1. **Moq Handson:-**

### ****Part 1: Understanding the Basics****

### What is Unit Testing?

Think of your program like a toy robot 🧸. Unit testing is like checking each button of the robot to make sure they work properly before giving it to someone.

### What is Mocking?

Sometimes your robot might talk to a friend robot or go online to do something. During testing, you don’t want to:

Call the friend robot (send real email 📧)

Go online (call real database 🌐)

So, we create **pretend robots** (mock objects) that behave like the real ones — but are safe, fast, and controllable.

### What is Moq?

Moq is a tool that helps us create these **pretend robots** (mock objects) very easily in C#.

### What is TDD (Test Driven Development)?

First write tests like a checklist .  
Then build your toy robot (code) so it passes the checklist.

### What is Dependency Injection (DI)?

Imagine if your robot uses batteries . Instead of fixing the battery inside, we give it from outside. That’s called **injection**.

Similarly, if your code uses another class (like EmailSender), instead of making it inside, we give it from outside. This makes testing easier.

1. **MailSender.cs (in CustomerCommLib project)**

**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@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;**

**}**

**}**

**}**

1. **CustomerComm.cs (in CustomerCommLib project)**

**namespace CustomerCommLib**

**{**

**public class CustomerComm**

**{**

**private readonly IMailSender \_mailSender;**

**public CustomerComm(IMailSender mailSender)**

**{**

**\_mailSender = mailSender;**

**}**

**public bool SendMailToCustomer()**

**{**

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

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

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

**return true;**

**}**

**}**

**}**

1. **CustomerCommTests.cs (in CustomerComm.Tests test project)**

**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 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]**

**public void SendMailToCustomer\_ShouldReturnTrue()**

**{**

**bool result = \_customerComm.SendMailToCustomer();**

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

**}**

**}**

**}**

**Required NuGet Packages**

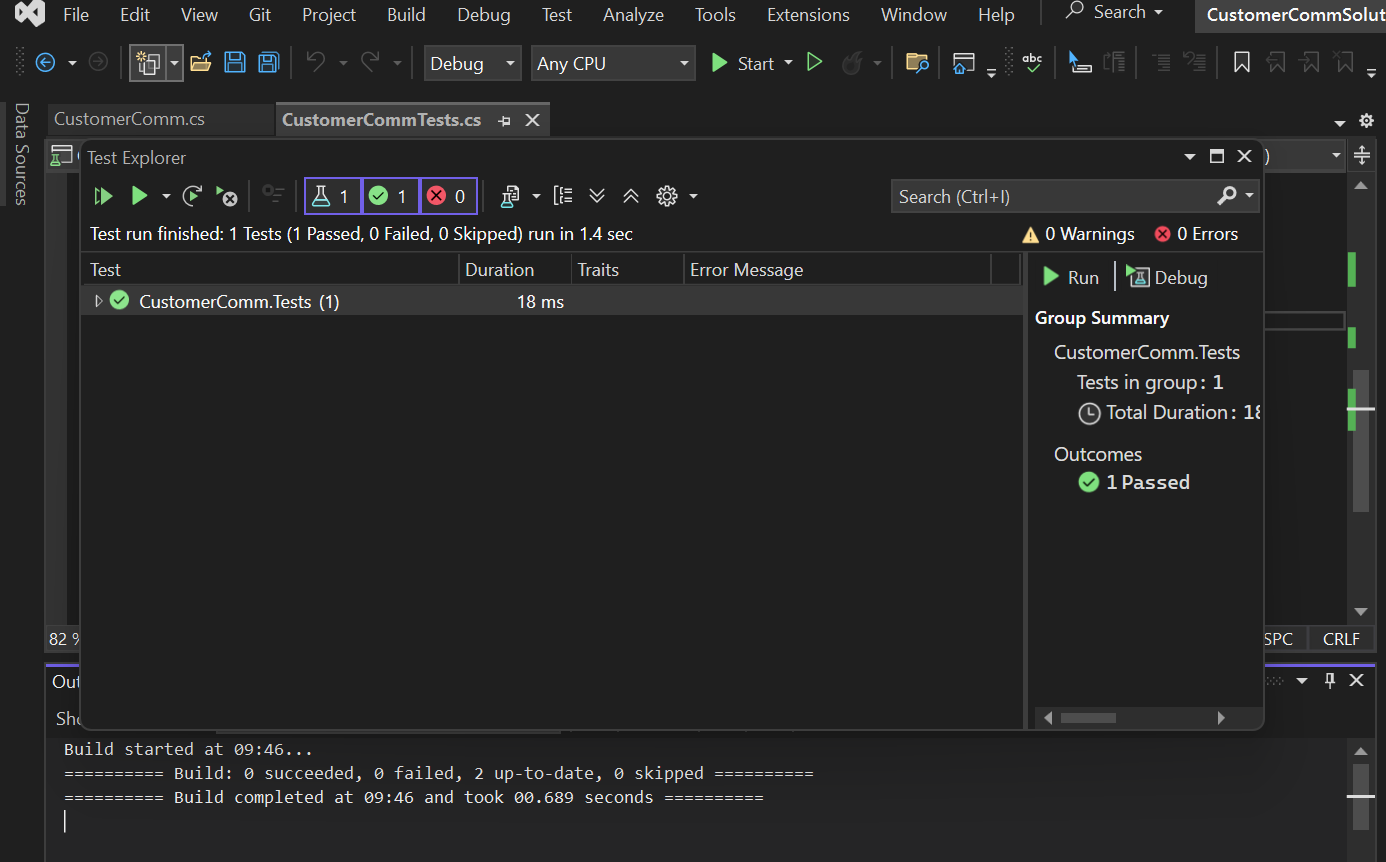
Install-Package NUnit

Install-Package NUnit3TestAdapter

Install-Package Microsoft.NET.Test.Sdk

Install-Package Moq

**OUTPUT:-**

****