Step 2 : From SOAP to WCF-RESTful-POST

Web.config

```
<?xml version="1.0"?>
<configuration>
  <system.web>
    <compilation debug="true" targetFramework="4.0" />
  </system.web>
  <system.serviceModel>
    <services>
      <service name="RestService.RestServiceImpl" behaviorConfiguration="ServiceBehaviour">
        <!-- Service Endpoints -->
        <!-- Unless fully qualified, address is relative to base address supplied above -->
        <endpoint address ="" binding="webHttpBinding" contract="RestService.IRestServiceImpl" behaviorConfiguration="web">
          <!--
              Upon deployment, the following identity element should be removed or replaced to reflect the
              identity under which the deployed service runs.  If removed, WCF will infer an appropriate identity
              automatically.
          -->
        </endpoint>
      </service>
    </services>
    <behaviors>
      <serviceBehaviors>
        <behavior name="ServiceBehaviour">
          <!-- To avoid disclosing metadata information, set the value below to false and remove the metadata endpoint above before deployment -->
          <serviceMetadata httpGetEnabled="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>
      <endpointBehaviors>
        <behavior name="web">
          <webHttp/>
        </behavior>
      </endpointBehaviors>
    </behaviors>
    <serviceHostingEnvironment multipleSiteBindingsEnabled="true" />
  </system.serviceModel>
  <system.webServer>
    <modules runAllManagedModulesForAllRequests="true"/>
  </system.webServer>
</configuration>
```

RestServiceImpl.svc.cs

```
using System.Data;
using System.Data.SqlClient;

namespace RestService
{
    public class RestServiceImpl : IRestServiceImpl
    {
        #region IRestServiceImpl Members

        const string connStr = "server=localhost;uid=sa;pwd=kash;database=minipassport";

        public string XMLData(string id)
        {
            return "You requested product " + id;
        }

        public string JSONData(string id)
        {
            return "You requested product " + id;
        }
```

```
public string ReturnEmail(string user)
{
    SqlConnection dbConn = new SqlConnection(connStr);
    string sqlStr = "Select email from users where username = '" + user + "';";
    dbConn.Open();
    SqlCommand dbCommand = new SqlCommand(sqlStr, dbConn);
    SqlDataReader dbReader = dbCommand.ExecuteReader();
    dbReader.Read();
    string _name = dbReader[0].ToString();
    dbReader.Close();
    dbConn.Close();
    return _name;
}
```

```
public PersonData AddUser(RequestData rData)
{
    bool returnBool = false;

    var data = rData.details.Split('|');
    var response = new PersonData
    {
        Name = data[0],
        User = data[1],
        Email = data[2],
        Password = data[3]
    };

    SqlConnection dbConn = new SqlConnection(connStr);
    string sqlStr = "INSERT INTO users(username,name,email,password) values('" + data[0] + "', '" + data[1] + "', '" + data[2] + "', '" + data[3] + "');";
    SqlCommand dbCommand = new SqlCommand(sqlStr, dbConn);
    try
    {
        dbConn.Open();
        if (dbCommand.ExecuteNonQuery() != 0)
        {
            returnBool = true;
        }
    }
    catch
    {
        returnBool = false;
    }
    dbConn.Close();
    return response;
}
```

```
public bool AddUser2(RequestData rData)
{
    bool returnBool = false;

    var data = rData.details.Split('|');
    var response = new PersonData
    {
        Name = data[0],
        User = data[1],
        Email = data[2],
        Password = data[3]
    };

    SqlConnection dbConn = new SqlConnection(connStr);
    string sqlStr = "INSERT INTO users(username,name,email,password) values('" + data[0] + "', '" + data[1] + "', '" + data[2] + "', '" + data[3] + "');";
    SqlCommand dbCommand = new SqlCommand(sqlStr, dbConn);
    try
    {
        dbConn.Open();
        if (dbCommand.ExecuteNonQuery() != 0)
        {
            returnBool = true;
        }
        catch
        {
            returnBool = false;
        }
        dbConn.Close();
        return returnBool;
    }



    #endregion



  }
}
```

ResponseData.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Runtime.Serialization;
using System.Web;

namespace RestService
{
    [DataContract]
    public class ResponseData
    {
        [DataMember]
        public string Name { get; set; }

        [DataMember]
        public string Age { get; set; }

        [DataMember]
        public string Exp { get; set; }

        [DataMember]
        public string Technology { get; set; }
    }
}
```

RequestData.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Runtime.Serialization;
using System.Text;

namespace RestService
{
    [DataContract(Namespace = "")]
    public class RequestData
    {
        [DataMember]
        public string details { get; set; }
    }
}
```

PersonData.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Runtime.Serialization;
using System.Web;

namespace RestService
{
    [DataContract]
    public class PersonData
    {
        [DataMember]
        public string Name { get; set; }

        [DataMember]
        public string User { get; set; }

        [DataMember]
        public string Email { get; set; }

        [DataMember]
        public string Password { get; set; }
    }
}
```

IRestServiceImpl.cs

```
using System.ServiceModel;
using System.ServiceModel.Web;

namespace RestService
{

    [ServiceContract]
    public interface IRestServiceImpl
    {

        [OperationContract]
        [WebInvoke(Method = "GET",
            ResponseFormat = WebMessageFormat.Json,
            BodyStyle = WebMessageBodyStyle.Wrapped,
            UriTemplate = "json/{id}")]
        string JSONData(string id);
```

```
    [OperationContract]
    [WebInvoke(Method = "GET",
        ResponseFormat = WebMessageFormat.Xml,
        BodyStyle = WebMessageBodyStyle.Wrapped,
        UriTemplate = "xml/{user}")]
    string ReturnEmail(string user);


    [OperationContract]
    [WebInvoke(Method = "POST",
        ResponseFormat = WebMessageFormat.Xml,
        RequestFormat = WebMessageFormat.Xml,
        BodyStyle = WebMessageBodyStyle.Bare,
        UriTemplate = "adduser")]
    PersonData AddUser(RequestData rData);


    [OperationContract]
    [WebInvoke(Method = "POST",
        ResponseFormat = WebMessageFormat.Xml,
        RequestFormat = WebMessageFormat.Xml,
        BodyStyle = WebMessageBodyStyle.Bare,
        UriTemplate = "adduser2")]
        bool AddUser2(RequestData rData);


    }
}
```