SYSTINT Project VT2025

Task 3 – Develop a service/process integration with a web service

Table of Contents

Project Overview

This task contains:

- A hands-on exercise on developing simple web services
- Publishing the web service on a web server (your VM will host the web service as a web server)
- Develop a Windows Application that consumes the developed web service.

The Goals of this task

The second part of the project covers how web services are developed and consumed in applications. You will learn how to create web services using C#, set up IIS, and develop a desktop application that will use the web service developed. Thus, the goals are:

- To familiarize you with the components of web services.
- To be able to create web services and register these services in IIS.
- To demonstrate language neutrality of web services by developing client applications that consume these services in Windows applications. It is also possible to develop clients using Java and PHP.

Learning outcomes

After doing this exercise, students will learn how to develop simple SOAP-based web services and register these services in IIS. Students will also learn how to develop windows based applications using C# and understand how client applications consume or use web services.

Good Luck!

Creating and Consuming Web Services

In this section, you will learn how web services could be developed and provisioned by web servers. Also, you will learn how these services are consumed in applications. You will set up your virtual environment as a web server and consume the web service hosted by this web server in your application using the **localhost** URL. The web service you will create is a simple service that will provide four services, and these services are simple arithmetic operations: **+**, **-**, **X**, and **÷**.

Web Services

WCF (Windows Communication Foundation) is suitable for all service development needs in the .NET Framework developed.

Installing IIS and Other Required Components on Windows Server 2016

In this task, you will create a local web server, create a simple web service, and develop a Windows application to consume the web service developed. To create the web service, you need to install and configure Internet Information Services, IIS.

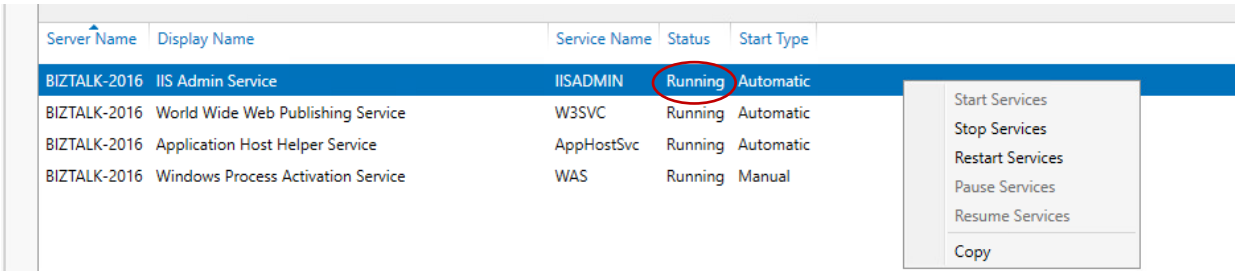
IIS is a Web server, which is a secure, flexible, and manageable Web server for hosting anything on the Web-the services which could be hosted range from media streaming to web applications. IIS's scalable and open architecture is ready to handle the most demanding tasks.

Note:

Your virtual environments are ready. IIS is already installed and configured, so you hardly need to change any configurations. (If you want to configure an ISS on your private Windows 2016 Server, then you can refer to **Appendix 1**. If you want to install and configure IIS on Windows 7, 8, 10, or any other Windows server, you can refer to Microsoft Developer forums).

Make sure that IIS Service is running,

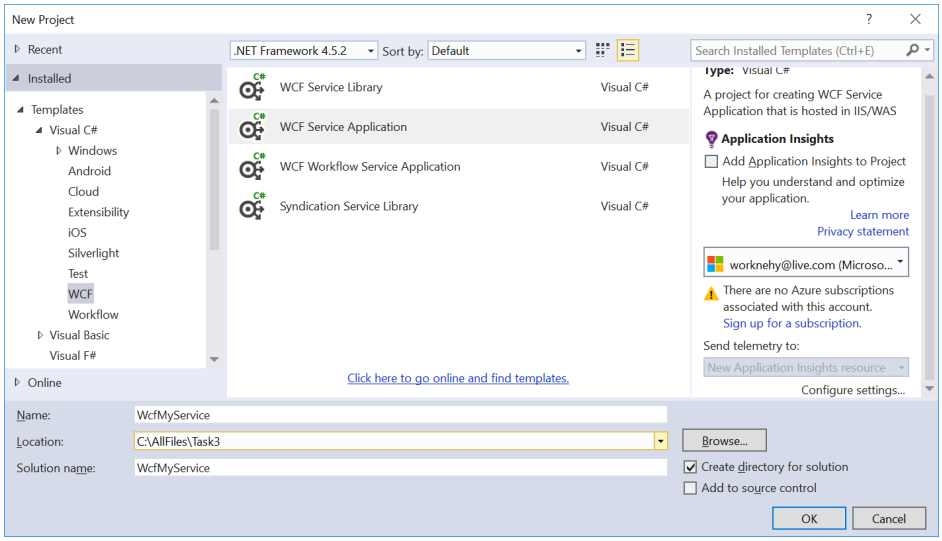
- From the **Start** menu select **Server Manager**, and from **Service Manager** select **IIS** on the left panel
- Make sure IIS service is running, if it is not running as illustrated below Right click on **IISAdmin Service** then click on **Start**



Creating the ASP.NET WCF Web Service

Steps

- Start **Visual Studio**, select **New Project…** from **Start Page**
- Under **Templates,** Expand **Visual C#** then Select **WCF**
- Chose **WCF Service Application**
- Specify folder location as **C:\AllFiles\Task3**
- Enter – **WcfMyService** under both **Name** and **Solution name**, then click on **OK**

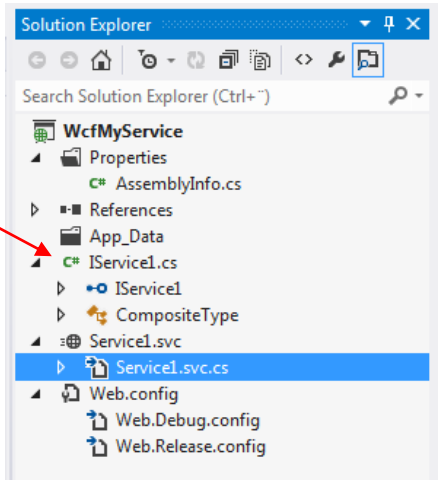


Define interfaces (contracts)

- Go to the **Solution Explorer** on the right side

- Then open the **IService1.cs** by double clicking on it to add the following code,
- Add the code just below "`// TODO: Add your service operations here`" as illustrated below

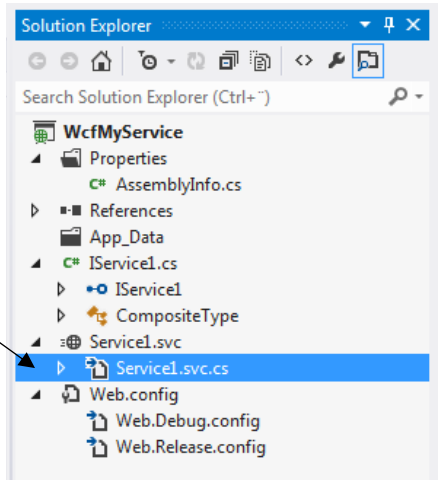
```
// TODO: Add your service operations here

[OperationContract]
double Add(double x, double y);

[OperationContract]
double Subtract(double x, double y);

[OperationContract]
double Multiply(double x, double y);

[OperationContract]
double Divide (double x, double y);
```



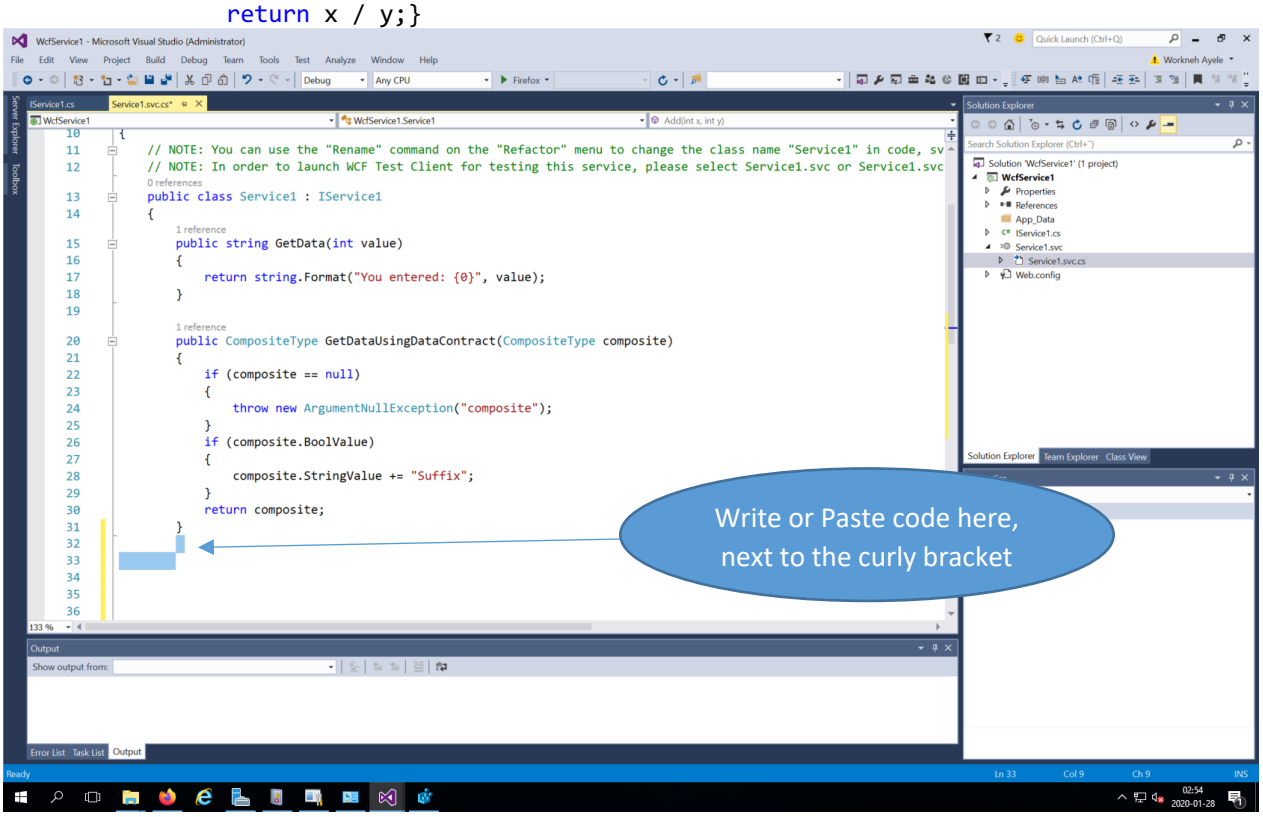
`[OperationContract]`– this indicates or defines methods that are parts of the service contract, the interfaces are the service contracts.

