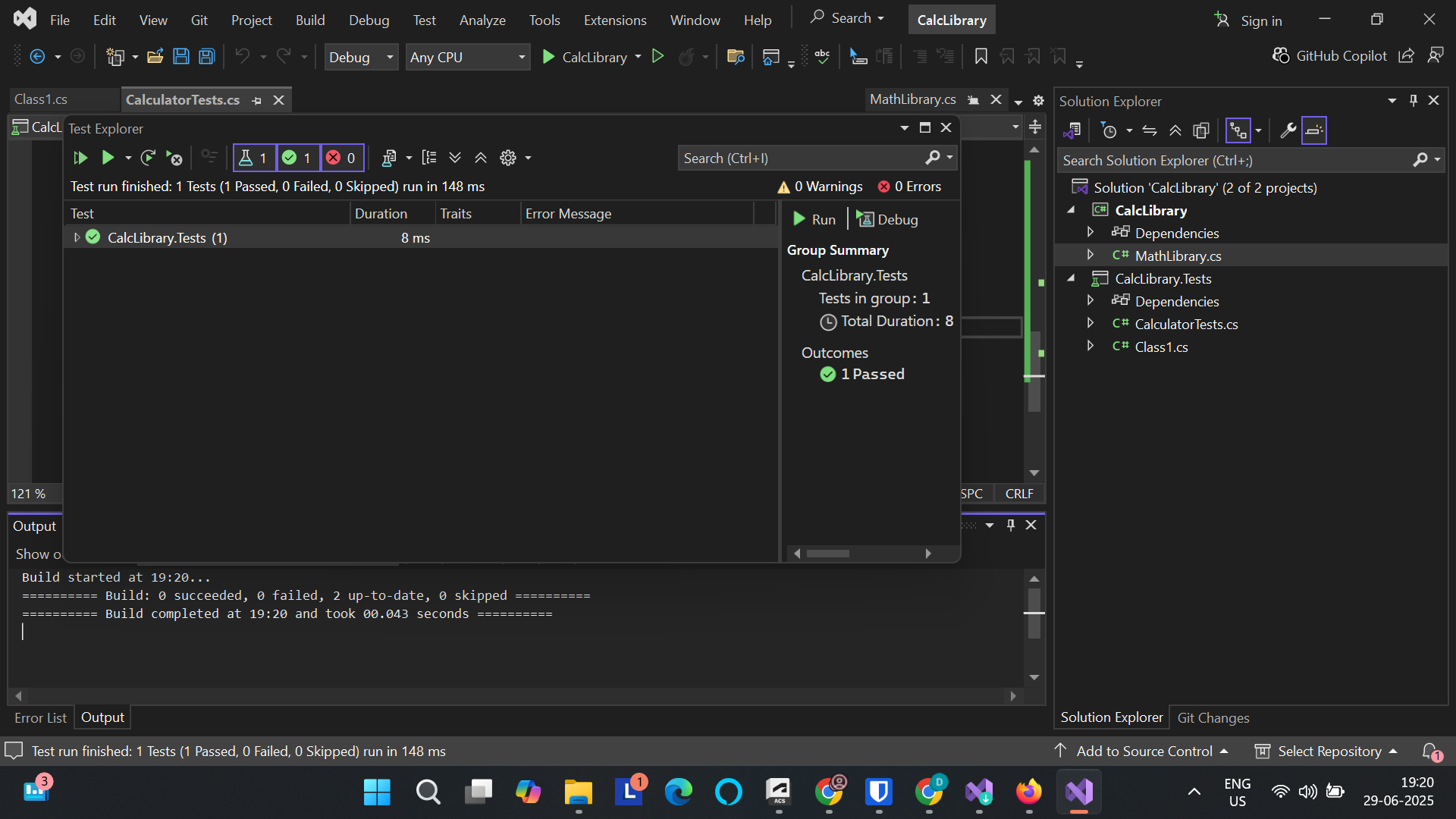
Assert.AreEqual(8, result);

