**Week 3**

Unit Testing: writing testable code with MOQ .

EXERCISE 1

Created a **Class Library (Language C#)** project using Visual Studio IDE, and named it as **CustomerCommLib.**

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("dipanjan1604@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;

}

}

}

Created another class named **CustomeComm.cs**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace CustomerCommLib

{

internal class CustomerComm

{

private IMailSender \_mailSender;

public CustomerComm(IMailSender mailSender)

{

\_mailSender = mailSender;

}

public bool SendMailToCustomer()

{

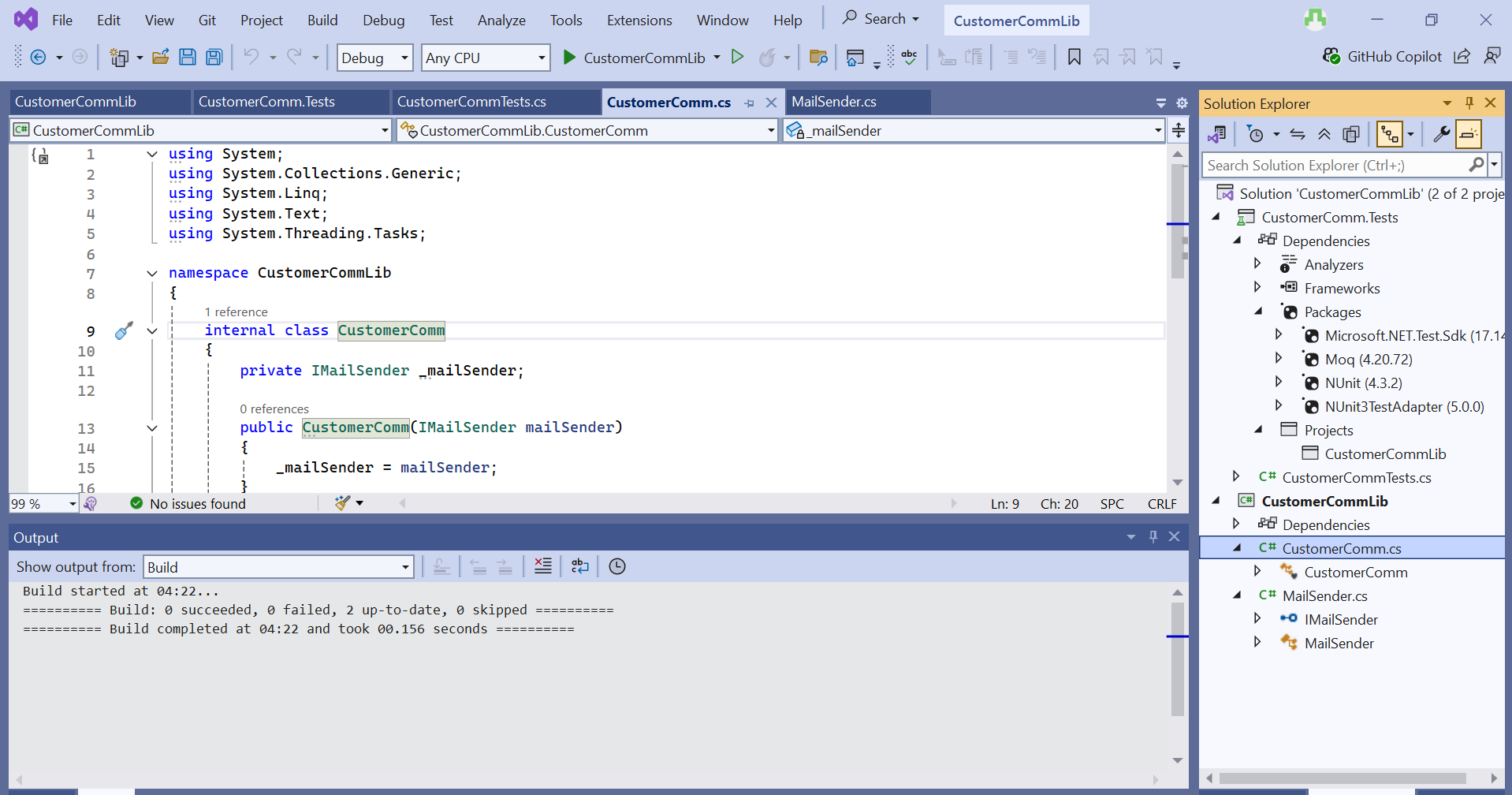
return \_mailSender.SendMail("dipanjan1604@gmail.com.com", "lets see");

}

}

**}**

**Code Implementation**



Created another test project within the solution and gave project reference to our project **CustomerCommLib.** Installed the framework **Nunit.Framework** , **Moq** and **NUnit3TestAdapter**.

using System;

using NUnit.Framework;

using Moq;

using CustomerCommLib;

namespace CustomerCommTests.Tests

{

public interface IMailSender

{

bool SendMail(string to, string body);

}

public class CustomerComm

{

private readonly IMailSender \_mailSender;

public CustomerComm(IMailSender mailSender)

{

\_mailSender = mailSender;

}

public bool SendMailToCustomer()

{

return \_mailSender.SendMail("customer@example.com", "Hello Customer");

}

}

[TestFixture]

public class CustomerCommTests

{

private Mock<IMailSender>? \_mockMailSender;

private CustomerComm? \_customerComm;

[OneTimeSetUp]

public void Setup()

{

\_mockMailSender = new Mock<IMailSender>();

\_mockMailSender

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

.Returns(true);

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

}

[Test]

public void SendMailToCustomer\_ShouldReturnTrue\_WhenMailIsSentSuccessfully()

{

Assert.That(\_customerComm, Is.Not.Null);

var result = \_customerComm!.SendMailToCustomer();

Assert.That(result, Is.Not.Null);

}

}

}

**Code Implementation and test case passing**

