Internet Application Development

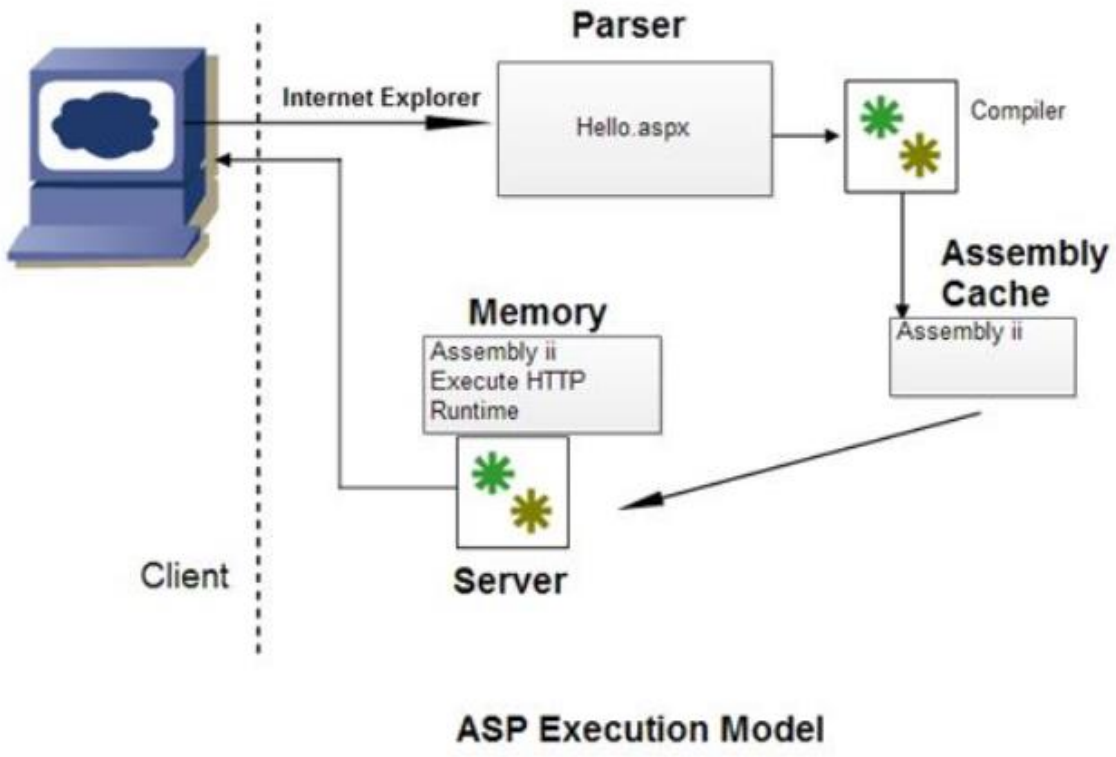
Web Services - Assignment

Dated: Thursday, 15th May 2025

Registration No: 03-3-1-058-2022

Full Name: Zainab Tariq

Q1) Draw web services execution model?



ASP Execution Model

Q2) Develop a web service with four web methods as follows:

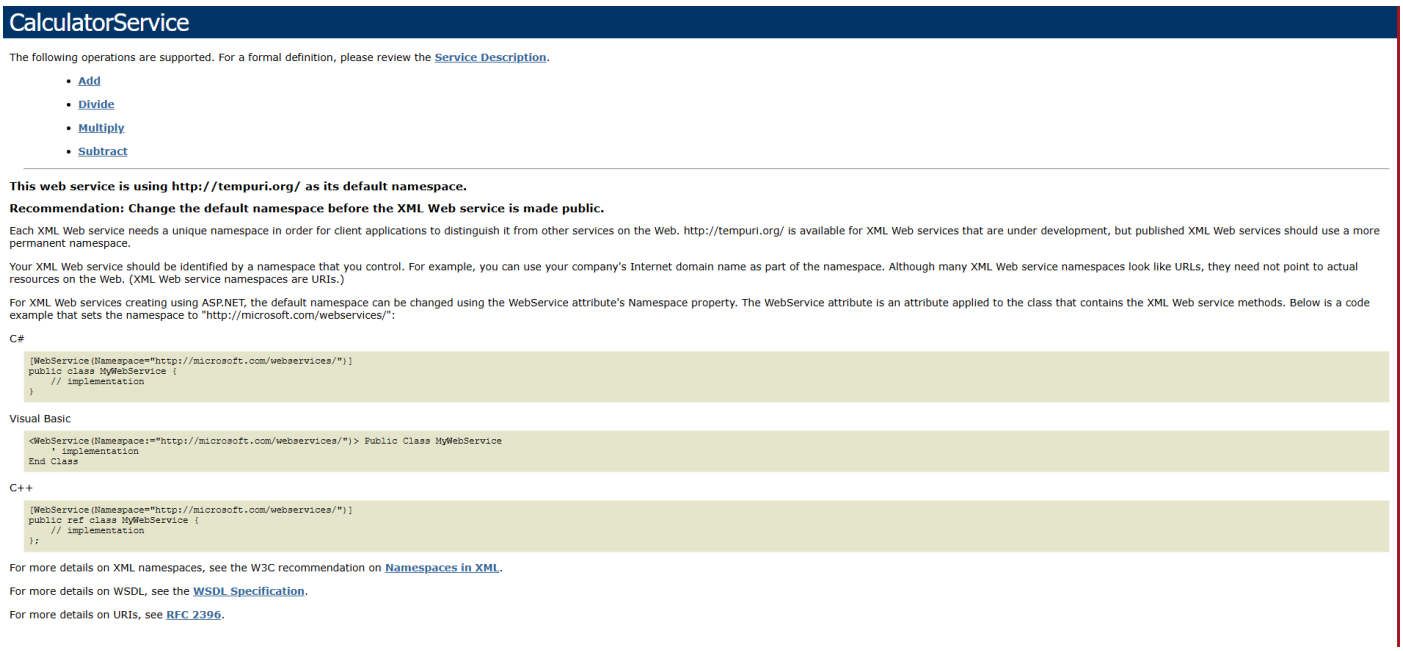
(a) Add
(b) Subtract
(c) Multiply
(d) Divide

Assume all above methods need two parameters and return a single value as string value.

CODE

```
<%@ WebService Language="VB" CodeBehind="CalculatorService.asmx.vb"
Class="CalculatorWebServiceVB.CalculatorService" %>


Imports System.Web.Services
Imports System.Web.Services.Protocols
Imports System.ComponentModel

Namespace CalculatorWebServiceVB
```

```
    <WebService(Namespace:="http://tempuri.org/")>
    <WebServiceBinding(ConformsTo:=WsiProfiles.BasicProfile1_1)>
    <ToolboxItem(False)>
    Public Class CalculatorService
        Inherits System.Web.Services.WebService

        <WebMethod()>
        Public Function Add(a As Double, b As Double) As Double
            Return a + b
        End Function

        <WebMethod()>
        Public Function Subtract(a As Double, b As Double) As Double
            Return a - b
        End Function

        <WebMethod()>
        Public Function Multiply(a As Double, b As Double) As Double
            Return a * b
        End Function

        <WebMethod()>
        Public Function Divide(a As Double, b As Double) As String
            If b = 0 Then
                Return "Error: Cannot divide by zero."
            End If
            Return (a / b).ToString()
        End Function

    End Class

End Namespace
```



CalculatorService

The following operations are supported. For a formal definition, please review the Service Description.

- Add
- Divide
- Multiply
- Subtract

This web service is using http://tempuri.org/ as its default namespace.

Recommendation: Change the default namespace before the XML Web service is made public.

Each XML Web service needs a unique namespace in order for client applications to distinguish it from other services on the Web. http://tempuri.org/ is available for XML Web services that are under development, but published XML Web services should use a more permanent namespace.

Your XML Web service should be identified by a namespace that you control. For example, you can use your company's Internet domain name as part of the namespace. Although many XML Web service namespaces look like URLs, they need not point to actual resources on the Web. (XML Web service namespaces are URIs.)

