Lab :- 6
Name :- Nisarg .K. Amlani
Roll :- Ce001
Id :- 22ceueg082

---

1) Use the calculator WCF service/ any other WCF service developed in the previous labs.

2) Host the service in the Windows Form application. Define metadata and application endpoints over TCP protocol in the host application.

3) Develop a Windows Form client application to consume the service.

4) Update the host and client applications to provide options for consuming the service with different protocols (Pipe, HTTP and TCP).

Code IService.cs

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

```
using System.ServiceModel;
using System.Text;
using System.Threading.Tasks;

namespace WCFServiceLib
{
    [ServiceContract]
    public interface IService
    {
        [OperationContract]
        string GetData(int value1, int value2, char op);
    }
}
```

Code Service.cs

```
namespace WCFServiceLib
{
    public class Service : IService
    {
        public string GetData(int value1, int value2, char op)
        {
            double res ;
            switch(op)

            {
                case '+': res = value1 + value2; break;
                case '-': res = value1 - value2; break;
                case '*': res = value1 * value2; break;
                case '/': res = value1 / value2; break;
                default: return "Invalid Operation";
            }
            return string.Format("Your Result is : {0}",res);

        }
    }
}
```

Code Form1.cs (Host App)

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

namespace WcfServiceHost
{

    public partial class Form1 : Form
    {
        private ServiceHost _sh = null;
        public Form1()
        {
            InitializeComponent();
        }

        protected override void OnLoad(EventArgs e)
        {
            base.OnLoad(e);
            Uri tcp = new Uri("net.tcp://localhost:8010/TcpBinding");
            Uri pipe = new Uri("net.pipe://localhost/NetNamedPipeBinding");
            Uri http = new Uri("http://localhost:8733/Design_Time_Addresses");

            _sh = new ServiceHost(typeof(Service), tcp,http,pipe);
            NetTcpBinding tcpb = new NetTcpBinding();
            NetNamedPipeBinding nppb = new NetNamedPipeBinding();
            NetHttpBinding httpb = new NetHttpBinding();
            ServiceMetadataBehavior metadataBehavior = new ServiceMetadataBehavior();
            _sh.Description.Behaviors.Add(metadataBehavior);
```

```
_sh.AddServiceEndpoint(typeof(IMetadataExchange),MetadataExchangeBindings.Crea
teMexTcpBinding(), "mex");
            _sh.AddServiceEndpoint(typeof(IService), tcpb, tcp);

            _sh.AddServiceEndpoint(typeof(IMetadataExchange),
MetadataExchangeBindings.CreateMexNamedPipeBinding(),
                "mex");
            _sh.AddServiceEndpoint(typeof(IService), nppb, pipe);

            _sh.AddServiceEndpoint(typeof(IMetadataExchange),
MetadataExchangeBindings.CreateMexHttpBinding(), "mex");
            _sh.AddServiceEndpoint(typeof(IService), httpb, http);

            _sh.Open();
            reslbl.Text = "Service is running...";
        }
    }
}
```

Code Form1.cs(Client App)

```
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;
using WCFClient.Service;

namespace WCFClient
{
    public partial class Form1 : Form
    {
        public Form1()
        {
```

```
            InitializeComponent();
        }

        private void Calculate_Click(object sender, EventArgs e)
        {
            char operation = ' ';
            if (opr.Text.Equals("+")) operation = '+';
            if (opr.Text.Equals("-")) operation = '-';
            if (opr.Text.Equals("*")) operation = '*';
            if (opr.Text.Equals("/")) operation = '/';

            WCFClient.Service.ServiceClient client ;
            WCFClient.Service.ServiceClient client1 =
                new WCFClient.Service.ServiceClient(

WCFClient.Service.ServiceClient.EndpointConfiguration.NetHttpBinding_IService);
            WCFClient.Service.ServiceClient client2 =
                new WCFClient.Service.ServiceClient(

WCFClient.Service.ServiceClient.EndpointConfiguration.NetTcpBinding_IService);

            client = client1;
            // client = client2;

            string res = client.GetData(int.Parse(value1.Text), int.Parse(value2.Text),
operation);
            this.result.Text = res;
        }
    }
}
```

Code App.config (Service)

```
<?xml version="1.0" encoding="utf-8" ?>
<confi guration>
<appSettings>
<add key="aspnet:UseTaskFriendlySynchronizationContext" value="true" />
</appSettings>
<system.web>
<compilation debug="true" />
```

