Query String

The most common approach to pass information to server is by using a query string in the URL. This approach is mostly found in search engines. For example, if you perform a search on the Google website, you'll be redirected to a new URL that make use of your search parameters. For example:

http://www.google.com/search?q=shahu+college

The query string is the portion of the URL after the question mark. In example above, it defines a single variable named *q*, which contains the string *shahu+college*.

The advantage of the query string is that it's lightweight and doesn't apply any kind of burden on the server.
However, it has limitation also:

Information is limited to simple strings, which must contain URL-legal characters.

Information is clearly visible to the user and to anyone else who cares to eavesdrop on the Internet.

Many browsers impose a limit on the length of a URL (usually from 1 KB to 2 KB), you can't place a large amount of information in the query string and still be assured of compatibility with most browsers.

The Response.Redirect() methodis used to build query string:

```
Response.Redirect("newpage.aspx?recordID=10");
```

You can send multiple parameters separated with an ampersand (&):

```
Response.Redirect("newpage.aspx?recordID=10&class=BCATY");
```

The receiving page has an easier time working with the query string. It can receive the values from the QueryString dictionary collection exposed by the built-in Request object:

```
string ID = Request.QueryString["recordID"];
classname = Request.QueryString["class"];
```

Cookies

Cookies provide a way to store information for later use. ***Cookies are small files that are created in the web browser's memory (if they're temporary)*** or on the client's hard drive (if they're permanent). One advantage of cookies is that they work transparently, without the user being aware that information needs to be stored.
As cookies are stored on the user's computer as plaintext, you should never use them to store any sensitive data, such as a password.

The following line of code is used to create cookies:

```
HttpCookie myCookie = new HttpCookie("CookieName");
myCookie.Expires = DateTime.Now.AddMonths(3);
myCookie.Value = "My Cookie";
Response.Cookies.Add(myCookie);
```

To read cookie the line of code can be implemented as

```
HttpCookie myCookie = Request.Cookies.Get("CookieName");
if (myCookie != null)
{
Label1.Text = myCookie.Value; // would display "Cookie value"
}
```

Creating and Using Web Services

Steps to create web service:

Step 1: Open .Net Visual Studio, create an Empty web site.

Step 2: Right click on App location in Solution Explorer and Select “Add New Item”, form add new item dialog box select and add “WebService” and file with “.asmx” extension.
This file contains some program line as given below

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.Services;

/// <summary>
/// Summary description for WebService
/// </summary>
[WebService(Namespace = "http://tempuri.org/")]
[WebServiceBinding(ConformsTo = WsiProfiles.BasicProfile1_1)]
// To allow this Web Service to be called from script, using ASP.NET AJAX, uncomment the following line.
// [System.Web.Script.Services.ScriptService]
public class WebService : System.Web.Services.WebService {

    public WebService () {

        //Uncomment the following line if using designed components
        //InitializeComponent();
    }