For XML Web services creating using ASP.NET, the default namespace can be changed using the WebService attribute's Namespace property. The WebService attribute is an attribute applied to the class that contains the XML Web service methods. Below is a code example that sets the namespace to "http://microsoft.com/webservices/":

C#

```
[WebService(Namespace="http://microsoft.com/webservices/")]
public class MyWebService {
    // implementation
}
```

Visual Basic

```
<WebService(Namespace:="http://microsoft.com/webservices/")> Public Class MyWebService
    ' implementation
End Class
```

C++

```
[WebService(Namespace="http://microsoft.com/webservices/")]
public ref class MyWebService {
    // implementation
};
```

For more details on XML namespaces, see the W3C recommendation on Namespaces in XML.

For more details on WSDL, see the WSDL Specification.

For more details on URIs, see RFC 2396.

Using guidelines provided in class write code for above web service.

References (Previous version of book): PART 5 ■ ■ ■ **Web Services**

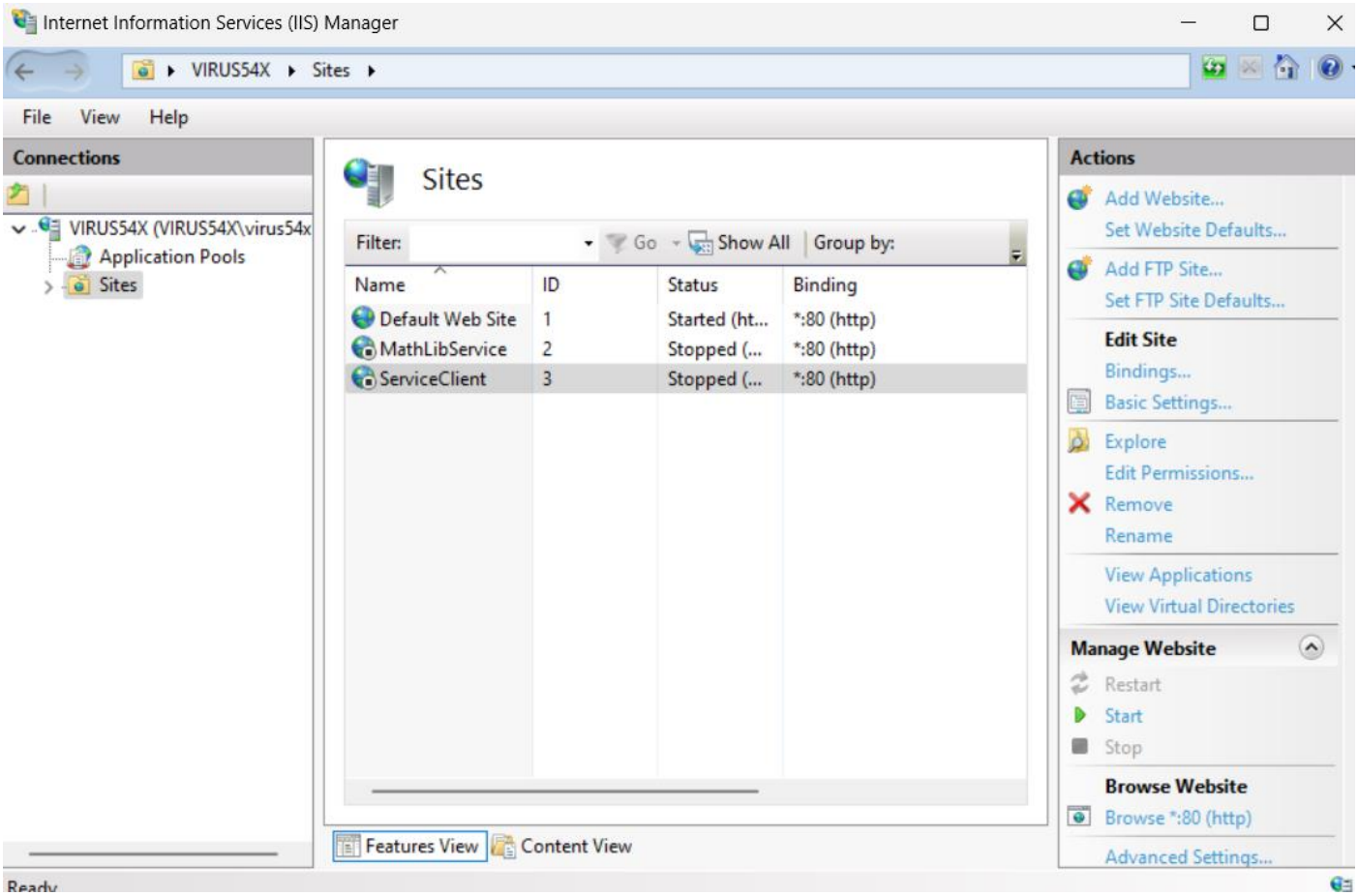
Q3) Test the web service using browser on local pc.