Define methods for the interfaces (for the contracts)

- Select or double click on **Service1.svc.cs** under **Service1.svc** Then add the following implementation code.
- Copy the following code and paste it in Service1.csv.cs code section indicated in the figure below with an arrow.

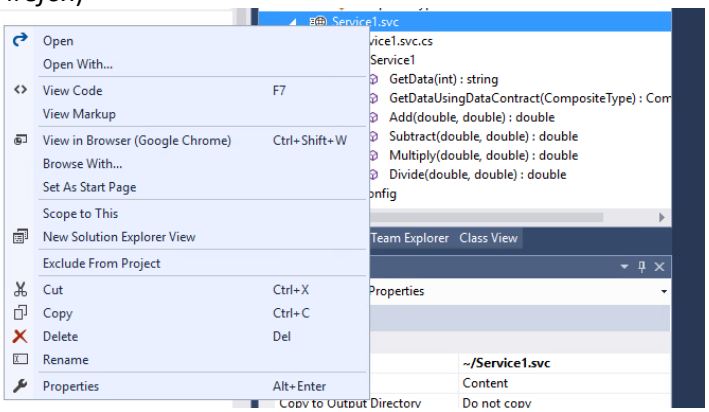



```
public double Add(double x, double y)
{return x + y;}

public double Subtract(double x, double y)
{return x - y;}

public double Multiply(double x, double y)
{return x * y;}

public double Divide (double x, double y)
{if (y == 0)
        return -1;
    else
        return x / y;}
```

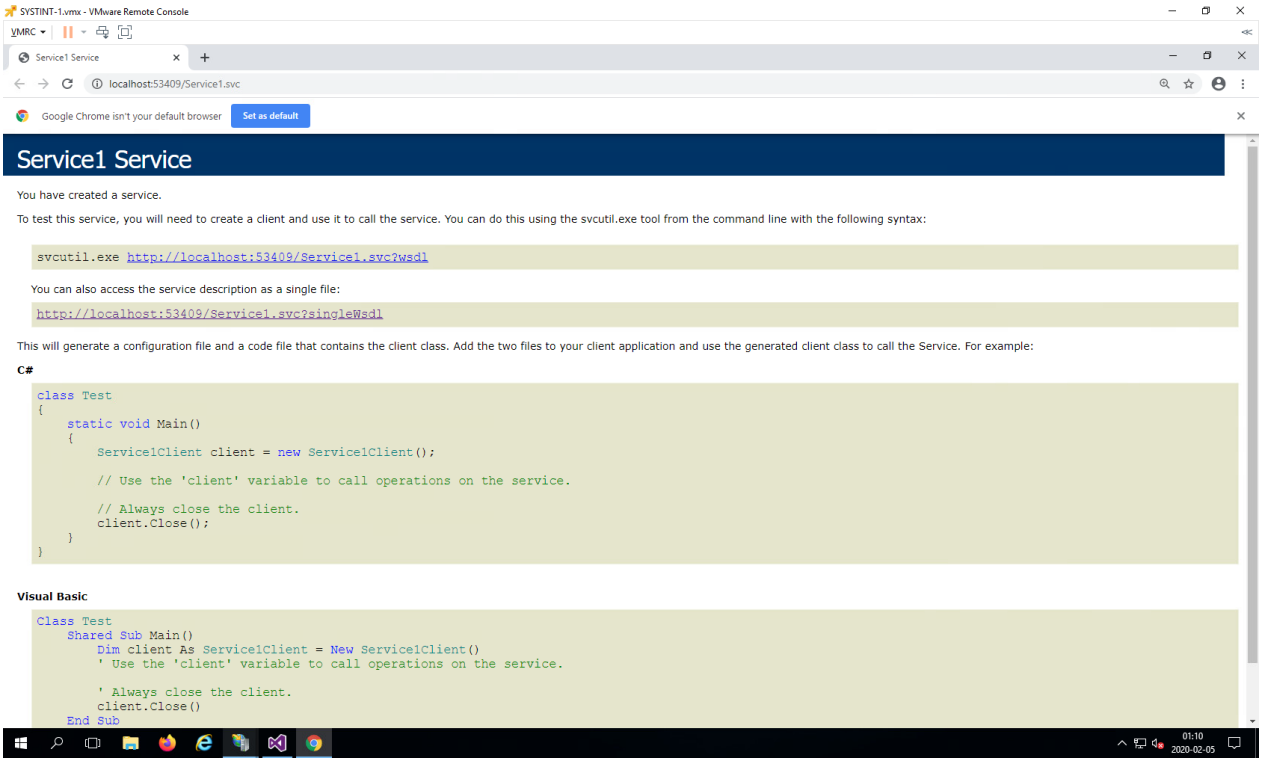


Testing if the web service is

- Right click on **Service1.svc**
- Select **View in Browser (Google Chrome)** (***Note**: The browser shown here could be Internet Explorer or Firefox*)



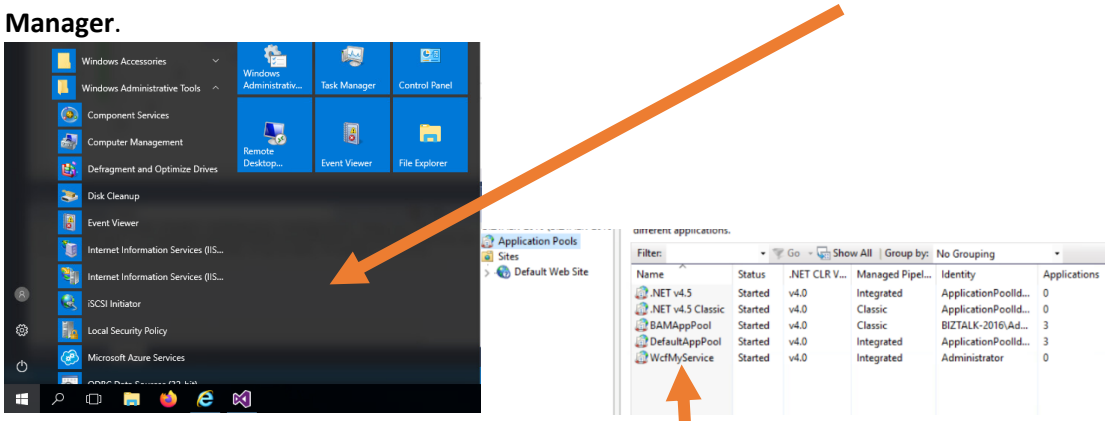
- You will see a confirmation that the service has been created as illustrated below.



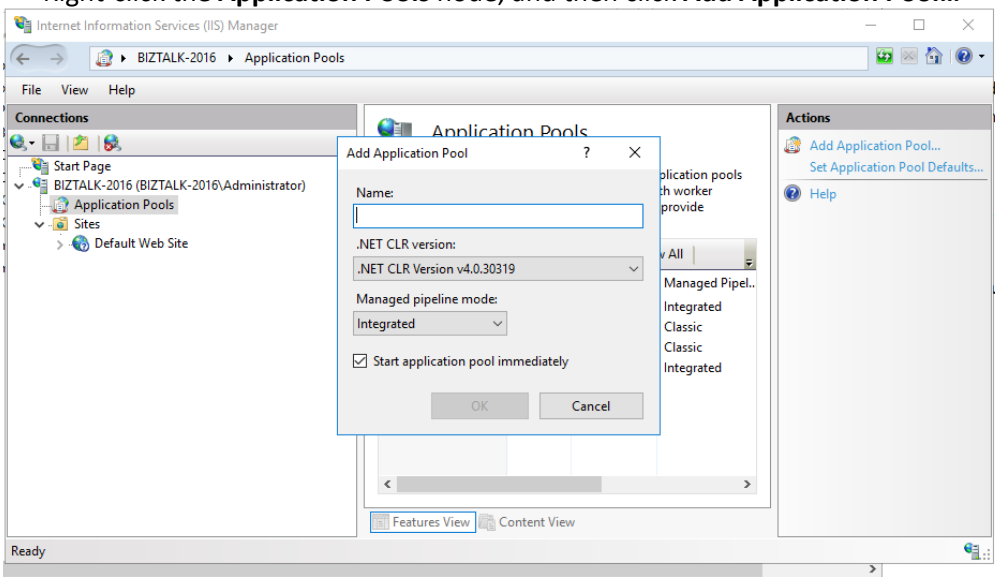
- (If you want you can examine the content of the WSDL file)
- **Build** the solution