    [WebMethod]
    public string HelloWorld() {
        return "Hello World";
    }
```

}

From above line of code the part

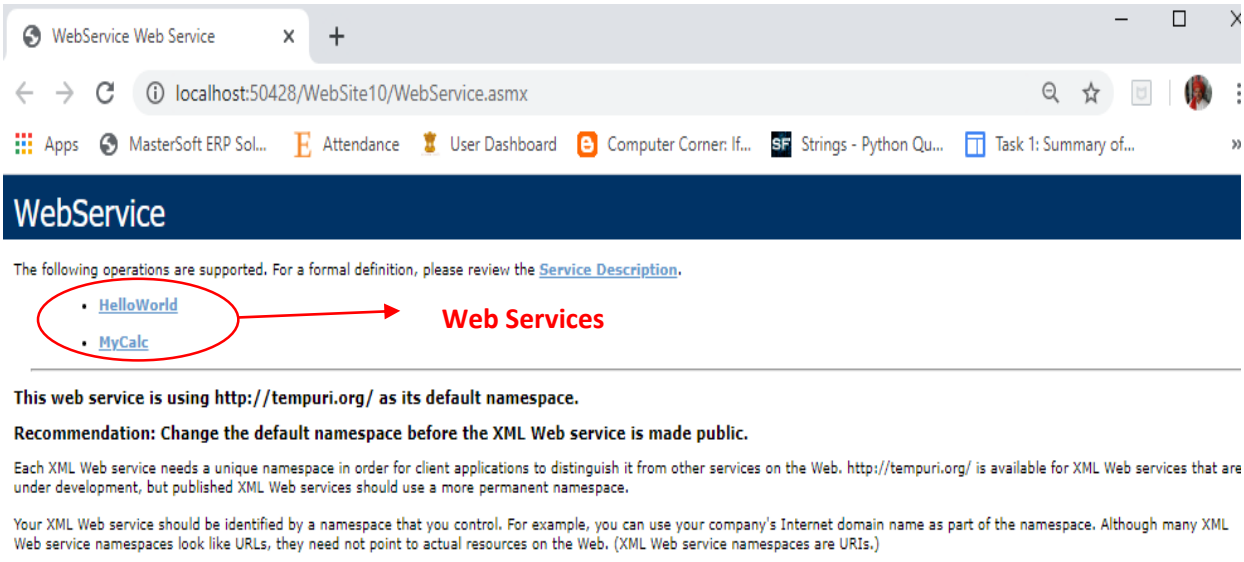
```
[WebMethod]
public string HelloWorld() {
    return "Hello World";
}
```

Containing the methods used as web services. You can add your own methods, like

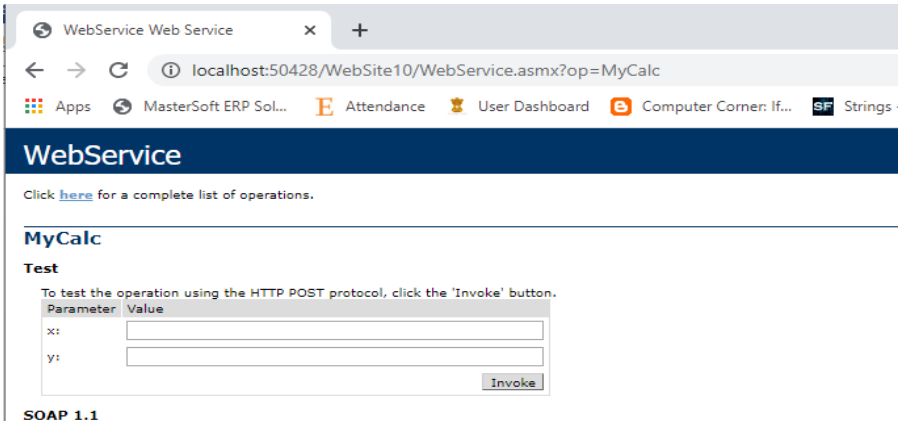
```
[WebMethod]
public int MyCalc(int x, int y)
{
    return x * y;
}
```

Accordingly go on adding methods as per the need of our web application.

Step 3: To test these newly added methods just execute the project, it will run in web browser, and the following output will be observed. After Executing it will display the names of web services.



To execute these services just click on the name of the web service. The output screen of "MyCalc" web service can be shown as



As MyCalc service requires two parameters, it will ask for two values.

Making the use of web service in the Web Application:

Step1: To make use of this Web Service add an Web Form (i.e. Default.aspx page) in the project.

Step 2: Right click in solution explorer, select “add web reference”, this will open Add Web Reference dialog box, from the dialog box, Click on “Select Web Services in this solution” as shown in the figure.

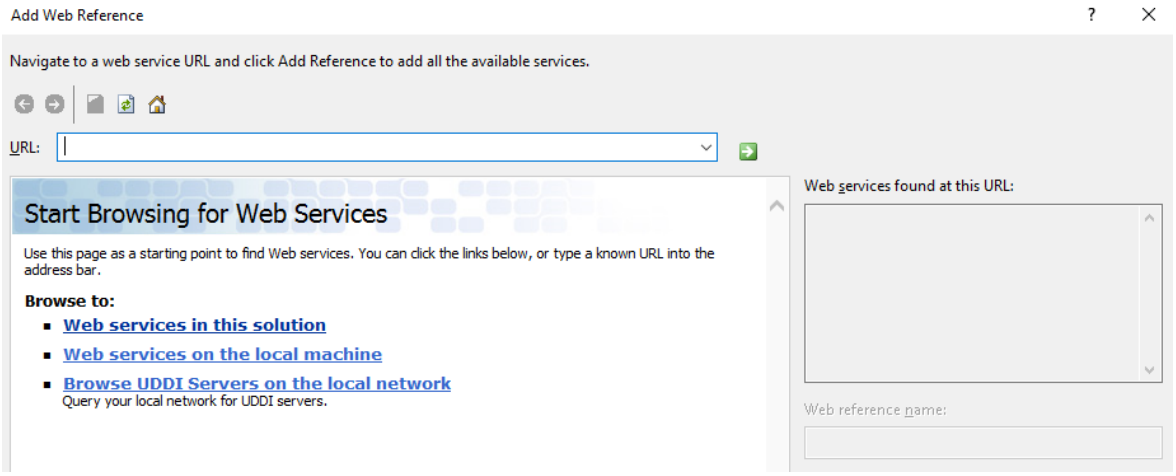


Fig 1. Add Web Reference

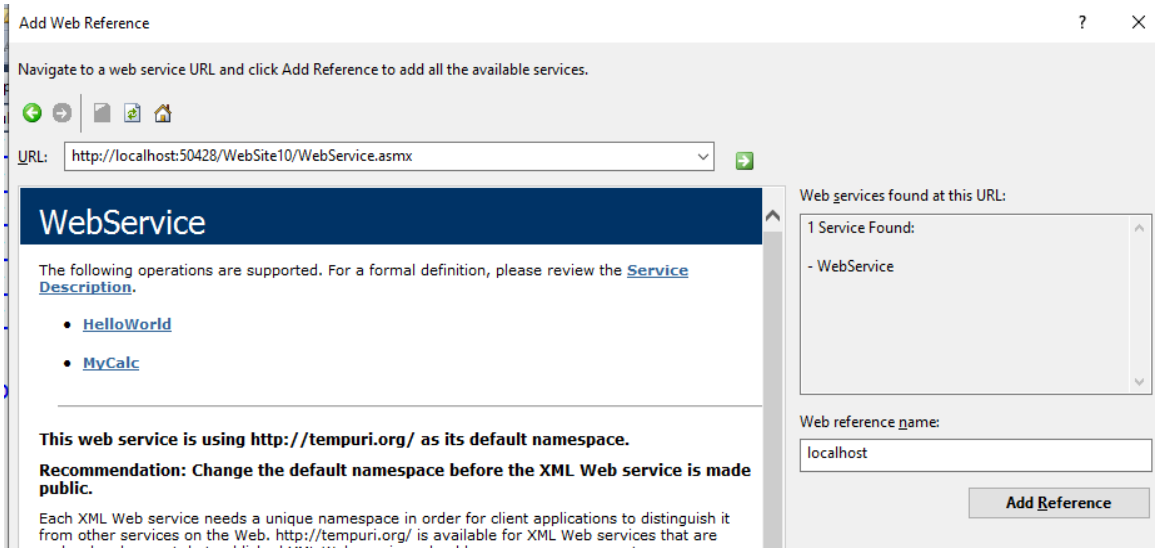


Fig 2. Showing URL for web services and Web Reference Name

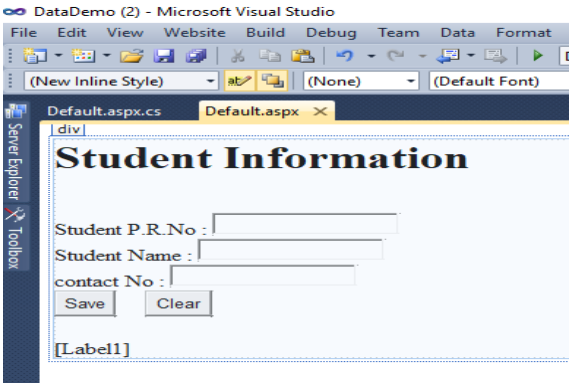
Step 3: From the dialog box above click on the Add Reference button. This will add “.disco” and “.wsdl” files in the application.

Step 4: Now add two textbox controls and a button control, and on the click event of button control add the following line of code in code behind model of the Default.aspx file.