Install Internet Information Services on your laptop/pc (if not installed)

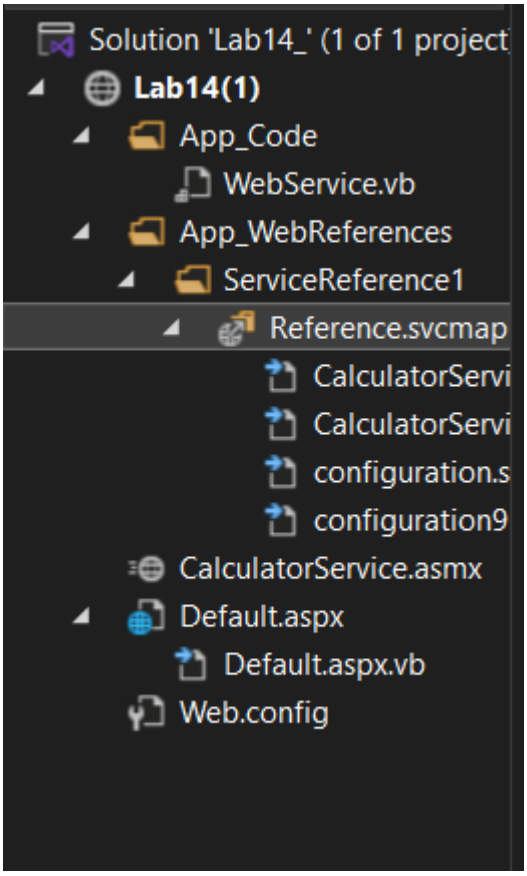
Using IIS manager create two separate applications as follows:

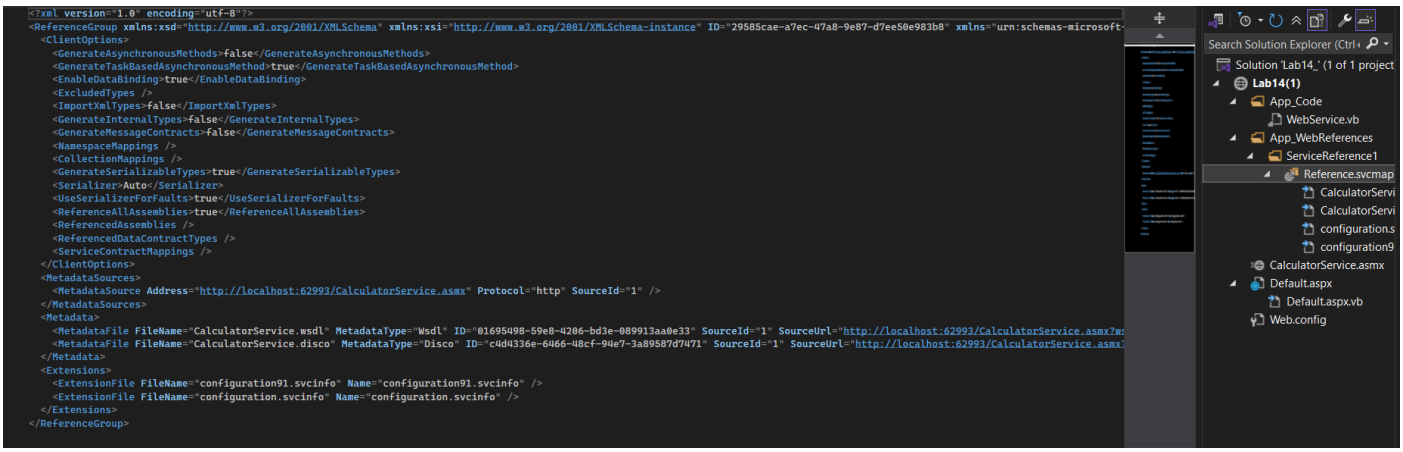
(a) MathLibService

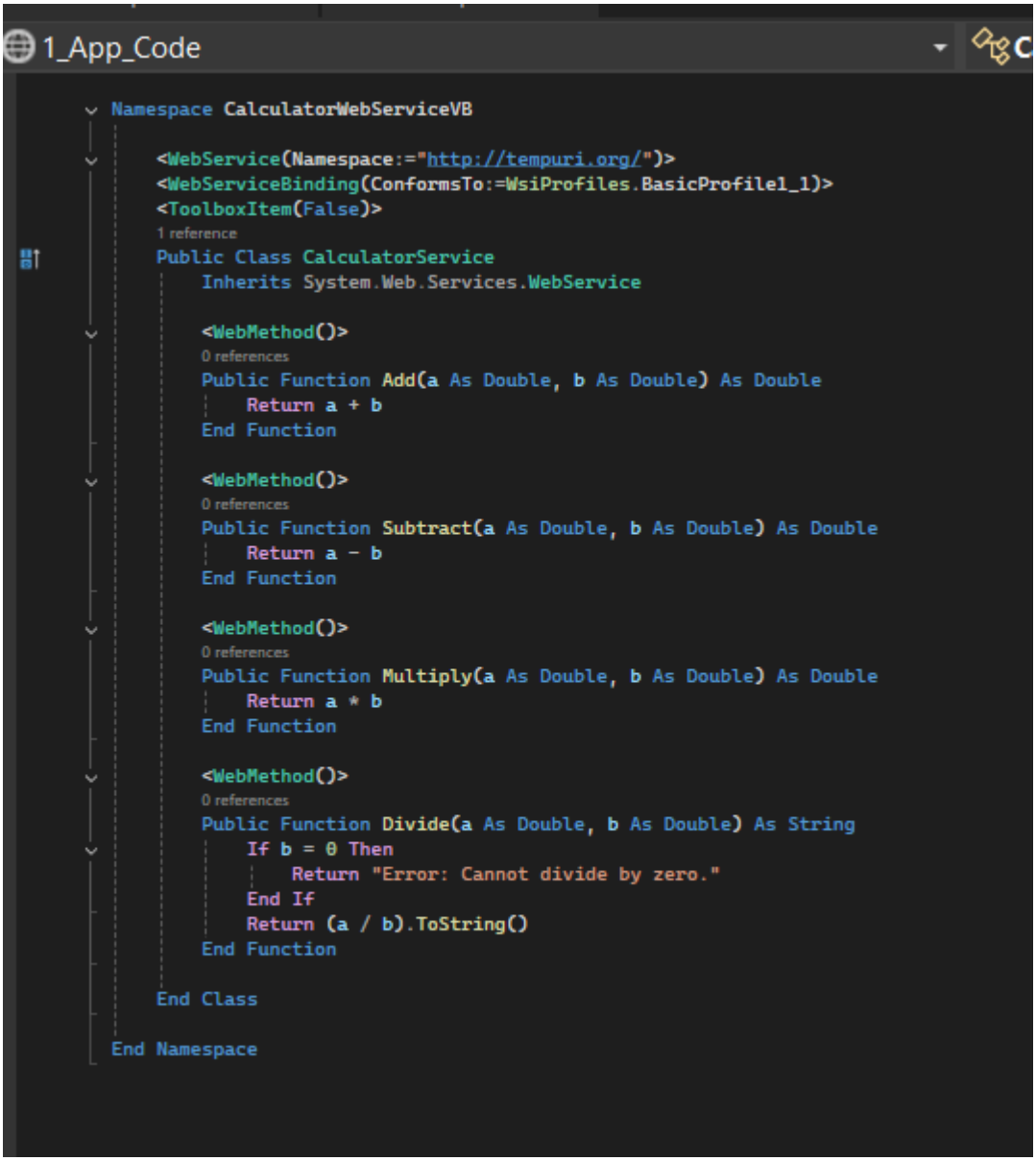
(b) ServiceClient



Q4) Implement a web application “ServiceClient” which will use the MathLibService as a web service. Develop a suitable web page for its demonstration. Use some text boxes to take input from user. Use proxy class to get answer (response from web service) and display it on page to user.