}

}

}

**Output:**

****

**1. Moq-Handson - Write Testable Code with Moq**

**Code:**

**Creating the CustomerCommLib solution first:**

**File 1:** [**CustomerComm.cs**](http://customercomm.cs)

**namespace CustomerCommLib**

**{**

**public class CustomerComm**

**{**

**private readonly IMailSender \_mailSender;**

**public CustomerComm(IMailSender mailSender)**

**{**

**\_mailSender = mailSender;**

**}**

**public bool SendMailToCustomer()**

**{**

**string email = "cust123@abc.com";**

**string message = "Some Message";**

**return \_mailSender.SendMail(email, message);**

**}**

**}**

**}**

**File 2:**[**MailSender.cs**](http://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;**

**}**

**}**

**}**

**Creating the unit test project CustomerComm.Tests**

**File:** [**CustomerCommTests.cs**](http://customercommtests.cs)

**using NUnit.Framework;**

**using Moq;**

**using CustomerCommLib;**

**namespace CustomerComm.Tests**

**{**

**[TestFixture]**

**public class CustomerCommTests**

**{**

**private Mock<IMailSender> mockMailSender;**

**private CustomerCommLib.CustomerComm customerComm;**

**[OneTimeSetUp]**

**public void Init()**

**{**

**// Create a mock object for IMailSender**

**mockMailSender = new Mock<IMailSender>();**

**// Setup: return true no matter what strings are passed**

**mockMailSender**

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

**.Returns(true);**

**// Inject mock into class under test**

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

**}**

**[TestCase]**

**public void SendMailToCustomer\_WhenCalled\_ReturnsTrue()**

**{**

**// Act**

**bool result = customerComm.SendMailToCustomer();**

**// Assert**

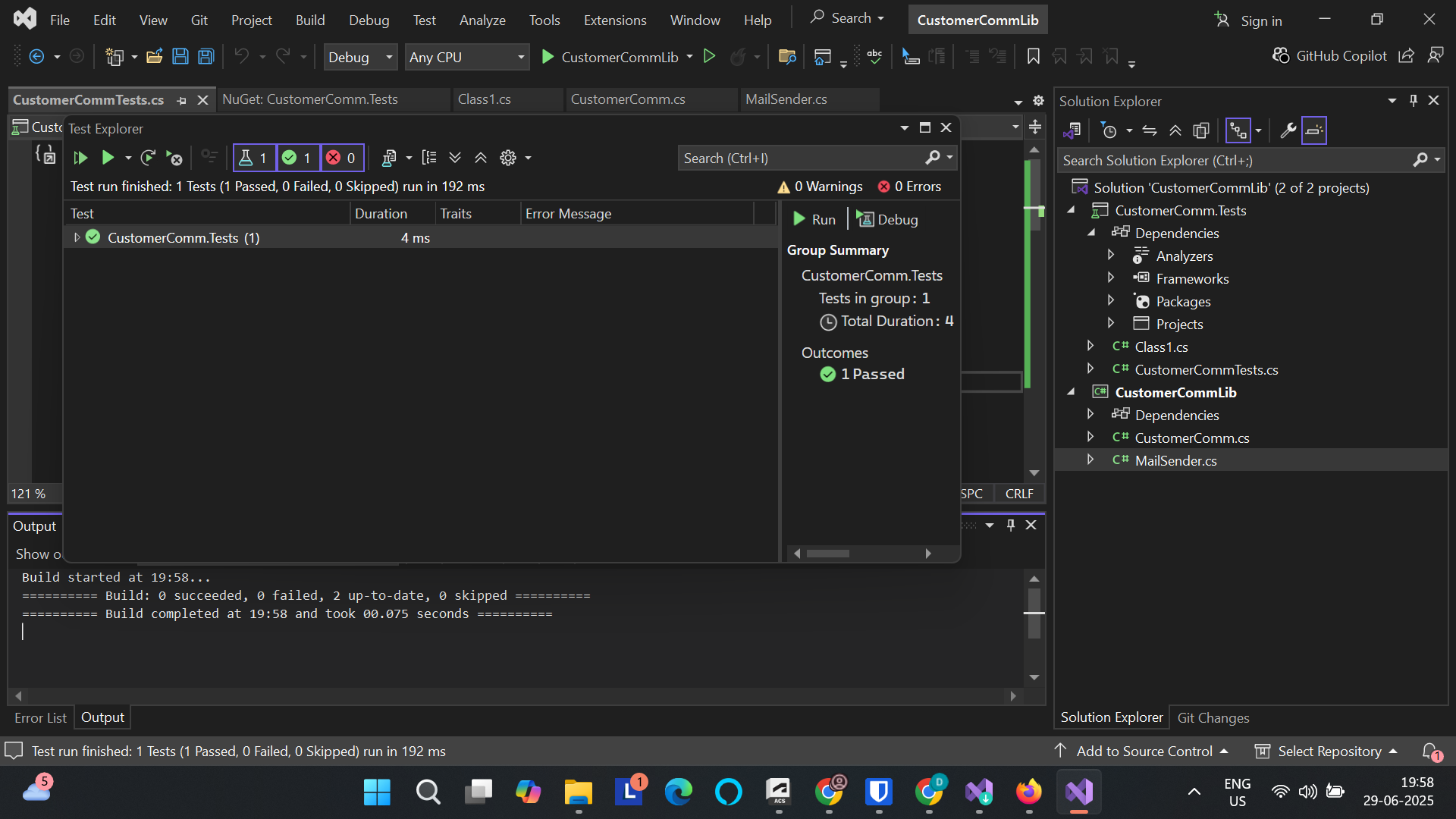
**Assert.That(result, Is.True);**

**}**

**}**

**}**

**Output:**

****