```
</system.web>
<!-- When deploying the service library project, the content of the confi g fi le must be added to the host's
app.confi g fi le. System.Confi guration does not support confi g fi les for libraries. -->
<system.serviceModel>
<services>
<service name="WCFServiceLib.Service">
<host>
<baseAddresses>
<add baseAddress =
"http://localhost:8733/Design_Time_Addresses/WCFServiceLib/Service/" />
</baseAddresses>
</host>
<!-- Service Endpoints -->
<!-- Unless fully qualifi ed, address is relative to base address supplied above -->
<endpoint address="" binding="basicHttpBinding" contract="WCFServiceLib.IService">
<!--
Upon deployment, the following identity element should be removed or replaced to refl ect the
identity under which the deployed service runs. If removed, WCF will infer an appropriate identity
automatically.
-->
<identity>
<dns value="localhost"/>
</identity>
</endpoint>
<!-- Metadata Endpoints -->
<!-- The Metadata Exchange endpoint is used by the service to describe itself to clients.
→

<!-- This endpoint does not use a secure binding and should be secured or removed before deployment -->
<endpoint address="mex" binding="mexHttpBinding" contract="IMetadataExchange"/>
</service>
</services>
<behaviors>
<serviceBehaviors>
<behavior>
<!-- To avoid disclosing metadata information,
```

```
set the values below to false before deployment -->
<serviceMetadata httpGetEnabled="True" httpsGetEnabled="True"/>
<!-- To receive exception details in faults for debugging purposes,
set the value below to true. Set to false before deployment
to avoid disclosing exception information -->
<serviceDebug includeExceptionDetailInFaults="False" />
</behavior>
</serviceBehaviors>
</behaviors>
</system.serviceModel>
</confi guration>
```

Code App.config (Host App)

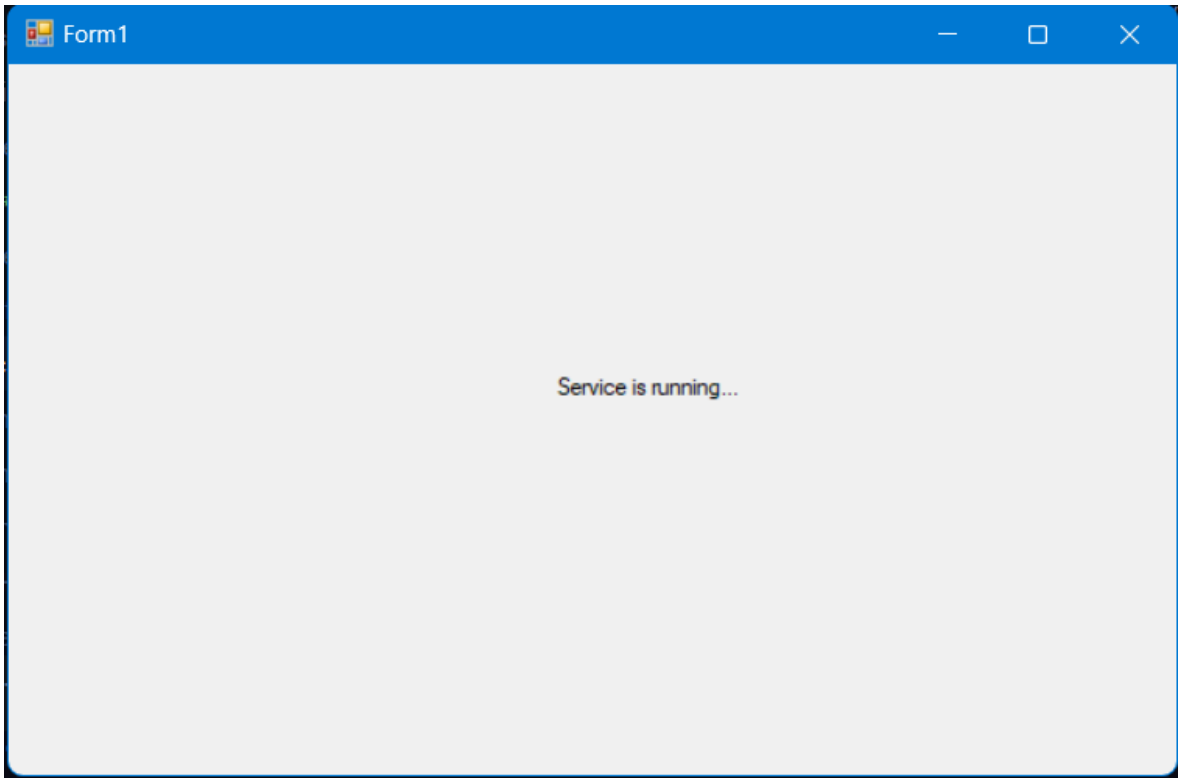
```
<?xml version="1.0" encoding="utf-8" ?>
<configuration>
   <startup>
      <supportedRuntime version="v4.0" sku=".NETFramework,Version=v4.8" />
   </startup>
</configuration>
```

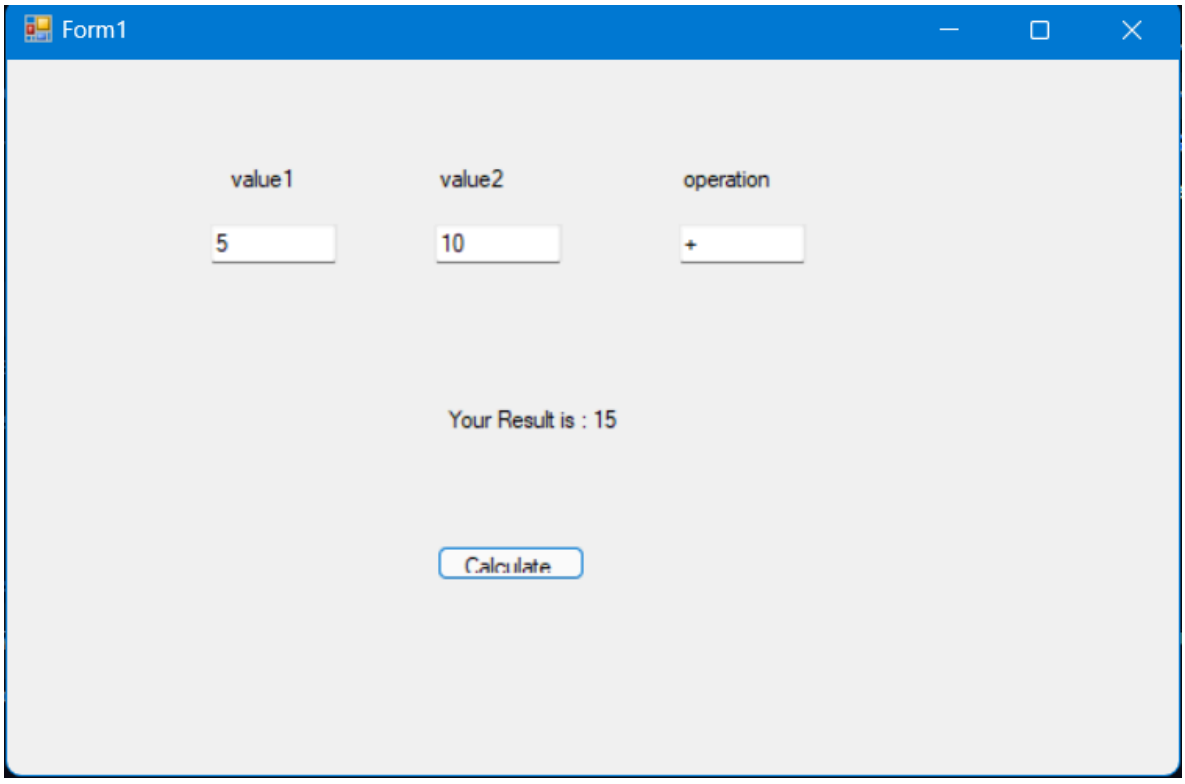
Code App.config (Client App)