Solution 'Lab14_' (1 of 1 project
Lab14(1)
App_Code
WebService.vb
App_WebReferences
ServiceReference1
Reference.svcmap
CalculatorServi
CalculatorServi
configuration.s
configuration9
CalculatorService.asmx
Default.aspx
Default.aspx.vb
Web.config


<?xml version="1.0" encoding="utf-8"?>
<ReferenceGroup xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" ID="29585cae-a7ec-47a8-9e87-d7ee50e983b8" xmlns="urn:schemas-microsoft-
<ClientOptions>
<GenerateAsynchronousMethods>false</GenerateAsynchronousMethods>
<GenerateTaskBasedAsynchronousMethod>true</GenerateTaskBasedAsynchronousMethod>
<EnableDataBinding>true</EnableDataBinding>
<ExcludedTypes />
<ImportXmlTypes>false</ImportXmlTypes>
<GenerateInternalTypes>false</GenerateInternalTypes>
<GenerateMessageContracts>false</GenerateMessageContracts>
<NamespaceMappings />
<CollectionMappings />
<GenerateSerializableTypes>true</GenerateSerializableTypes>
<Serializer>Auto</Serializer>
<UseSerializerForFaults>true</UseSerializerForFaults>
<ReferenceAllAssemblies>true</ReferenceAllAssemblies>
<ReferencedAssemblies />
<ReferencedDataContractTypes />
<ServiceContractMappings />
</ClientOptions>
<MetadataSources>
<MetadataSource Address="http://localhost:62993/CalculatorService.asmx" Protocol="http" SourceId="1" />
</MetadataSources>
<Metadata>
<MetadataFile FileName="CalculatorService.wsdl" MetadataType="Wsdl" ID="01695498-59e8-4206-bd3e-089913aa0e33" SourceId="1" SourceUrl="http://localhost:62993/CalculatorService.asmx?ws
<MetadataFile FileName="CalculatorService.disco" MetadataType="Disco" ID="c4d4336e-6466-48cf-94e7-3a89587d7471" SourceId="1" SourceUrl="http://localhost:62993/CalculatorService.asmx
</Metadata>
<Extensions>
<ExtensionFile FileName="configuration91.svcinfo" Name="configuration91.svcinfo" />
<ExtensionFile FileName="configuration.svcinfo" Name="configuration.svcinfo" />
</Extensions>
</ReferenceGroup>
Search Solution Explorer (Ctrl
Solution 'Lab14_' (1 of 1 project
Lab14(1)
App_Code
WebService.vb
App_WebReferences
ServiceReference1
Reference.svcmap
CalculatorServi
CalculatorServi
configuration.s
configuration9
CalculatorService.asmx
Default.aspx
Default.aspx.vb
Web.config



CalculatorService

The following operations are supported. For a formal definition, please review the **Service Description**.

- **Add**
- **Divide**
- **Multiply**
- **Subtract**

This web service is using http://tempuri.org/ as its default namespace.

Recommendation: Change the default namespace before the XML Web service is made public.

Each XML Web service needs a unique namespace in order for client applications to distinguish it from other services on the Web. http://tempuri.org/ is available for XML Web services that are under development, but published XML Web services should use a more permanent namespace.

Your XML Web service should be identified by a namespace that you control. For example, you can use your company's Internet domain name as part of the namespace. Although many XML Web service namespaces look like URLs, they need not point to actual resources on the Web. (XML Web service namespaces are URIs.)

For XML Web services creating using ASP.NET, the default namespace can be changed using the WebService attribute's Namespace property. The WebService attribute is an attribute applied to the class that contains the XML Web service methods. Below is a code example that sets the namespace to "http://microsoft.com/webservices/":

C#

```
[WebService(Namespace="http://microsoft.com/webservices/")]
public class MyWebService {
    // implementation
}
```

Visual Basic

```
<WebService(Namespace:="http://microsoft.com/webservices/")> Public Class MyWebService
    ' implementation
End Class
```

C++

```
[WebService(Namespace="http://microsoft.com/webservices/")]
public ref class MyWebService {
    // implementation
};
```

For more details on XML namespaces, see the W3C recommendation on **Namespaces in XML**.