```
localhost.WebService wc = new localhost.WebService();
int a = int.Parse(TextBox1.Text);
int b = int.Parse(TextBox2.Text);
int c = wc.MyCalc(a, b);
Label1.Text = "Answer is " + c.ToString();
```

Database Connectivity:
Inserting a record in the table:

Design of Application:



Programme code for the design:

```
<%@ Page Language="C#" AutoEventWireup="true" CodeFile="Default.aspx.cs" Inherits="_Default" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
  <title></title>
</head>
<body>
  <form id="form1" runat="server">
  <div>
  <h1>Student Information</h1><br />
  Student P.R.No : <asp:TextBox ID=txtprno runat="server"></asp:TextBox><br />
  Student Name : <asp:TextBox ID=txtsname runat="server"></asp:TextBox><br />
  contact No : <asp:TextBox ID = txtcno runat="server"></asp:TextBox><br />
  <asp:Button ID="btnSave" Text="Save" runat="server" onclick="btnSave_Click" />
  <asp:Button ID="btnClear" Text="Clear" runat="server" />

    <br />
    <asp:Label ID="Label1" runat="server" Text=""></asp:Label>
  </div>
  </form>
</body>
</html>
```

C# code for database connectivity:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Data.OleDb;
using System.Data;

public partial class _Default : System.Web.UI.Page
{
   OleDbConnection con = new OleDbConnection(@"Provider=Microsoft.ACE.OLEDB.12.0;Data
Source=E:\DataBaseDemo\student.accdb");
   OleDbCommand cmd;
   protected void Page_Load(object sender, EventArgs e)
   {
      con.Open();

   }
   protected void btnSave_Click(object sender, EventArgs e)
   {
       string s;
       s = "insert into studdtl values(" + txtprno.Text + ",'" + txtsname.Text + "'," + txtcno.Text + ")";
       cmd = new OleDbCommand();
       cmd.Connection = con;
       cmd.CommandText = s;
       cmd.ExecuteNonQuery();
       Label1.Text = "Record Saved..." + txtprno.Text;
    }
}
```