Configuring IIS to host the generated web services

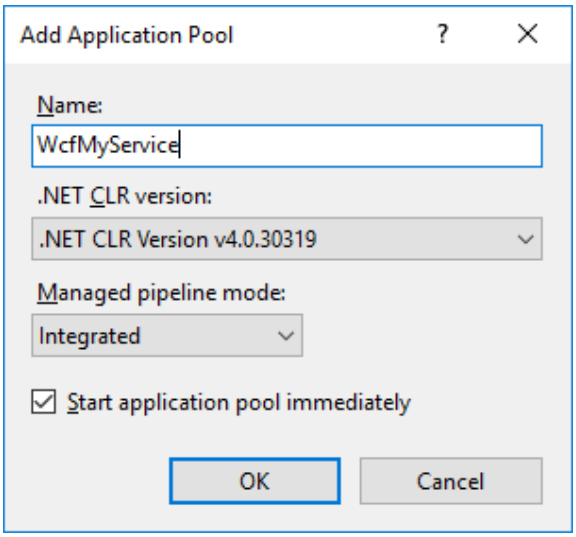
- On the **Start** menu, click **Administrative Tools**, and then click **Internet Information Services (IIS) Manager**.



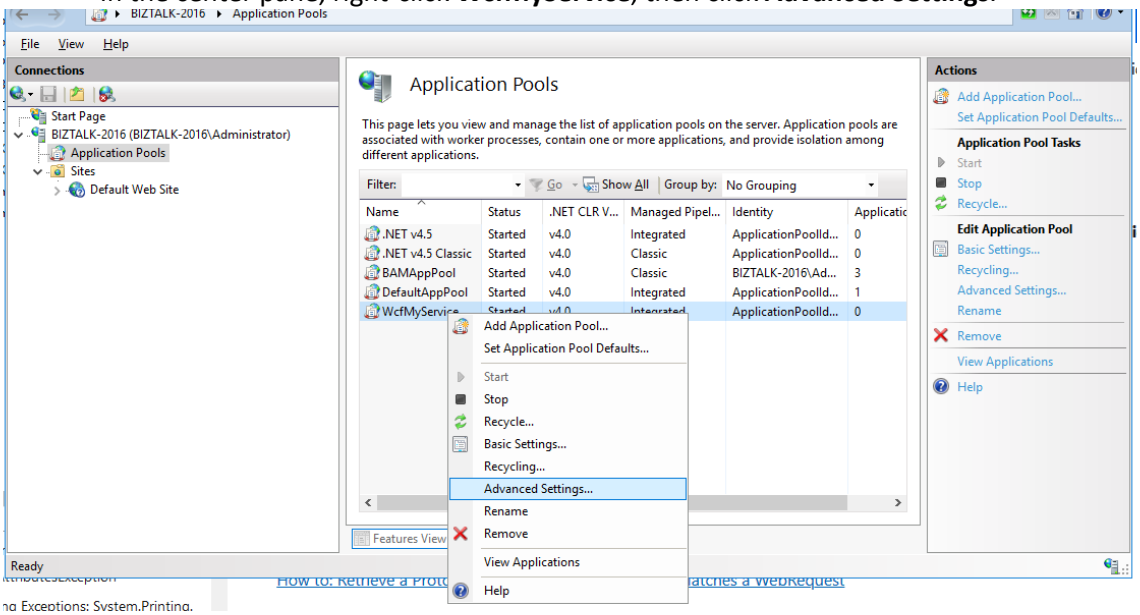
- In the **Internet Information Services (IIS) Manager** window, in the left pane, expand the **BIZTALK-2016** node, and then click **Application Pools**.
- Under **Application Pools** you will find **WcfMyService**, **Delete (Remove) WcfMyService, DO NOT DELETE ANYTHING ELSE!**
- Right-click the **Application Pools** node, and then click **Add Application Pool…**



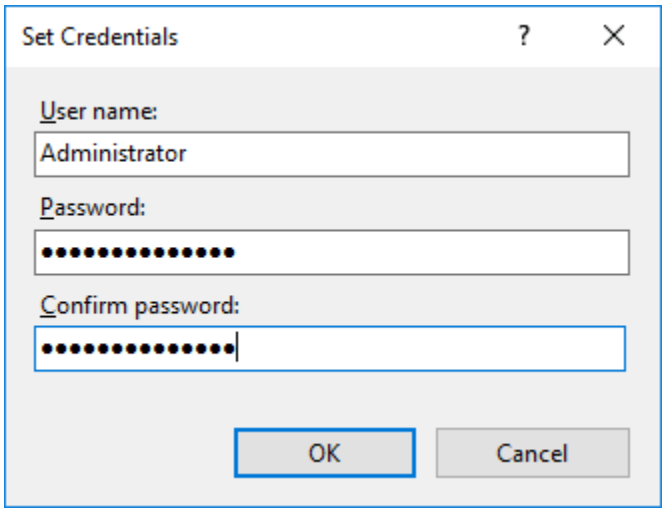
- In the **Add Application Pool** dialog box, in the **Name** box, enter **WcfMyService**.



- In the **.NET Framework version** list, select **.NET Framework v4.0.30319**, then click **OK**.
- In the center pane, right-click **WcfMyService**, then click **Advanced Settings**.



- In the **Advanced Settings** dialog box, click **Identity**, then click the **ellipses (…)** button.
- Check the **Custom Account** radio button and then click on the **Set …** button.

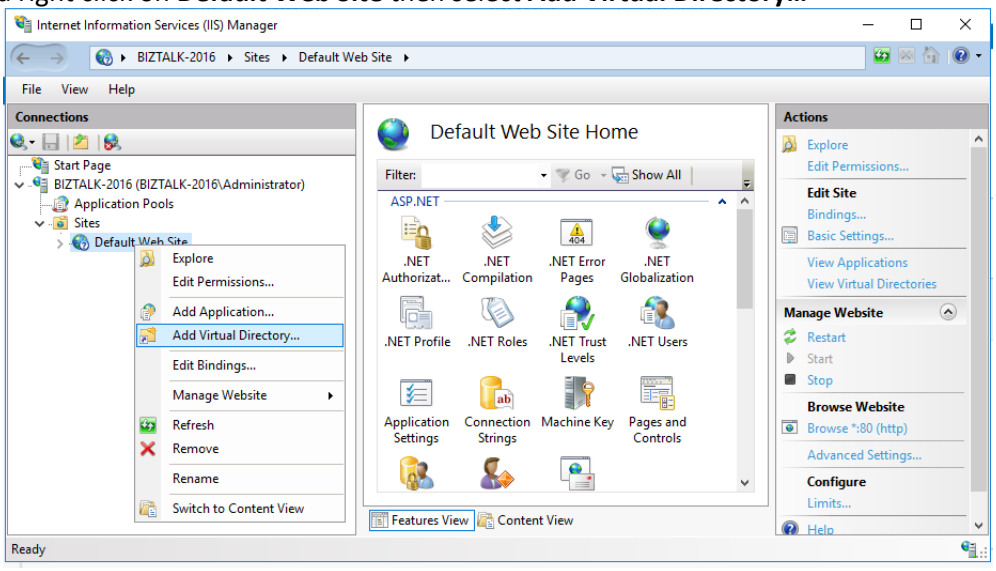


- In the **User name** box, enter the **Administrator**.
- In the **Password** and **Confirm password** boxes, enter **Syst1ntBZTLK19**, and then click **OK** three times. *These are the same login information you use to log into the environment as an Administrator.*
- Right-click **WcfMyService** and then click **Recycle**.

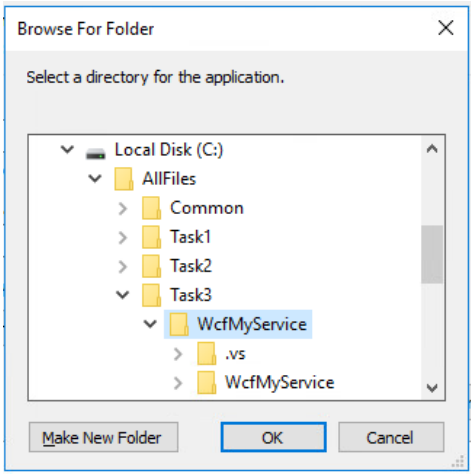
Adding Virtual Directory

***Note**: A **virtual directory** is a directory name used in URL addresses, which corresponds to a physical directory on the server. You can add a virtual directory that will include directory content in a site or an application without revealing the physical directory.*

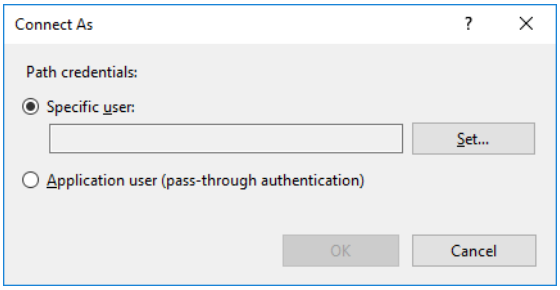
- In the **Internet Information Services (IIS) Manager** window, in the left pane, expand the **Sites** and right click on **Default Web Site** then select **Add Virtual Directory…**



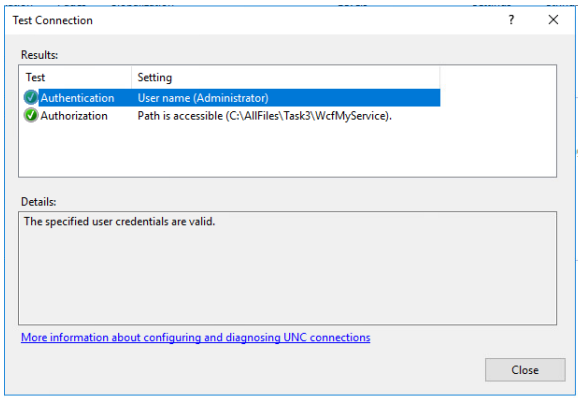
- Under **Alias:** enter **WcfMyServiceDir**
- Under the **Physical path:** locate the physical folder where the web service is located, see the Figure below.



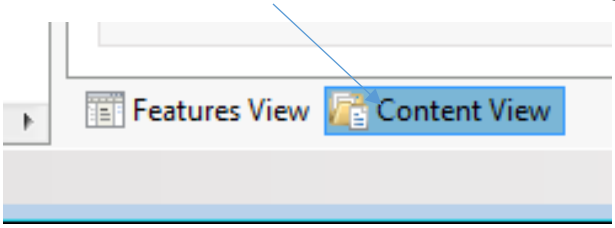
- From your **Add Virtual Directory** dialog box, Select **Connect as ...** button
- Then select **Specific User**: as illustrated below.



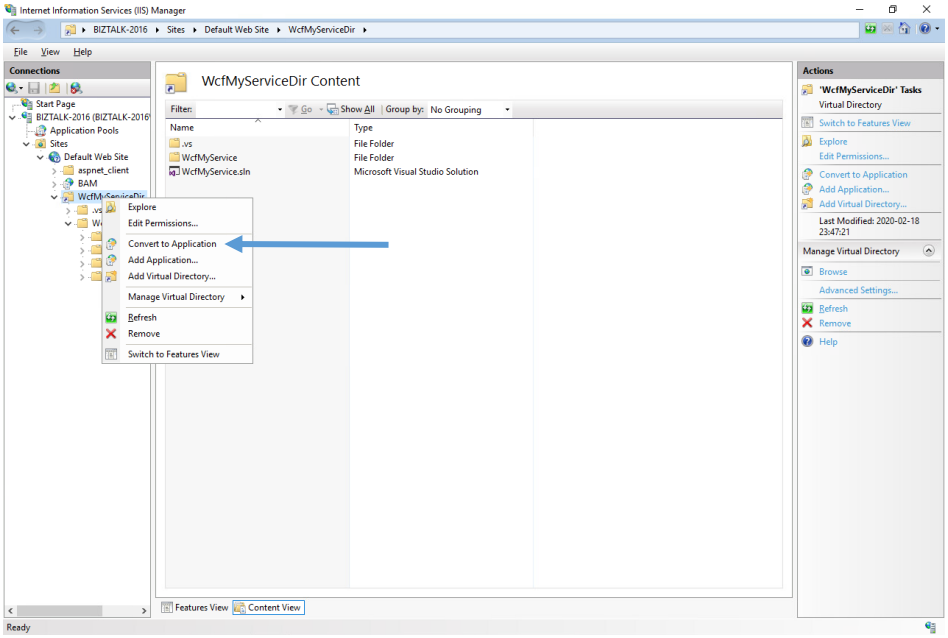
- Click on **Set …** button, specify username: **Administrator**, password: **Syst1ntBZTLK19**, confirm passwords: **Syst1ntBZTLK19** of the current user, and then click on **OK**
- From your **Add Virtual Directory** dialog box to check if the connection works, click on **Test Settings …** make sure that your dialog box looks like below.



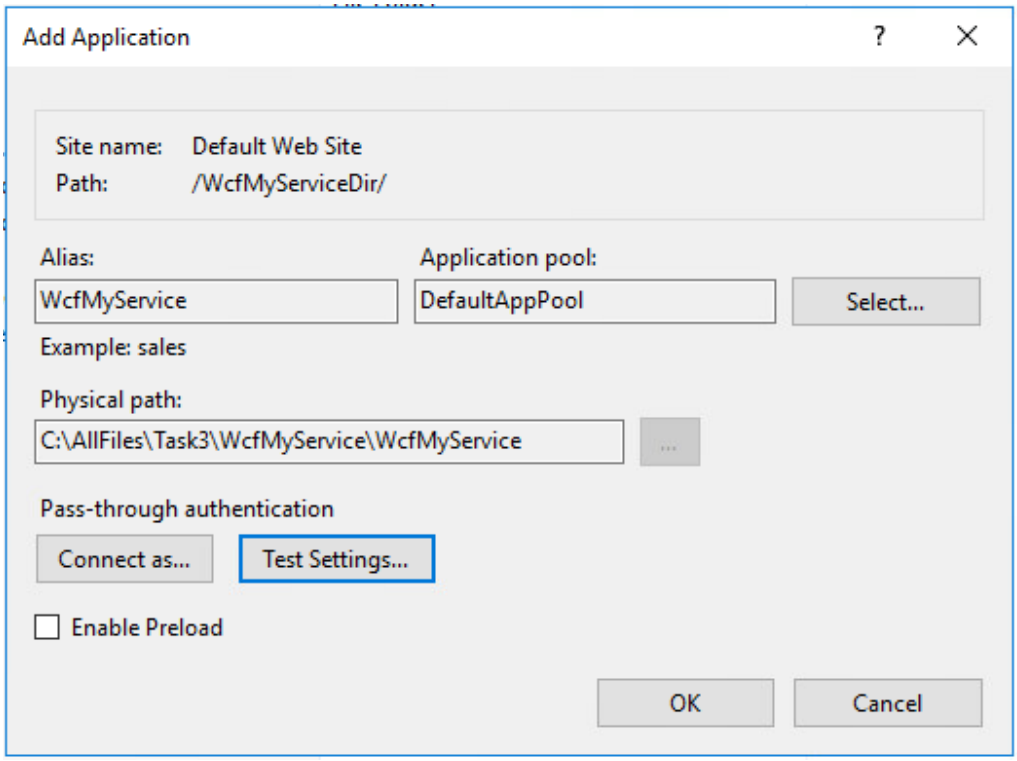
- Finally, click on **Close** and click on **OK**
- Now click on the **WcfMyServiceDir** folder and then Switch to **Content View** on the lower edge of the **Internet Information Service (IIS) Manager** window.



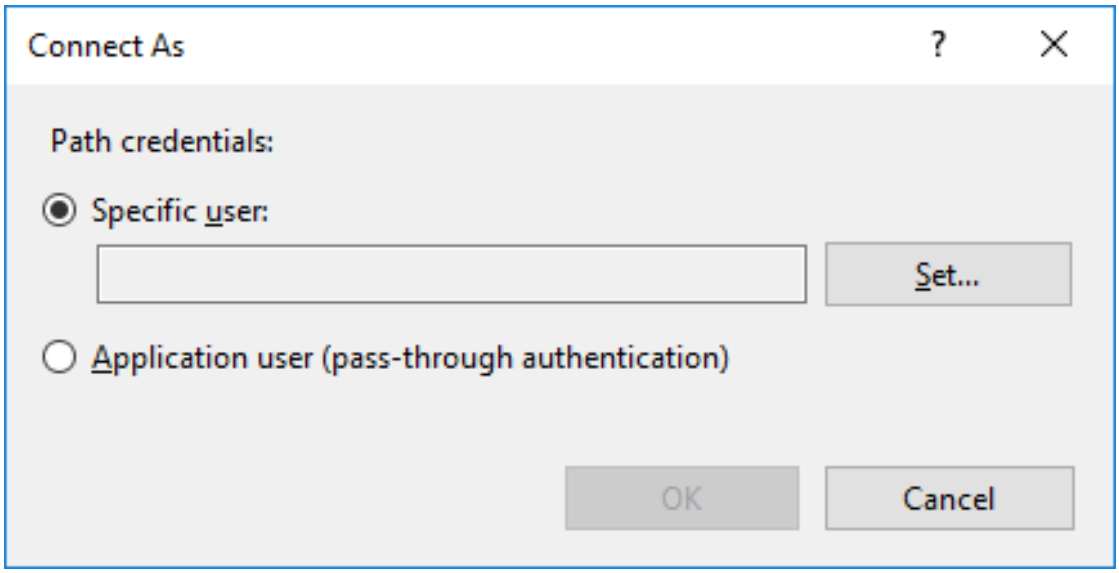
- By now the virtual directory is created manually as **WcfMyServiceDir,**
- Right-click on **WcfMyServiceDir** folder from the right panel, select **Convert to Application** and click on **OK** button



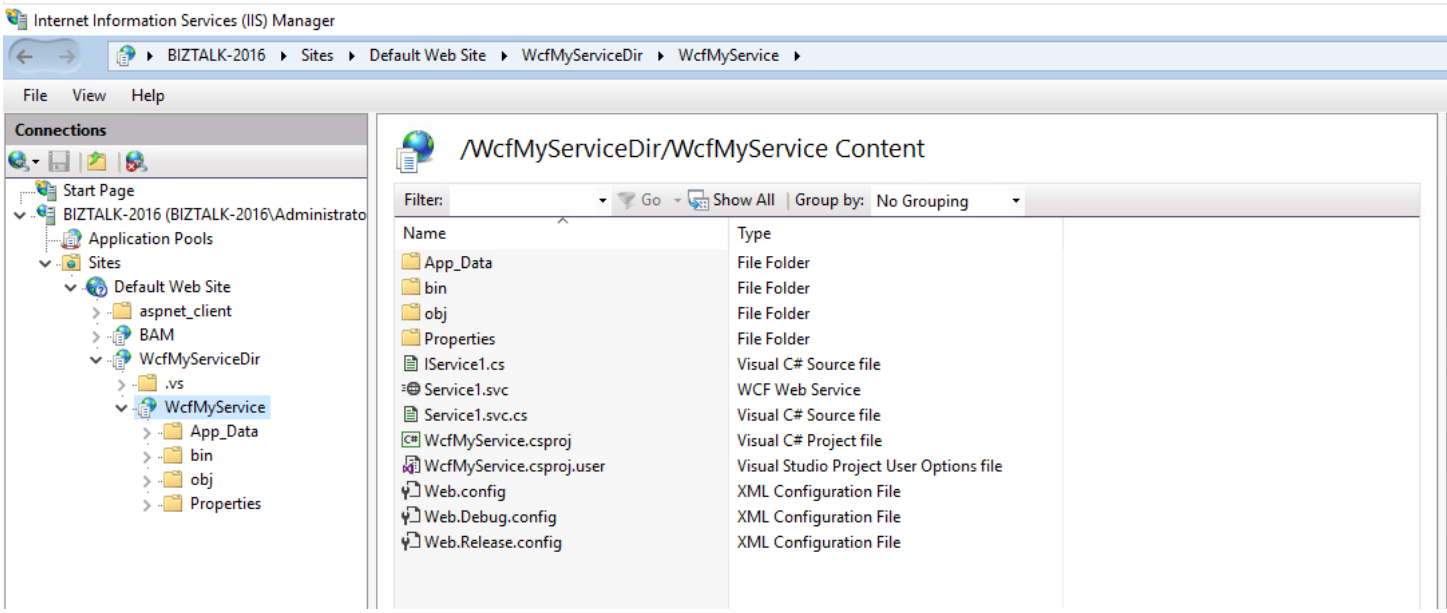
- Right-click on the folder containing the service i.e., the **WcfMyService** then click on **Convert Application**



- On the **Add Application** dialog box, click on **Connect as …**

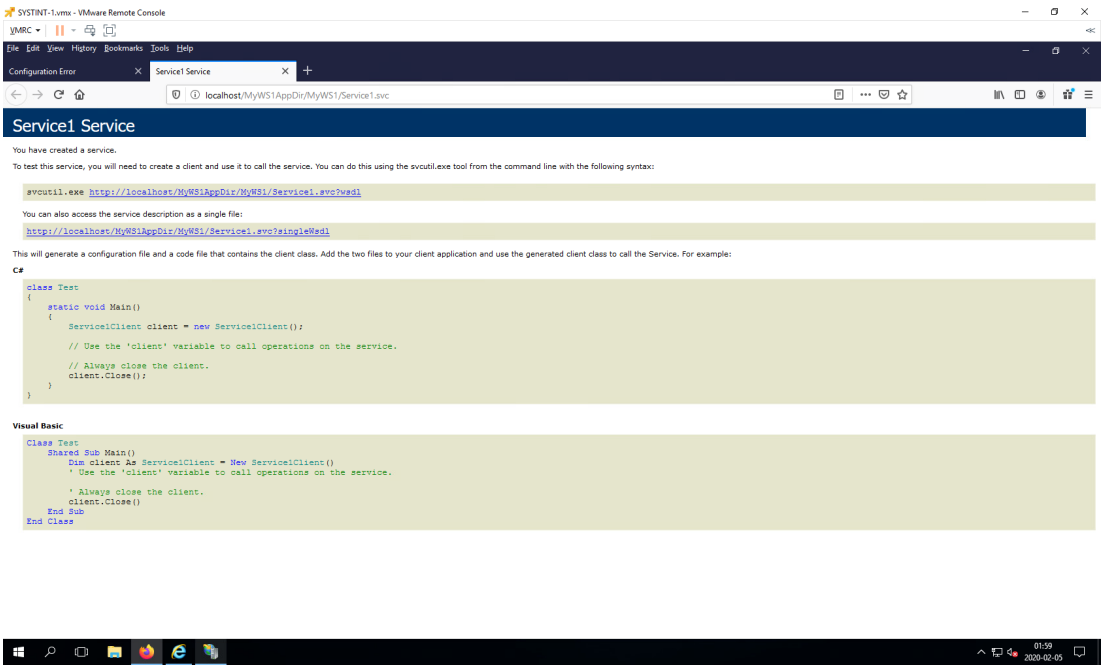


- Then select **Specific User**:
- Select **Set …** button to specify username: **Administrator**, password: **Syst1ntBZTLK19**, Confirm passwords: **Syst1ntBZTLK19** of the current user > Click on **OK**
- You can check if the connection works by clicking on **Test Settings …** then click on **OK**



Testing if the web service is ready

- Select **Service1.svc** then click on **Browse** under Actions panel on the right side of the window
- You should see the service description as illustrated below in a browser



Note:

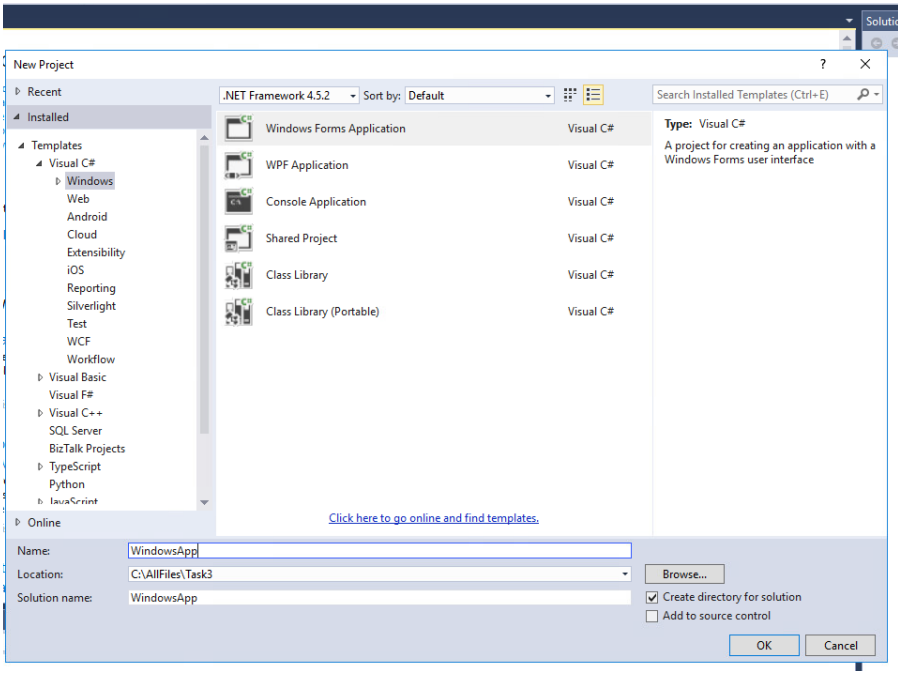
*You should **copy** the **URL** of the **Service1.svc** "http://localhost/WcfMyServiceDir/WcfMyService/Service1.svc" from the address bar so that you can add it in your service reference when developing the windows application.*

Creating a Windows client to consume web services

Creating a windows application

Steps

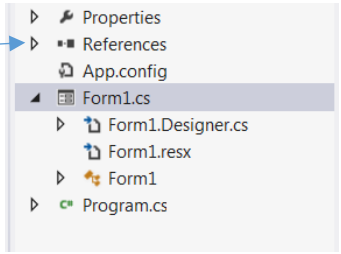
- start **Visual Studio 2015**
- select **New Project…** in Start Page
- select **Windows** under **Visual C#**
- select **Windows Forms Application**
- specify **Name** (WindowApp) and **Location** (C:\AllFiles\Task3)