For more details on WSDL, see the **WSDL Specification**.

For more details on URIs, see **RFC 2396**.

CalculatorService

Click here for a complete list of operations.

Add

Test

To test the operation using the HTTP POST protocol, click the 'Invoke' button.

Parameter	Value
a:	4
b:	5

Invoke

SOAP 1.1

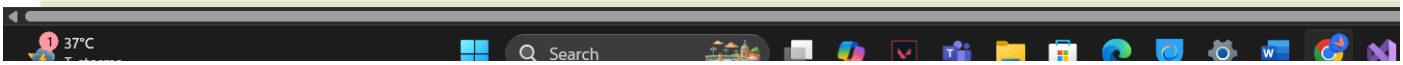
The following is a sample SOAP 1.1 request and response. The placeholders shown need to be replaced with actual values.

```
POST /CalculatorService.asmx HTTP/1.1
Host: localhost
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://tempuri.org/Add"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <Add xmlns="http://tempuri.org/">
      <a>double</a>
      <b>double</b>
    </Add>
  </soap:Body>
</soap:Envelope>
```

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <AddResponse xmlns="http://tempuri.org/">
      <AddResult>double</AddResult>
    </AddResponse>
  </soap:Body>
</soap:Envelope>
```

SOAP 1.2

The following is a sample SOAP 1.2 request and response. The placeholders shown need to be replaced with actual values.



CalculatorService

Click here for a complete list of operations.

Multiply

Test

To test the operation using the HTTP POST protocol, click the 'Invoke' button.

Parameter	Value
a:	
b:	

Invoke

SOAP 1.1

The following is a sample SOAP 1.1 request and response. The placeholders shown need to be replaced with actual values.

```
POST /CalculatorService.asmx HTTP/1.1
Host: localhost
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://tempuri.org/Multiply"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <Multiply xmlns="http://tempuri.org/">
      <a>double</a>
      <b>double</b>
    </Multiply>
  </soap:Body>
</soap:Envelope>
```

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <MultiplyResponse xmlns="http://tempuri.org/">
      <MultiplyResult>double</MultiplyResult>
    </MultiplyResponse>
  </soap:Body>
</soap:Envelope>
```

SOAP 1.2

CalculatorService

Click here for a complete list of operations.

Subtract

Test

To test the operation using the HTTP POST protocol, click the 'Invoke' button.

Parameter	Value
a:	
b:	

Invoke

SOAP 1.1

The following is a sample SOAP 1.1 request and response. The placeholders shown need to be replaced with actual values.

```
POST /CalculatorService.asmx HTTP/1.1
Host: localhost
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://tempuri.org/Subtract"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <Subtract xmlns="http://tempuri.org/">
      <a>double</a>
      <b>double</b>
    </Subtract>
  </soap:Body>
</soap:Envelope>
```

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <SubtractResponse xmlns="http://tempuri.org/">
      <SubtractResult>double</SubtractResult>
    </SubtractResponse>
  </soap:Body>
</soap:Envelope>
```

SOAP 1.2

CalculatorService

Click here for a complete list of operations.

Divide

Test

To test the operation using the HTTP POST protocol, click the 'Invoke' button.

Parameter	Value
a:	
b:	

Invoke

SOAP 1.1

The following is a sample SOAP 1.1 request and response. The placeholders shown need to be replaced with actual values.

```
POST /CalculatorService.asmx HTTP/1.1
Host: localhost
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://tempuri.org/Divide"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soap="http://schemas.xmlsoap.org/s
  <soap:Body>
    <Divide xmlns="http://tempuri.org/">
      <a>double</a>
      <b>double</b>
    </Divide>
  </soap:Body>
</soap:Envelope>
```

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soap="http://schemas.xmlsoap.org/s
  <soap:Body>
    <DivideResponse xmlns="http://tempuri.org/">
      <DivideResult>string</DivideResult>
    </DivideResponse>
  </soap:Body>
</soap:Envelope>
```

Note:

(i) This is an individual student assignment.

(ii) Submission of copied work (by any means/through any channel) will lead to poor grades.

(iii) Deadline is 22:00 on 18th May 2025. Submit a report (pdf) including above solved problems on your own portal.