```
<?xml version="1.0" encoding="utf-8" ?>
<configuration>
   <startup>
      <supportedRuntime version="v4.0" sku=".NETFramework,Version=v4.8" />
   </startup>
   <system.serviceModel>
      <bindings>
         <netTcpBinding>
                              <binding name="NetTcpBinding_IService">
                                    <security>
                                          <transport sslProtocols="None" />
                                    </security>
                              </binding>
         </netTcpBinding>
```

```
      </bindings>
      <client>
         <endpoint address="net.tcp//localhost:8010/TcpBinding"
binding="netTcpBinding"
         bindingConfiguration="NetTcpBinding_IService" contract="tcp.IService"
name="NetTcpBinding_IService">
            <identity>
               <userPrincipalName value="NISARG\CEDDU" />
            </identity>
         </endpoint>
         <endpoint address="net.pipe://localhost/NetNamedPipeBinding"
                  binding="netNamedPipeBinding"
bindingConfiguration="NetNamedPipeBinding_IService"
                  contract="http.IService" name="NetNamedPipeBinding_IService">
            <identity>
               <userPrincipalName value="Nisarg" />
            </identity>
         </endpoint>
         <endpoint address="http://localhost:8733/Design_Time_Addresses"
                  binding="customBinding" bindingConfiguration="NetHttpBinding_IService"
                  contract="http.IService" name="NetHttpBinding_IService" />
      </client>
   </system.serviceModel>
</configuration>
```

Output :


Form1
Service is running...


Form1
value1
value2
operation
5
10
+
Your Result is : 15
Calculate