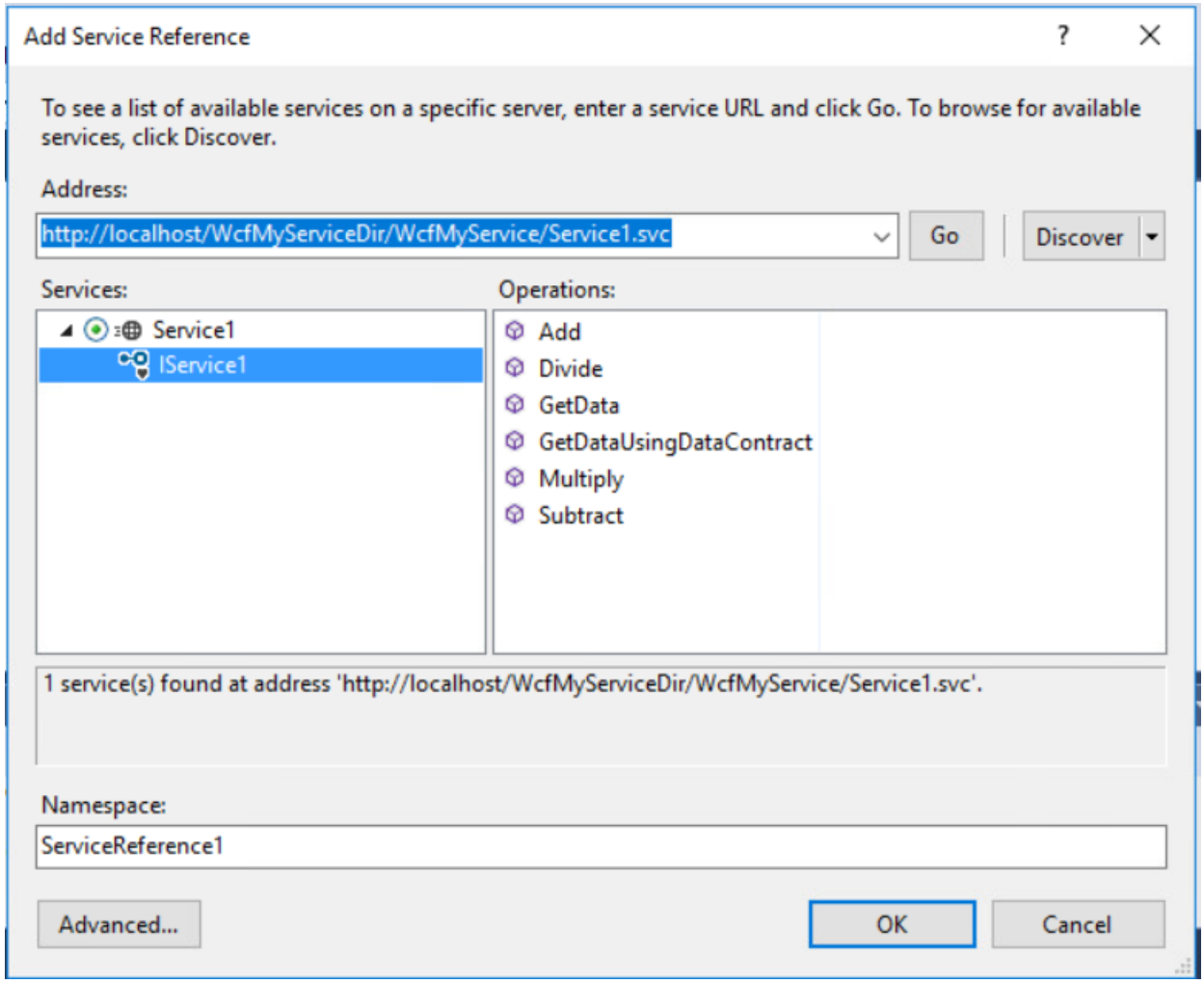
- click on **OK**

Add web service reference to the windows application

Steps



- Under **Solution Explorer** right-click on **References**
- Select **Add service reference**
  - under Address: enter the URL - http://localhost/WcfMyServiceDir/WcfMyService/Service1.svc then click **Go** to view the service

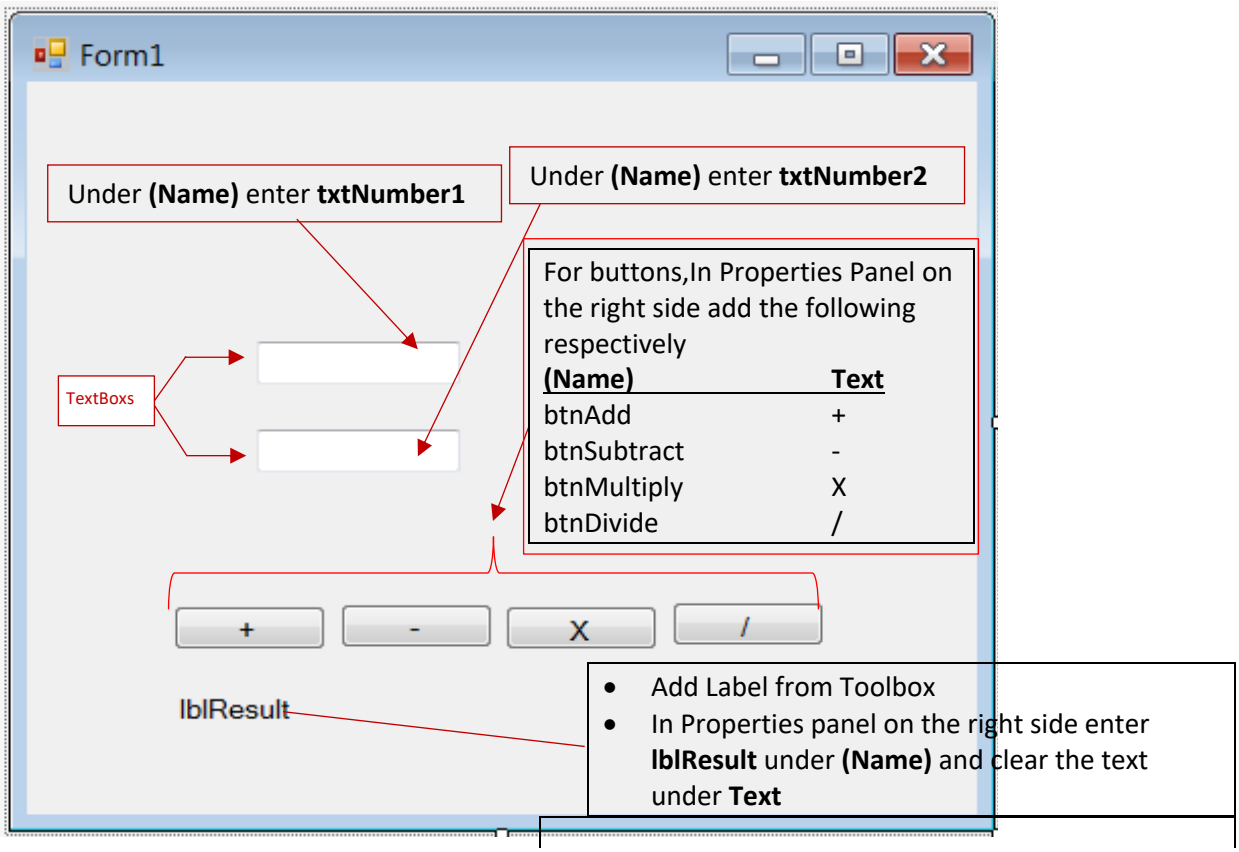


- You can now view available services and their corresponding services
- Make sure **Namespace** is **ServiceReference1**, click on **OK**

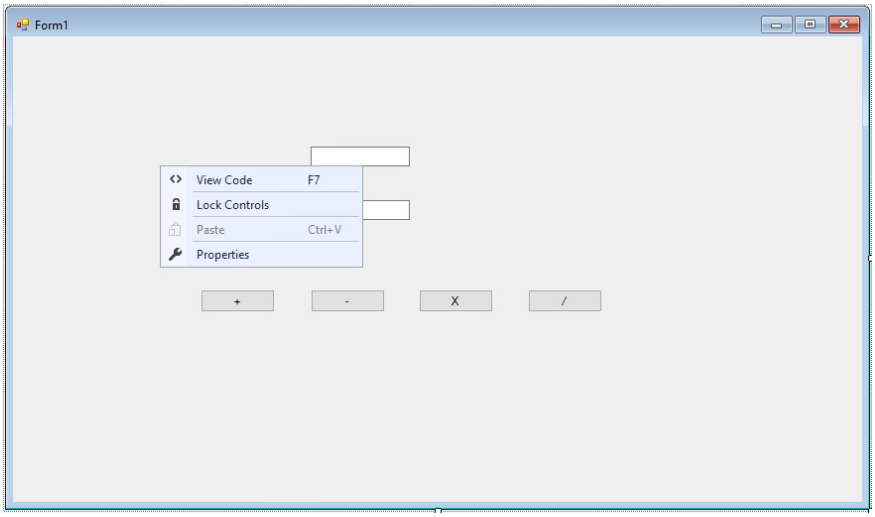
Add elements in windows controls as illustrated below

Steps

- From Toolbox, select and add TextBoxs, Buttons, and a Label, as illustrated below
- Name these controls as illustrated in windows properties
- select **OK**



- Right-click on the empty space on the form then click on **View Code**



- Read the following code, i.e., you could copy it or enter it manually in the next step

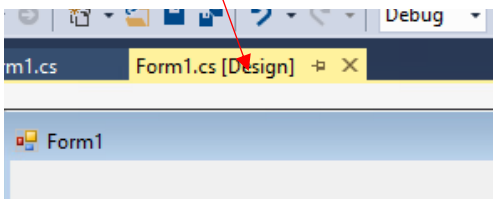
```
ServiceReference1.Service1Client sc = new ServiceReference1.Service1Client();
```

- Enter the code or (paste the code if you copied it) at the position indicated with a red arrow line as illustrated below. *If you enter the code manually Visual Studio will assist you by suggestion of valid inputs.*

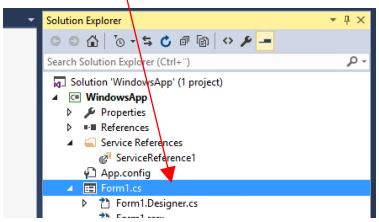
```
public partial class Form1 : Form
{
    ServiceReference1.Service1Client sc = new ServiceReference1.Service1Client();
    public Form1()
    {
        InitializeComponent();
    }
```

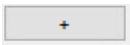
Add the *Add* method implementation code

- Select **Form1.cs [Design]** or Double click on **Form1.cs** to switch to the form layout



or



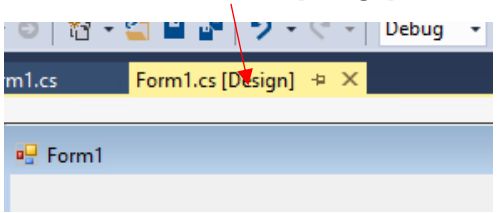
- Double click on Add button  then paste the following code in the curly brackets

```
{
    
}
```

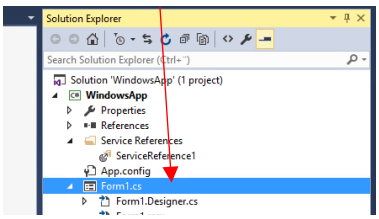
```
        lblResult.Text = "";
        double x = 0,y=0;
        if (double.TryParse(txtNumber1.Text, out x) && double.TryParse(txtNumber2.Text, out y))
            lblResult.Text = sc.Add(x, y).ToString();
        else
        {
            MessageBox.Show("Please enter valid number");
        }
```

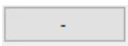
Add the Subtract method implementation code

- Select **Form1.cs [Design]** or Double click on **Form1.cs** to view the form layout



or



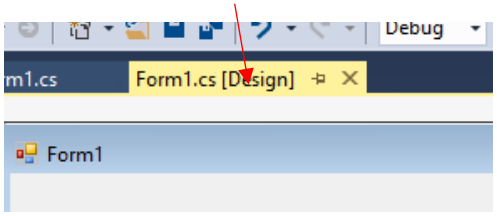
- Double click on subtract button  then paste the following code in the curly brackets

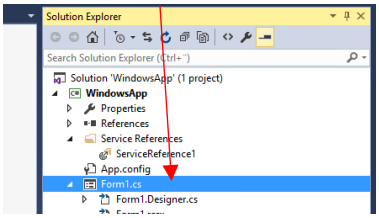
```
{
    
}
```

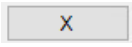
```
            double x = 0, y = 0;
lblResult.Text = "";
            if (double.TryParse(txtNumber1.Text, out x) && double.TryParse(txtNumber2.Text, out y))
                lblResult.Text = sc.Subtract(x, y).ToString();
            else
            {
                MessageBox.Show("Please enter valid number");
            }
```

Add the Multiply method implementation code

- Select **Form1.cs [Design]** or Double click on **Form1.cs** to view the form layout



or


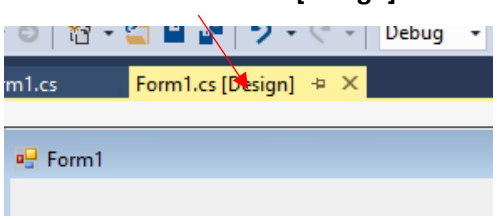
- Double click on multiply button  then paste the following code in the curly brackets

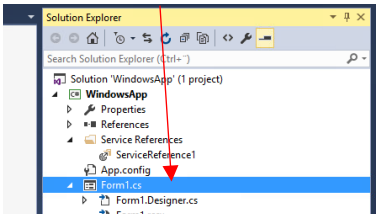

```
{
    
}
```

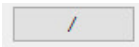
```
            lblResult.Text = "";
            double x = 0, y = 0;
            if (double.TryParse(txtNumber1.Text, out x) && double.TryParse(txtNumber2.Text, out y))
                lblResult.Text = sc.Multiply(x, y).ToString();
            else
            {
                MessageBox.Show("Please enter valid number");
            }
```

Add the *Division* method implementation code

- Select **Form1.cs [Design]** or Double click on **Form1.cs** to view the form layout



or


- Double click on Division button  then paste the following code in the curly brackets

```
{
|
}
```

```
lblResult.Text = "";
double x = 0, y = 0;
            if (!double.TryParse(txtNumber2.Text, out y))
                MessageBox.Show("You cannot divide by zero");

            else if (double.TryParse(txtNumber1.Text, out x) && double.TryParse(txtNumber2.Text, out y))
                lblResult.Text = sc.Division(x, y).ToString();
            else
                MessageBox.Show("Please enter valid number");
```

Here the complete code

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace WindowsApp
{
    public partial class Form1 : Form
    {
        ServiceReference1.Service1Client sc = new ServiceReference1.Service1Client();
        public Form1()
        {
            InitializeComponent();
        }

        private void btnAdd_Click(object sender, EventArgs e)
        {
            lblResult.Text = "";
            double x = 0, y = 0;
            if (double.TryParse(txtNumber1.Text, out x) &&
double.TryParse(txtNumber2.Text, out y))
                lblResult.Text = sc.Add(x, y).ToString();
            else
            {
                MessageBox.Show("Please enter valid number");
            }


        }

        private void btnSubtract_Click(object sender, EventArgs e)
        {
            lblResult.Text = "";
            double x = 0, y = 0;
```

```
            if (double.TryParse(txtNumber1.Text, out x) &&
double.TryParse(txtNumber2.Text, out y))
                lblResult.Text = sc.Subtract(x, y).ToString();
            else
            {
                MessageBox.Show("Please enter valid number");
            }


        }

        private void btnMultiply_Click(object sender, EventArgs e)
        {
            lblResult.Text = "";
            double x = 0, y = 0;
            if (double.TryParse(txtNumber1.Text, out x) &&
double.TryParse(txtNumber2.Text, out y))
                lblResult.Text = sc.Multiply(x, y).ToString();
            else
            {
                MessageBox.Show("Please enter valid number");
            }

        }

        private void btnDivide_Click(object sender, EventArgs e)
        {
            lblResult.Text = "";
            double x = 0, y = 0;
            if (!double.TryParse(txtNumber2.Text, out y))
                MessageBox.Show("You cannot divide by zero");

            else if (double.TryParse(txtNumber1.Text, out x) &&
double.TryParse(txtNumber2.Text, out y))
                lblResult.Text = sc.Division(x, y).ToString();
            else
                MessageBox.Show("Please enter valid number");

        }
    }
}
```

- To run the application, click on the **Start** icon on the toolbar
- Try to use the application by adding, subtracting, multiplying and dividing numbers

References

Configuring IIS Services

1. http://www.codeproject.com/Articles/94043/SOAP-Web-Services-Create-Once-Consume-Everywhere
2. https://support.microsoft.com/en-us/kb/323972

Creating the ASP.NET WCF Web Service

3. http://www.codeproject.com/Articles/94043/SOAP-Web-Services-Create-Once-Consume-Everywhere
4. Badrinarayanan Lakshmiraghavan, 2013, Practical ASP.NET Web API, publisher Apress
5. Jamie Kurtz and Brain Wortman, 2014, ASP.NET Web API 2 building a REST Service from Start to Finish, publisher Apress

IIS – What is IIS?

6. https://msdn.microsoft.com/en-us/library/aa733738(v=vs.60).aspx
7. https://technet.microsoft.com/en-us/library/cc771804(v=ws.10).aspx

Appendix 1

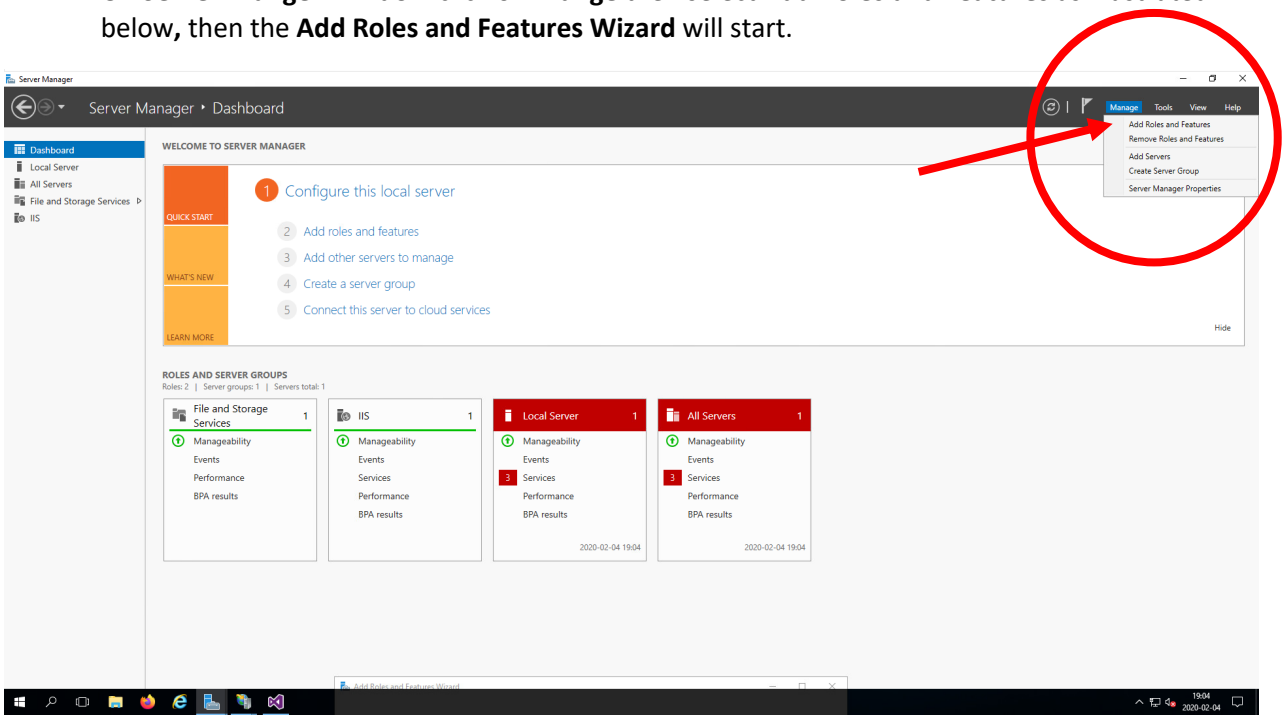
Installing IIS and Other Required Components on Windows Server 2016

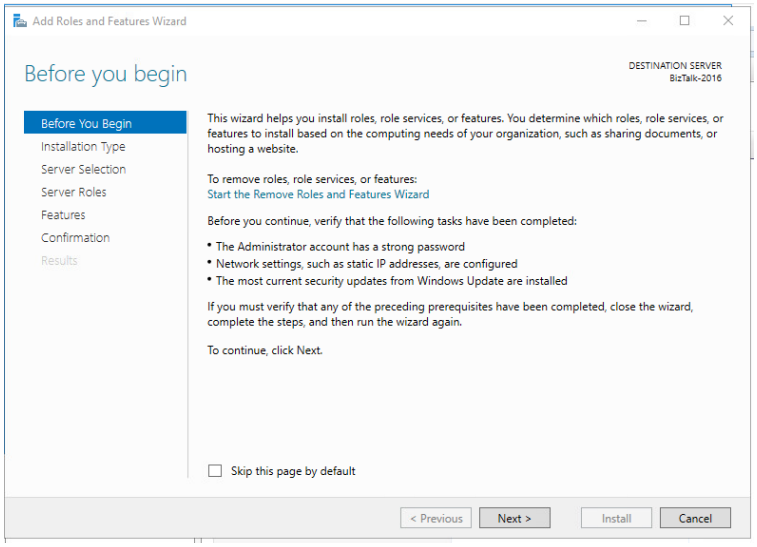
(This section describes the steps you would take if you were given or if you have a windows server 2016 machine of your own) The cloud-based infrastructure you are provisioned for the Lab will be pre-configured so that the IIS is installed and ready for work.

In this section you learn how IIS for web is installed, if IIS is not installed then it is not possible to install host web services. The server you are going to use acts as a web server, where the web service that you are going to create will be hosted and you will create an application to consume the web service. Here, the web service is added …will not do anything but if you are working on your own

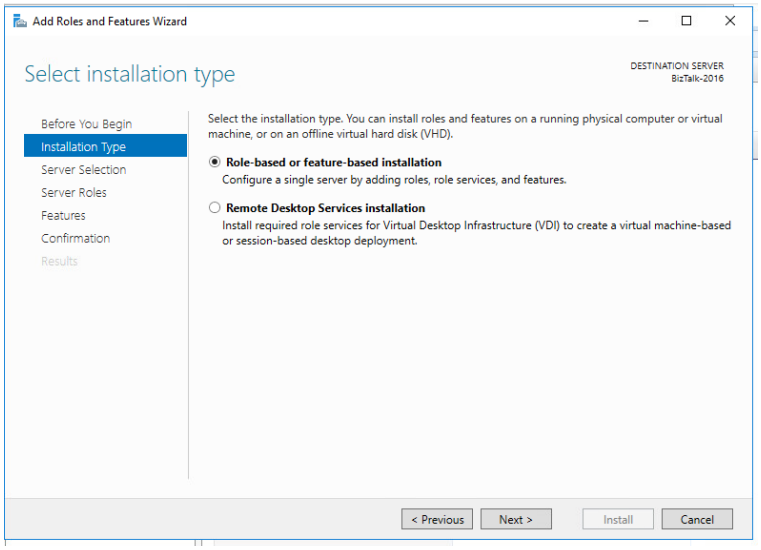
Steps (This is for demonstration purpose only so you don't need to do it if you have done before)

- Start **Server Manager** from **Start menu**
- On **Server Manger Window** click on **Mange** then select **Add Roles and Features** as illustrated below**,** then the **Add Roles and Features Wizard** will start.

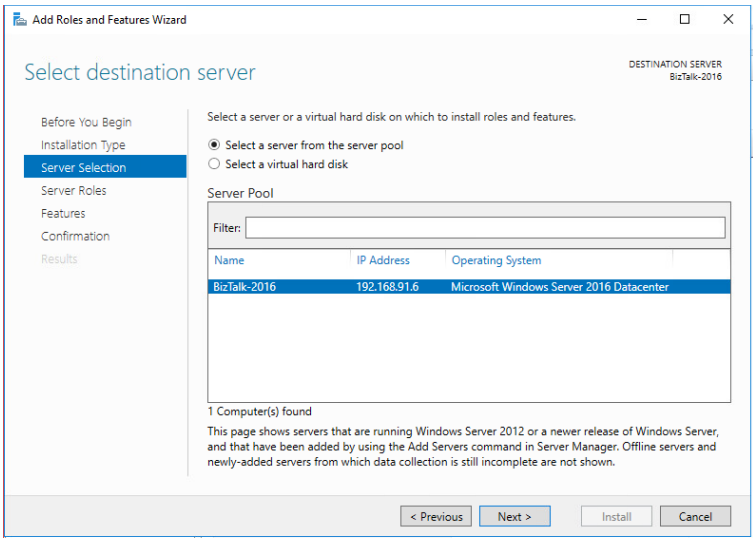




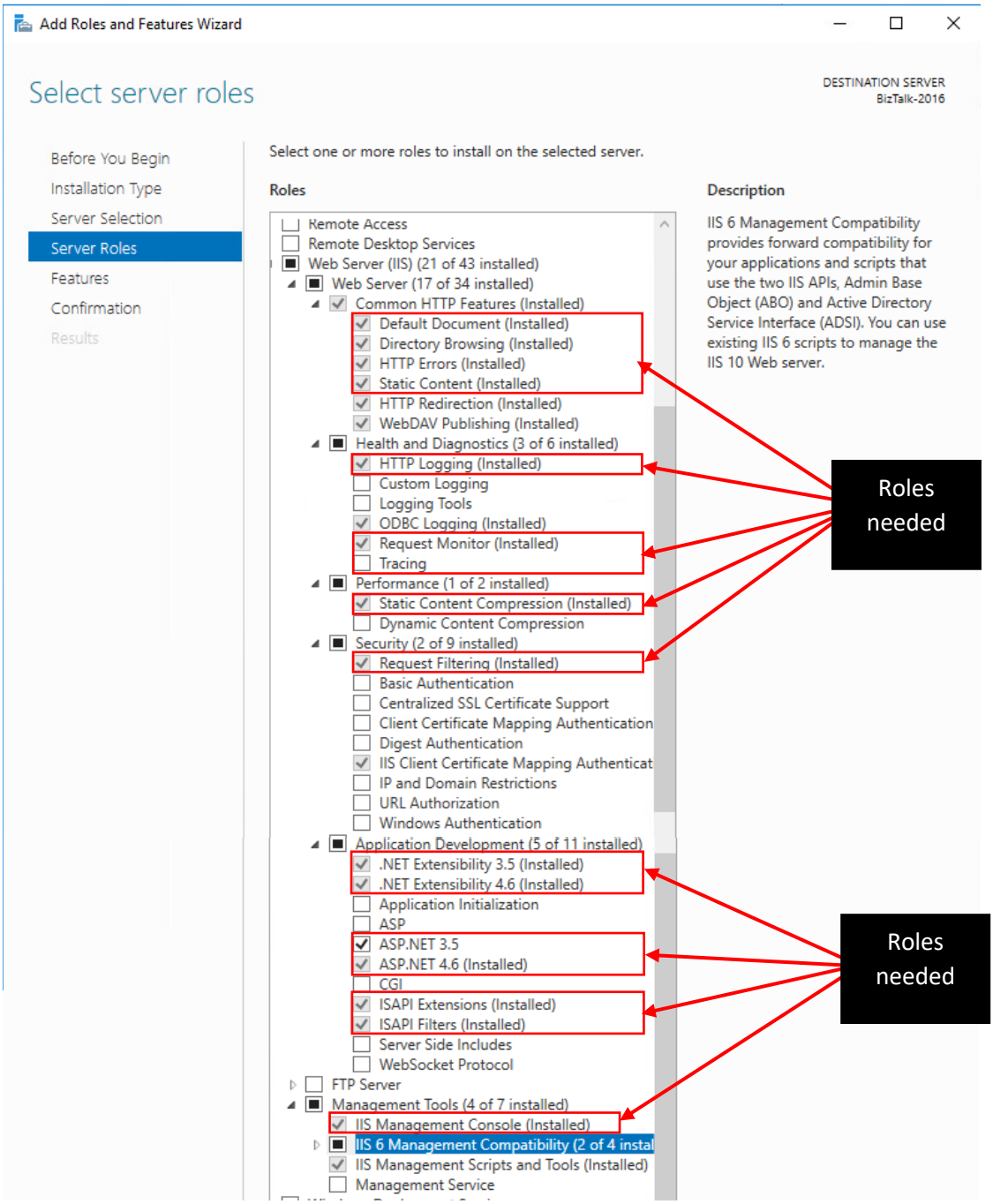
- Click on **Next**



- Select **Role-based or feature-based installation** click on **Next**



- Click on **Next**
- Under **Server Roles** > Make Sure that the following options are checked,



Next > Next > Install