## **Task1**

**Code**

1. **IMailSender.cs**

using System;

namespace CustomerCommLib

{

public interface IMailSender

{

bool SendMail(string toAddress, string message);

}

}

1. **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("abc\_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;

}

}

}

1. **CustomerComm.cs**

namespace CustomerCommLib

{

public class CustomerComm

{

private readonly IMailSender \_mailSender;

public CustomerComm(IMailSender mailSender)

{

\_mailSender = mailSender;

}

public bool SendMailToCustomer()

{

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

return true;

}

}

}

## **Task2**

**CustomerCommTests.cs**

using NUnit.Framework;

using Moq;

using CustomerCommLib;

namespace CustomerComm.Tests

{

[TestFixture] // Marks this class as a test suite

public class CustomerCommTests

{

private Mock<IMailSender> \_mockMailSender;

private CustomerCommLib.CustomerComm \_customerComm; // Fully qualify to avoid namespace/type ambiguity

[OneTimeSetUp] // Runs once before all tests

public void Init()

{

\_mockMailSender = new Mock<IMailSender>();

// Configure the mock to return true for any two strings

\_mockMailSender

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

.Returns(true);

// Inject mock into the class under test

\_customerComm = new CustomerCommLib.CustomerComm(\_mockMailSender.Object);

}

[TestCase] // Runs this test case with default setup

public void SendMailToCustomer\_WhenCalled\_ReturnsTrue()

{

// Act

bool result = \_customerComm.SendMailToCustomer();

// Assert

Assert.That(result, Is.True);

}

}

}

**OUTPUT:**

