**1. Moq -Handson**

**1.Write Testable Code with Moq**

**TASK 1:**

* **CODE:**

using System.Net;

using System.Net.Mail;

* MailSender.cs

namespace CustomerCommLib

{

public interface IMailSender

{

bool SendMail(string toAddress, string message);

}

}

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

{

private IMailSender \_mailSender;

public CustomerComm(IMailSender mailSender)

{

\_mailSender = mailSender;

}

public bool SendMailToCustomer()

{

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

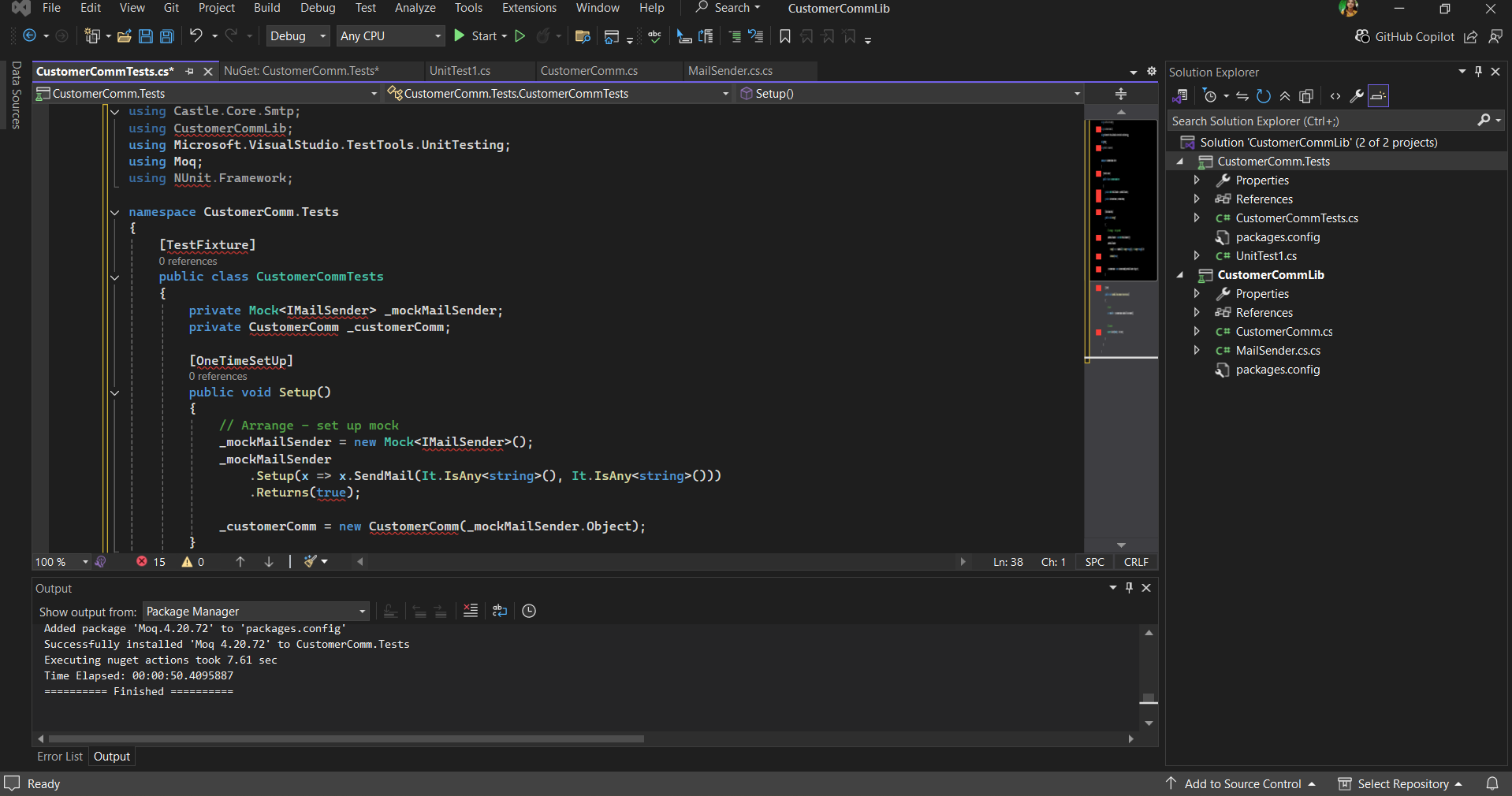
return true;

}

}

}

* **OUTPUT**:



**TASK 2:**

* **CODE:**

using NUnit.Framework;

using Moq;

using CustomerCommLib;

namespace CustomerComm.Tests

{

[TestFixture]

public class CustomerCommunicationTests

{

private Mock<IMailSender> \_mockMailSender;

private CustomerCommunication \_customerCommunication;

[OneTimeSetUp]

public void Init()

{

\_mockMailSender = new Mock<IMailSender>();

\_mockMailSender

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

.Returns(true);

\_customerCommunication = new CustomerCommunication(\_mockMailSender.Object);

}

[TestCase("user1@example.com", "Welcome!")]

[TestCase("user2@example.com", "Hi there!")]

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

{

// Act

bool result = \_customerCommunication.SendMailToCustomer(email, message);

// Assert

Assert.That(result, Is.True);

}

}

}

* **OUTPUT**:

