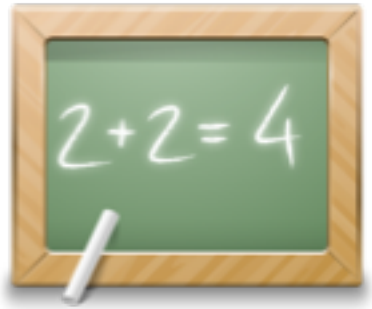
https://www.halvorsen.blog



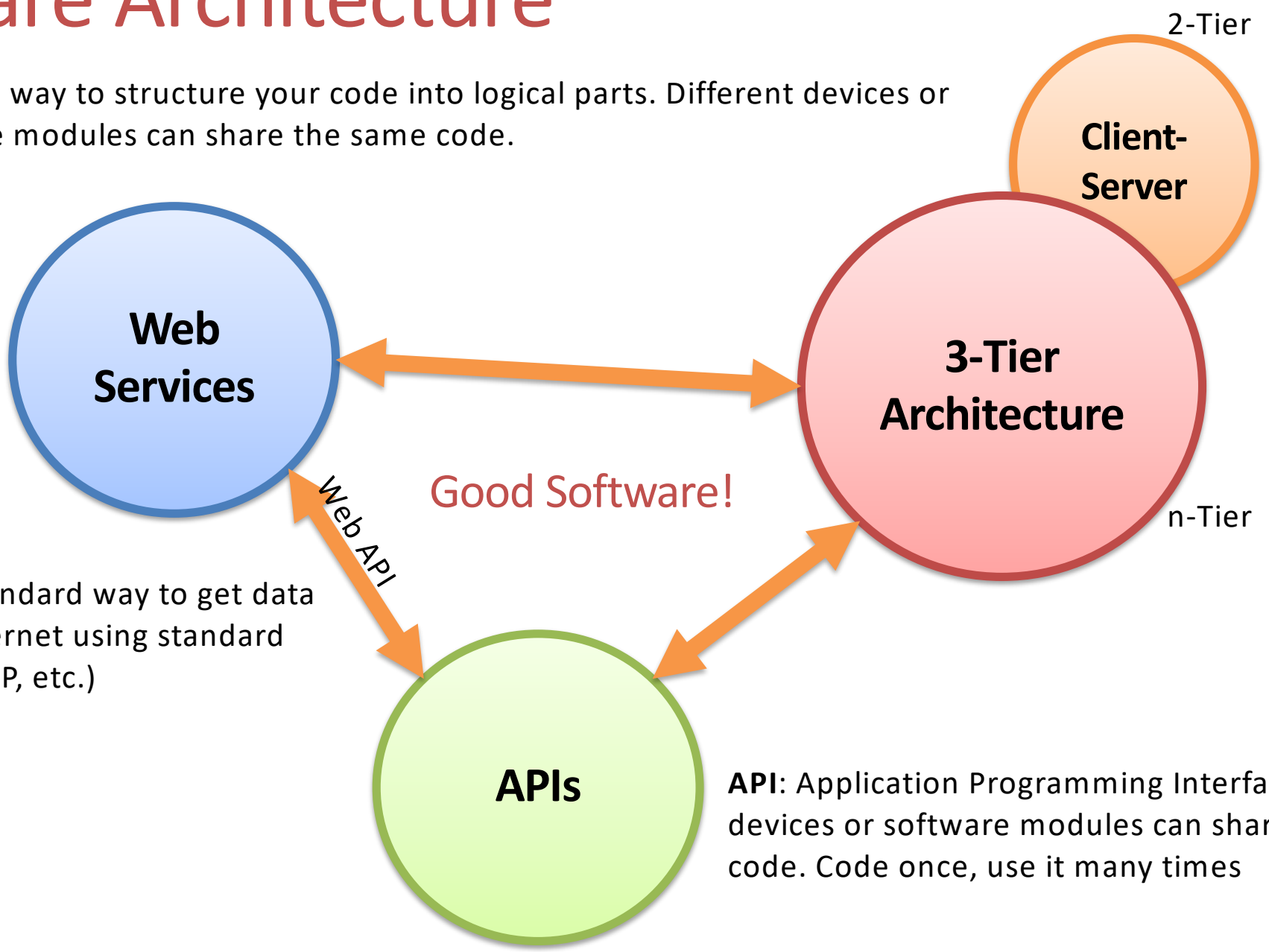
3-tier Architecture

Step by step Exercises

Hans-Petter Halvorsen

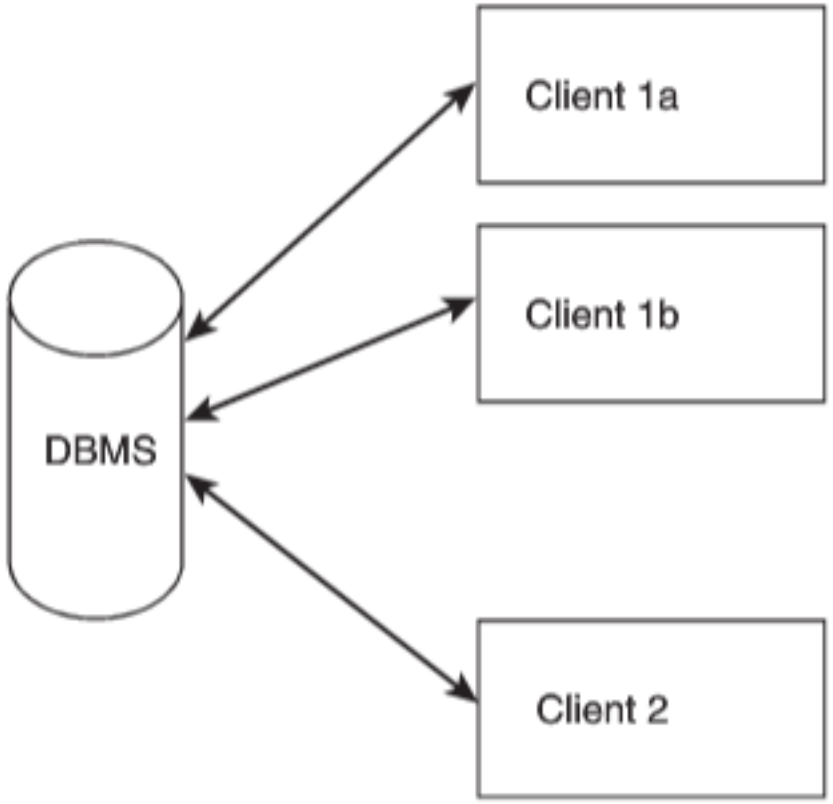
Software Architecture

3-Tier: A way to structure your code into logical parts. Different devices or software modules can share the same code.

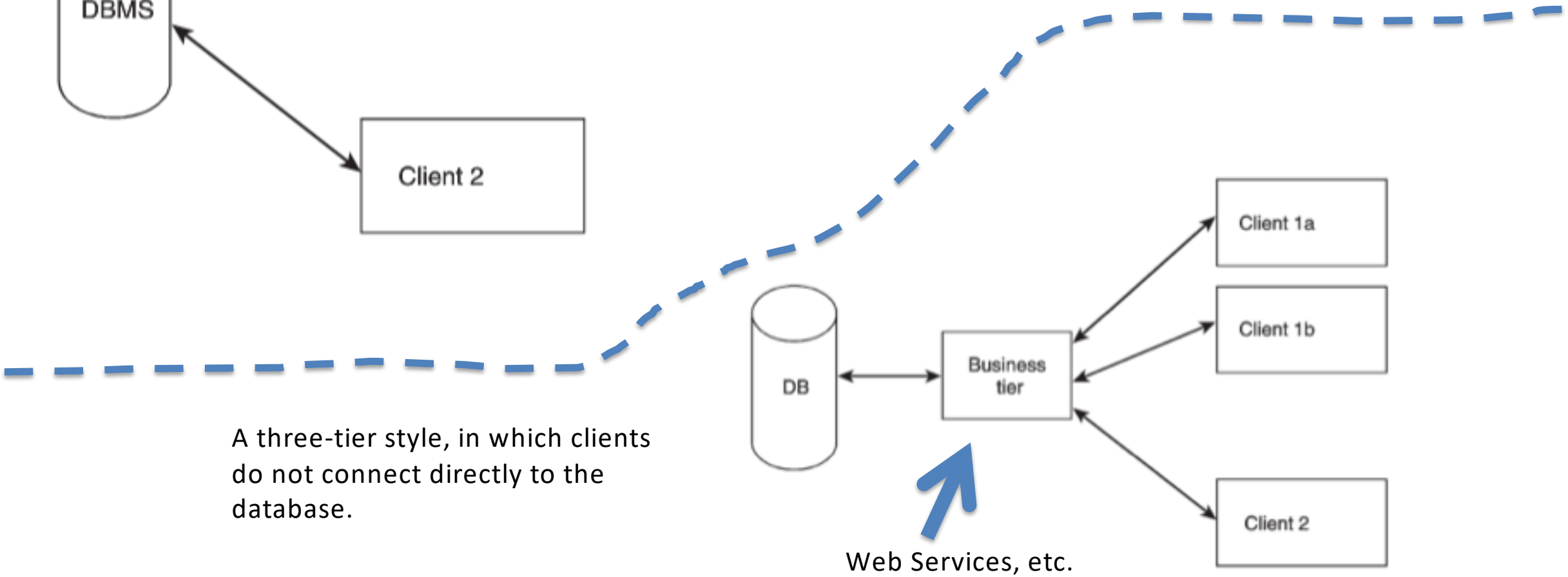


Web Services: A standard way to get data over a network/Internet using standard Web protocols (HTTP, etc.)

API: Application Programming Interface. Different devices or software modules can share the same code. Code once, use it many times



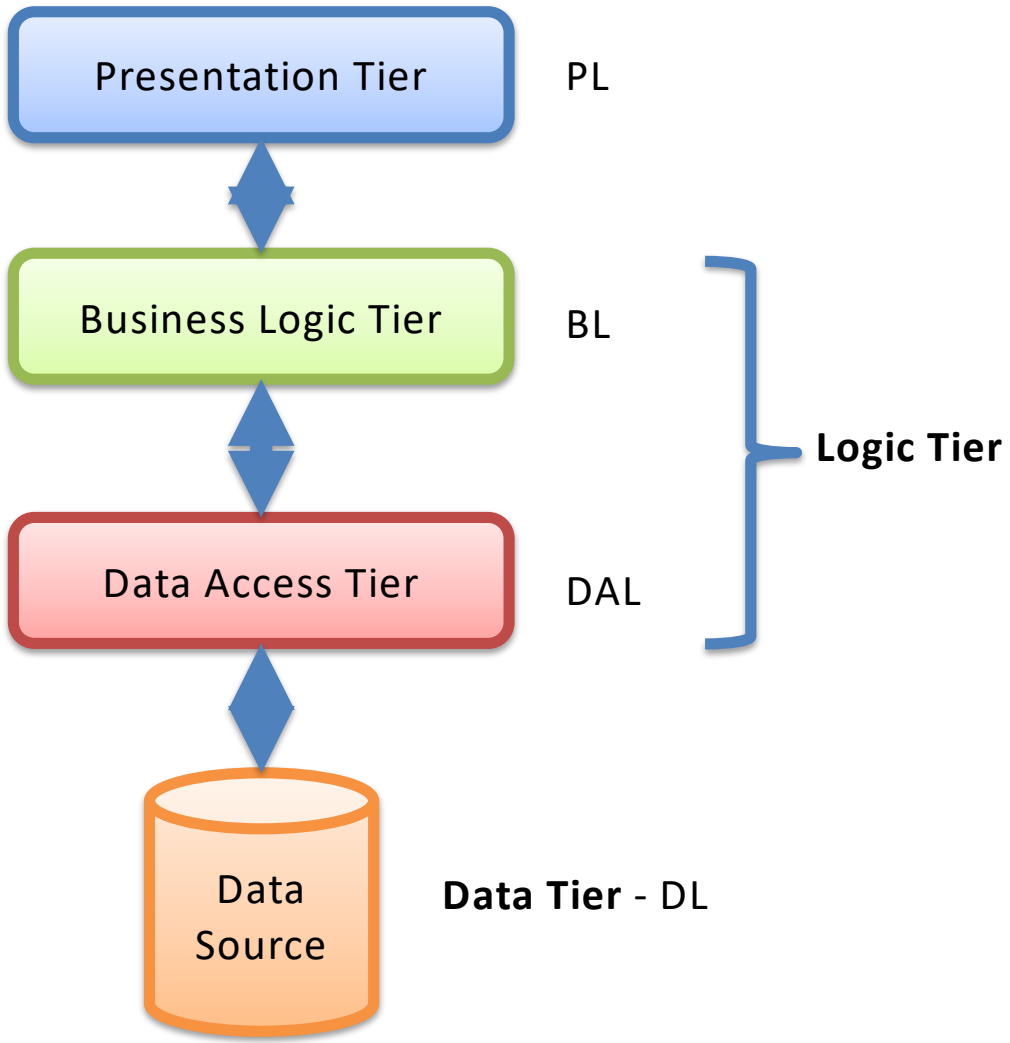
The database-centric style. Typically, the clients communicate directly with the database.



A three-tier style, in which clients do not connect directly to the database.

3-tier/layer Architecture

Note! The different layers can be on the same computer (Logic Layers) or on different Computers in a network (Physical Layers)



Why 3-Tier (N-Tier Architecture?)

- Flexible applications
- Reusable code
  - Code once, use many times
- Modularized
  - You need only to change part of the code
  - You can deploy only one part
  - You can Test only one part
  - Multiple Developers
- Different parts (Tiers) can be stored on different computers
- Different Platforms and Languages can be used
- etc.

Presentation tier

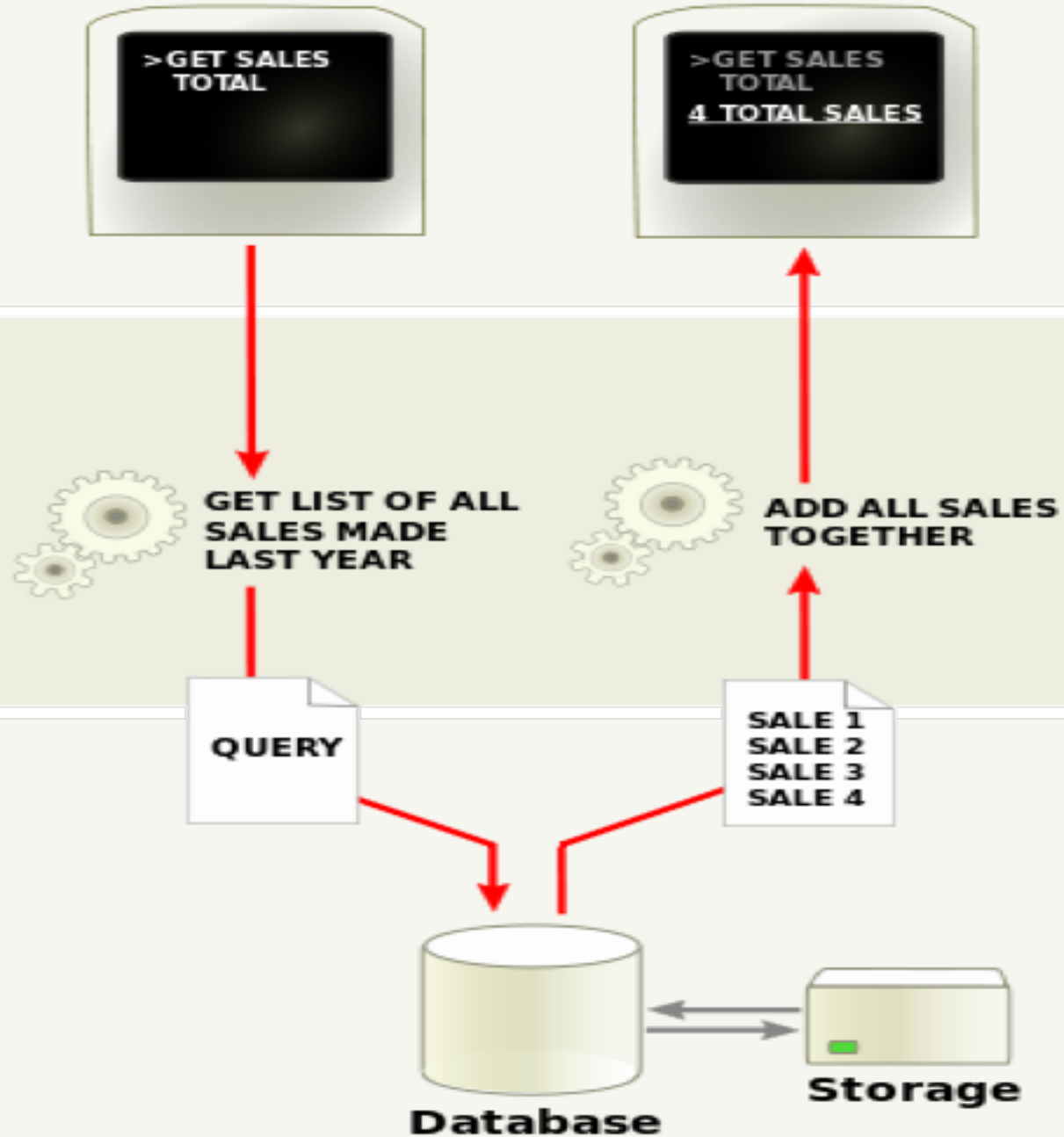
The top-most level of the application is the user interface. The main function of the interface is to translate tasks and results to something the user can understand.

Logic tier

This layer coordinates the application, processes commands, makes logical decisions and evaluations, and performs calculations. It also moves and processes data between the two surrounding layers.

Data tier

Here information is stored and retrieved from a database or file system. The information is then passed back to the logic tier for processing, and then eventually back to the user.



http://en.wikipedia.org/wiki/Multitier_architecture

3-tier/layer Architecture

Presentation Tier

- This is the topmost level of the application.
- The presentation tier displays information related to such services as browsing merchandise, purchasing and shopping cart contents.
- It communicates with other tiers by which it puts out the results to the browser/client tier and all other tiers in the network.
- In simple terms it is a layer which users can access directly such as a web page, or an operating systems GUI

Application tier (business logic, logic tier, data access tier, or middle tier)

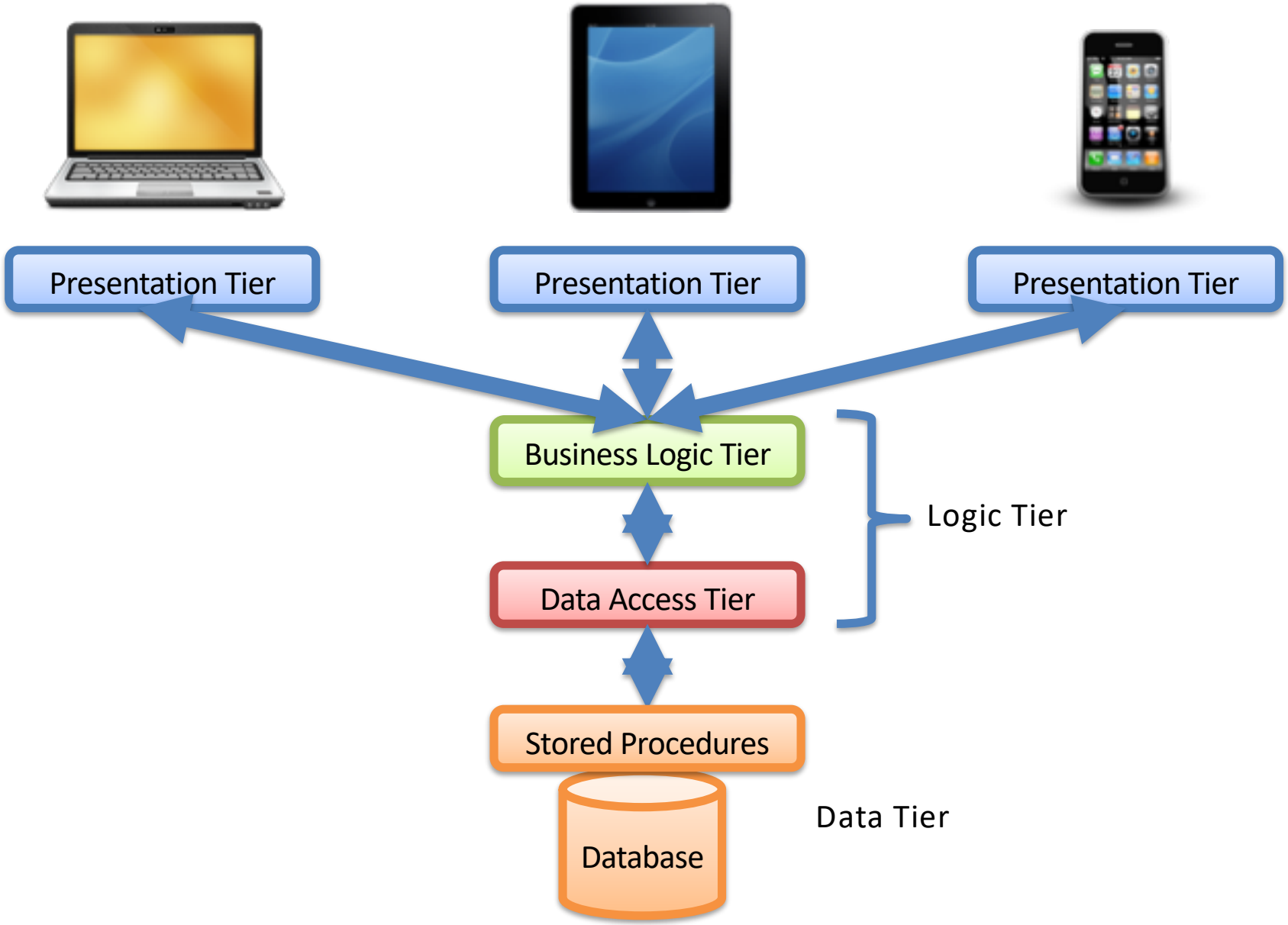
- The logical tier is pulled out from the presentation tier and, as its own layer.
- It controls an application's functionality by performing detailed processing.

Data tier

- This tier consists of database servers. Here information is stored and retrieved.
- This tier keeps data neutral and independent from application servers or business logic.
- Giving data its own tier also improves scalability and performance.

http://en.wikipedia.org/wiki/Multitier_architecture

3-tier Architecture



3-tier Architecture

Presentation Tier

Presentation Tier

Presentation Tier

Business Logic Tier

Logic Tier

Data Access Tier

Stored Procedures

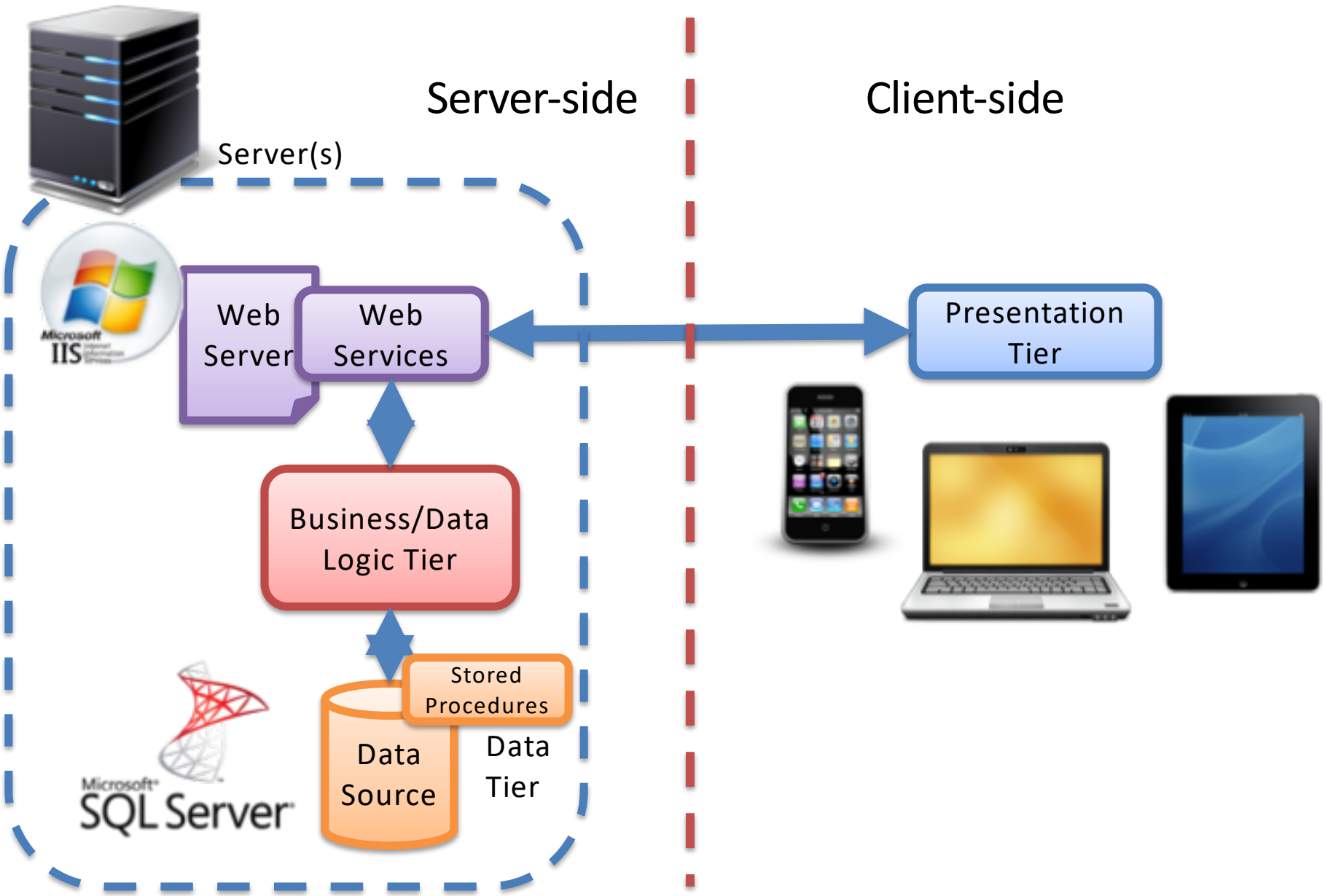
Database

Data Tier

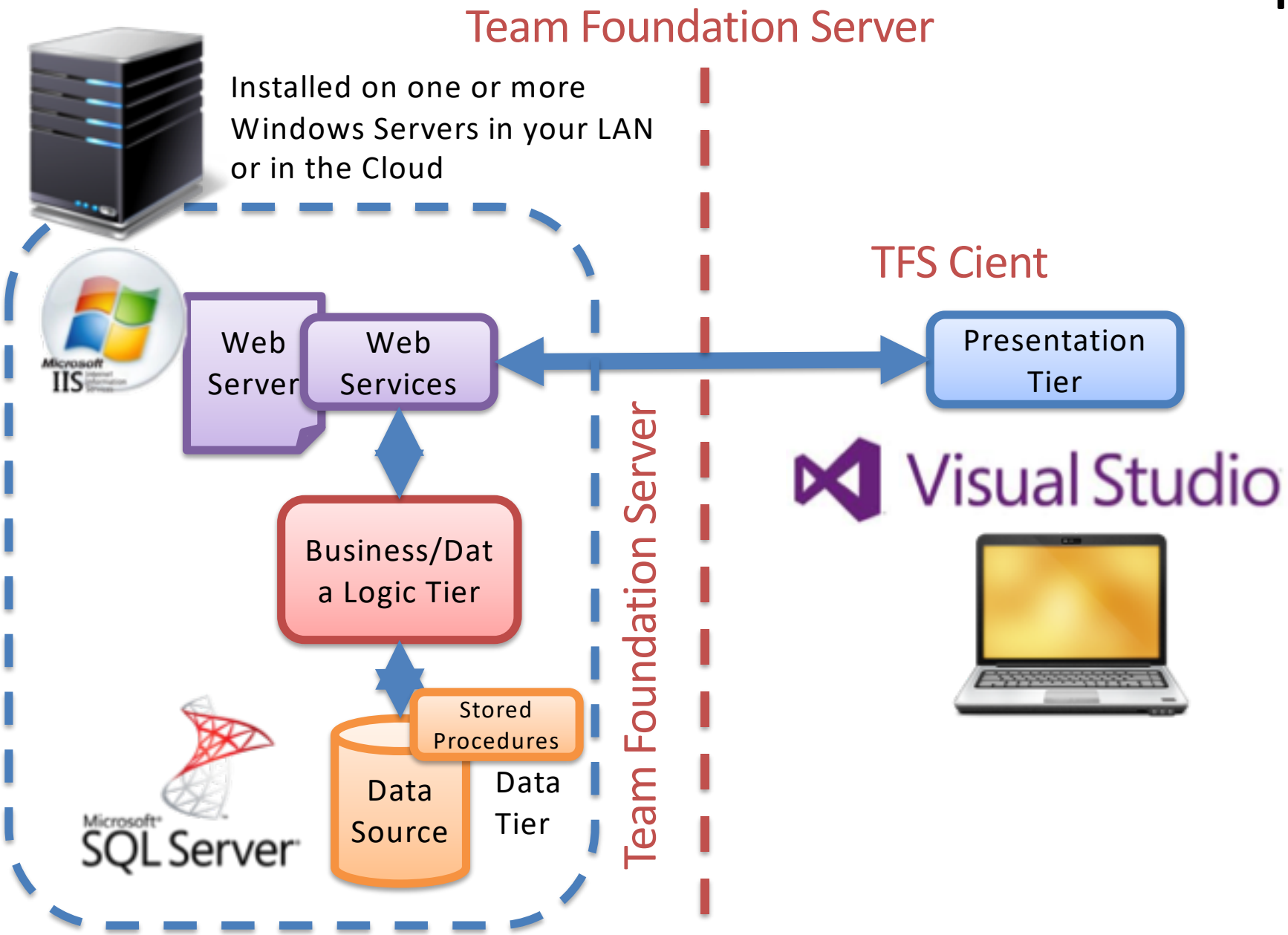
Different Devices can share the same Business and Data Access Code

The different Tiers can be physical or logical

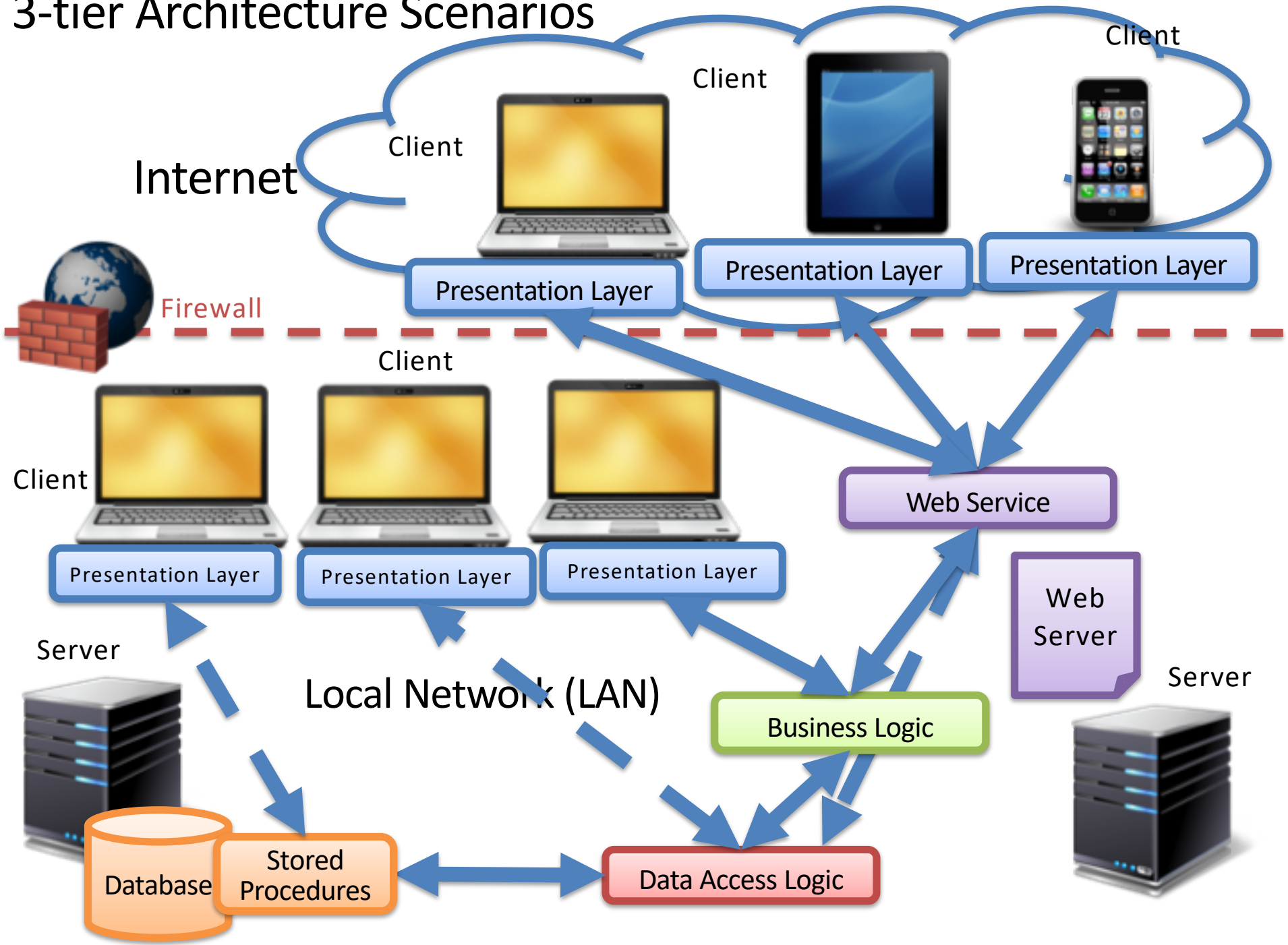
3-tier + WebService Architecture - Example



3-tier + WebService Architecture - Example



3-tier Architecture Scenarios

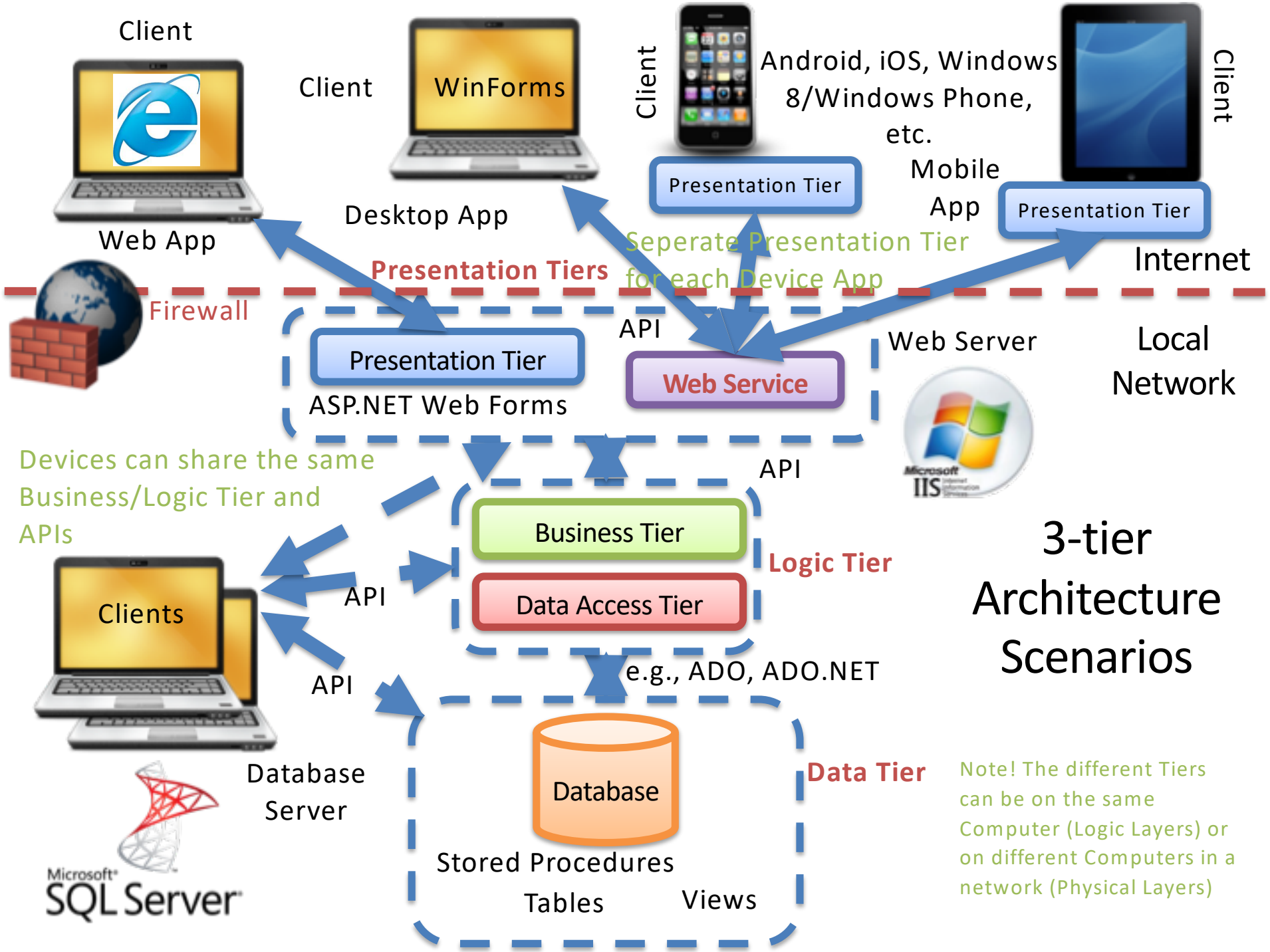


Exercises

1. Create **Data Tier** (Database)
2. Create **Logic Tier** (Database Communication Logic)

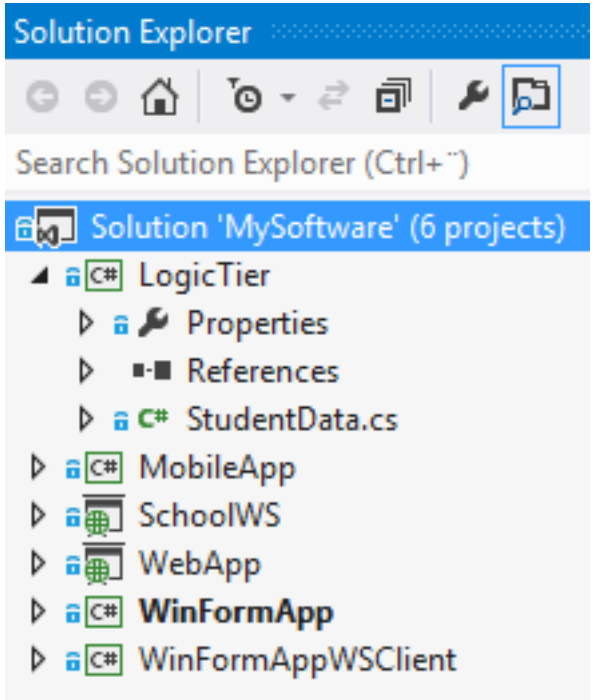
Create **Presentation Tier** (User Interface Logic):

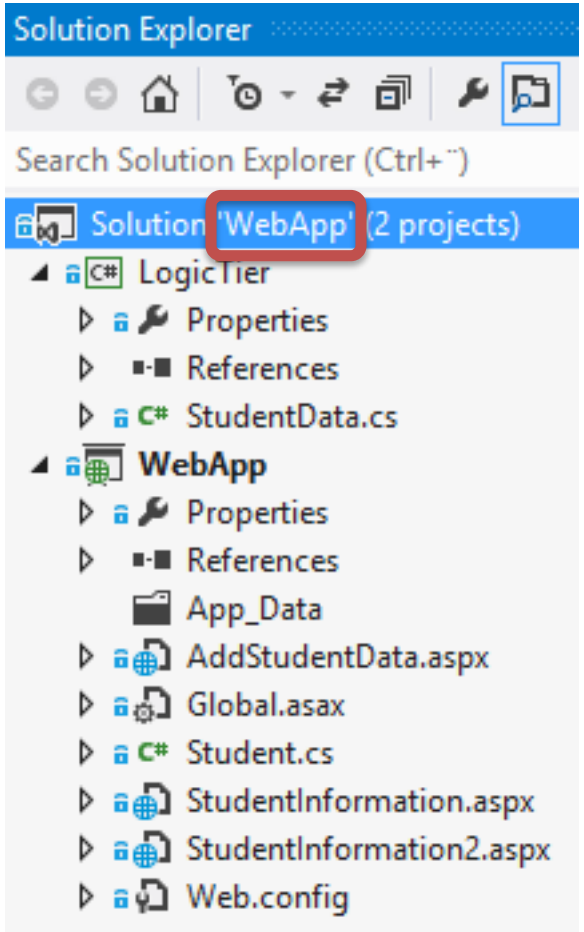
3. **WebApp**: Using ASP.NET Web Forms (WS normally not needed)
4. **Desktop App**: Using WinForms
   A. Without Web Services (We assume the App will be used only in the LAN and that we have direct access to the Database)
   B. With Web Services (We assume the App should be used on Internet outside the Firewall without direct DB access)


Client
Web App
Client
WinForms
Desktop App
Client
Android, iOS, Windows 8/Windows Phone, etc.
Mobile App
Client
Presentation Tier
Presentation Tier
Seperate Presentation Tier for each Device App
Internet
Presentation Tiers
Firewall
API
Presentation Tier
ASP.NET Web Forms
Web Service
Web Server
Local Network
Devices can share the same Business/Logic Tier and APIs
API
Business Tier
Logic Tier
Data Access Tier
Clients
API
API
e.g., ADO, ADO.NET
Database Server
Database
Data Tier
Stored Procedures
Tables
Views
3-tier Architecture Scenarios
Note! The different Tiers can be on the same Computer (Logic Layers) or on different Computers in a network (Physical Layers)

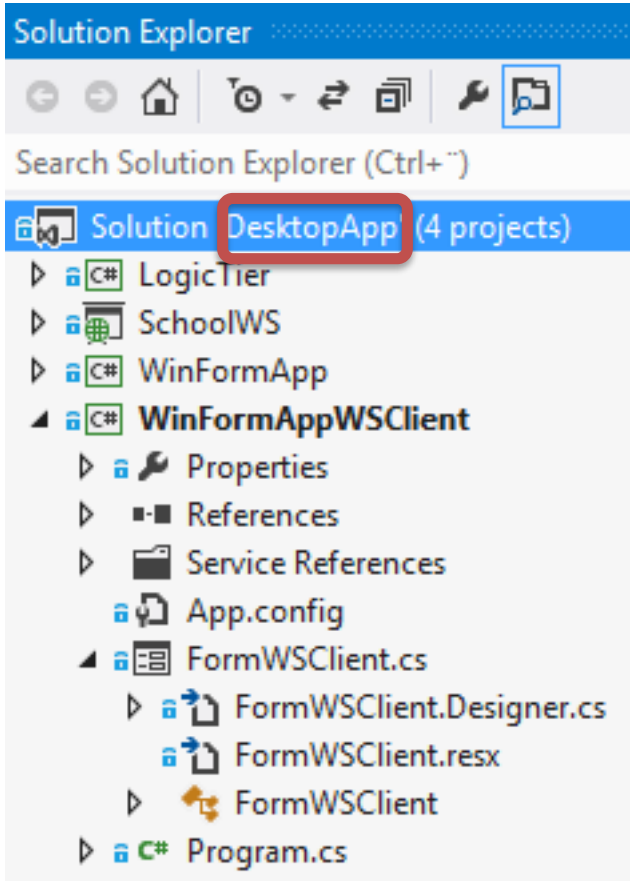
Visual Studio Projects

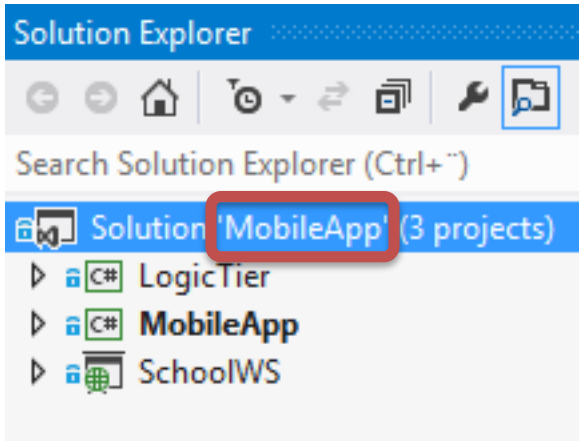
Solution with all Projects (Logic Tier, Web Service, Desktop App, Web App, Mobile App)



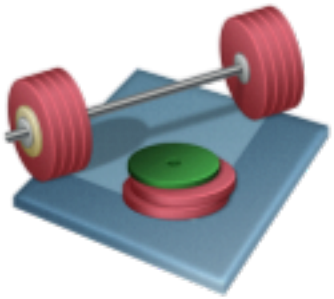


Solution with Projects used by Web App (Logic Tier, Web App)





Data Tier



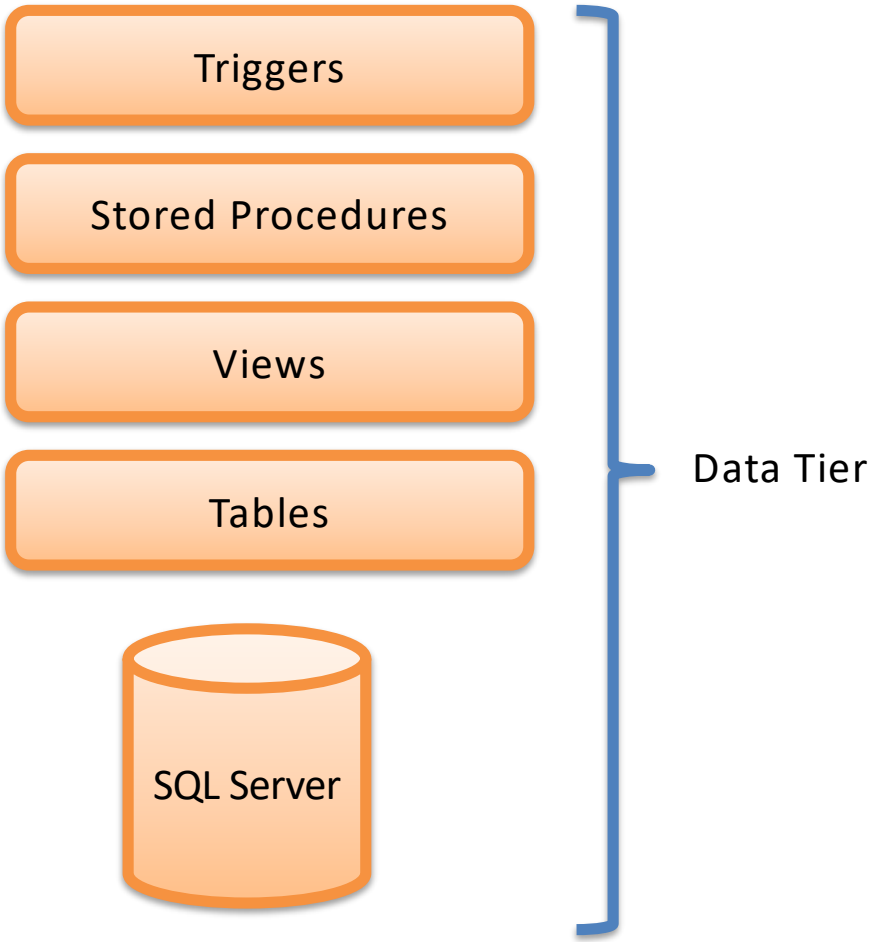
We are going to create the Database / Data Layer/Tier, including:

1. Tables
2. Views
3. Stored Procedures
4. Triggers
5. Script for some “Dummy” Data

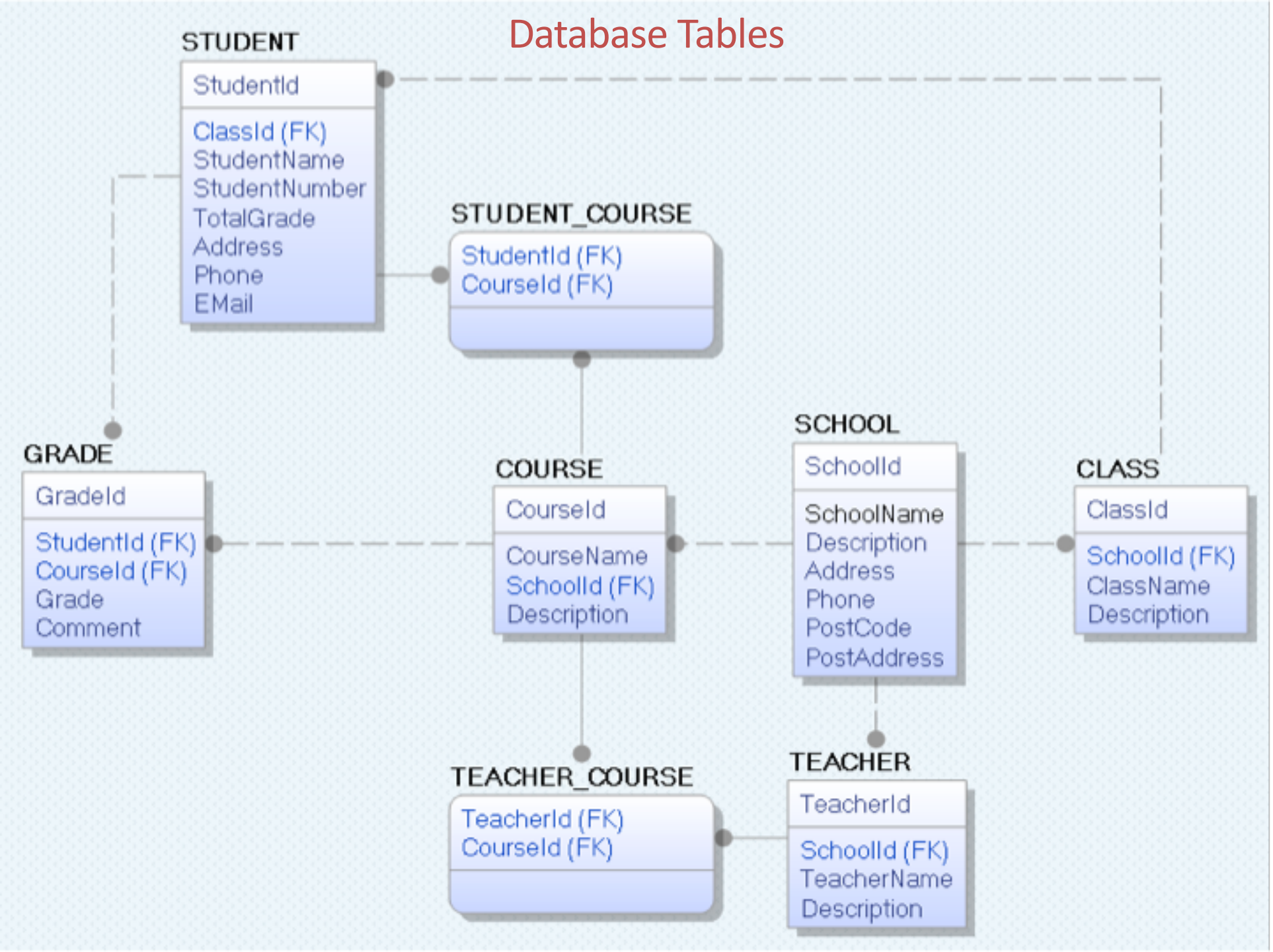
Note! Install them in this order

Download Zip Files with Tables, Views, Stored Procedures and Triggerse in order to create the Data Tier in SQL Server (The ZIP File is located on the same place as this File)

Data Tier


Triggers
Stored Procedures
Views
Tables
SQL Server
Data Tier

Database Tables



Execute the different Scripts inside SQL Server Management Studio

School Tables Script.sql - macwin8.SCHOOL (MACWIN8\Hans-Petter (53)) - Microsoft SQL Server Management Studio

File Edit View Query Project Debug Tools Window Help

New Query | SCHOOL | Execute | Debug

Object Explorer

Connect

- macwin8 (SQL Server 11.0.2100 - MACWIN8\Hans-Petter)
  - Databases
    - System Databases
    - SCHOOL
      - Database Diagrams
      - Tables
        - System Tables
        - FileTables
        - dbo.CLASS
        - dbo.COURSE
        - dbo.GRADE
          - Columns
          - Keys
          - Constraints
          - Triggers
            - CalcAvgGrade
          - Indexes
          - Statistics
        - dbo.SCHOOL
        - dbo.STUDENT
        - dbo.STUDENT_COURSE
        - dbo.TEACHER
        - dbo.TEACHER_COURSE
      - Views
        - System Views
        - dbo.CourseData
        - dbo.StudentData
      - Synonyms
      - Programmability
        - Stored Procedures
          - System Stored Procedures
          - dbo.StudentGrade
            - Parameters
              - @StudentId (int, Input, No default)
              - @CourseId (int, Input, No default)
              - @Grade (int, Input, No default)
              - Returns integer
        - Functions
        - Database Triggers
        - Assemblies
        - Types
        - Rules
        - Defaults
        - Sequences
      - Service Broker

School Tables Scrip...8\Hans-Petter (53)) | SQLQuery1.sql - ma...8\Hans-Petter (52))

```
/*-----------------------------------------------------------------------------------------
Last Updated Date:  2011.04.01
Last Updated By:    hans.p.halvorsen@hit.no
Description:        Lab Work - School Tables

(C) Hans-Petter Halvorsen
-----------------------------------------------------------------------------------------

/*
SCHOOL
CLASS
COURSE
TEACHER
TEACHER_COURSE
STUDENT
STUDENT_COURSE
GRADE
*/

if not exists (select * from dbo.sysobjects where id = object_id(N'[SCHOOL]') and OBJECTPROPERTY(id, N'IsUserTable') = 1)
CREATE TABLE [SCHOOL]
(
    [SchoolId] [int] IDENTITY(1, 1) NOT NULL PRIMARY KEY,
    [SchoolName] [varchar](50) NOT NULL UNIQUE,
    [Description] [varchar](1000) NULL,
    [Address] [varchar](50) NULL,
    [Phone] [varchar](50) NULL,
    [PostCode] [varchar](50) NULL,
    [PostAddress] [varchar](50) NULL,

)
GO


if not exists (select * from dbo.sysobjects where id = object_id(N'[CLASS]') and OBJECTPROPERTY(id, N'IsUserTable') = 1)
CREATE TABLE [CLASS]
(
    [ClassId] [int] IDENTITY(1, 1) NOT NULL PRIMARY KEY,
    [SchoolId] [int] NOT NULL FOREIGN KEY REFERENCES [SCHOOL] ([SchoolId]),
    [ClassName] [varchar](50) NOT NULL,
    [Description] [varchar](1000) NULL,

)
GO


if not exists (select * from dbo.sysobjects where id = object_id(N'[COURSE]') and OBJECTPROPERTY(id, N'IsUserTable') = 1)
CREATE TABLE [COURSE]
(
    [CourseId] [int] IDENTITY(1, 1) NOT NULL PRIMARY KEY,
```

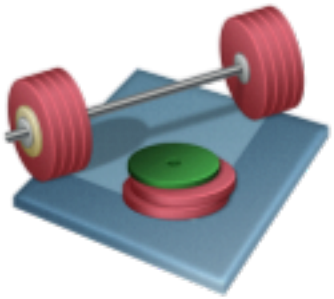
100 %

Connected. (1/1) | macwin8 (11.0 RTM) | MACWIN8\Hans-Petter (53) | SCHOOL | 00:00:00 | 0 rows

Ready | Ln 1 | Col 1 | Ch 1 | INS

You are finished with the Exercise

Create Logic Tier



ASP.NET Web Forms

Presentation Tier

WinForms

Presentation Tier

Windows Store App

Presentation Tier

Visual Studio

Logic Tier

Purpose:

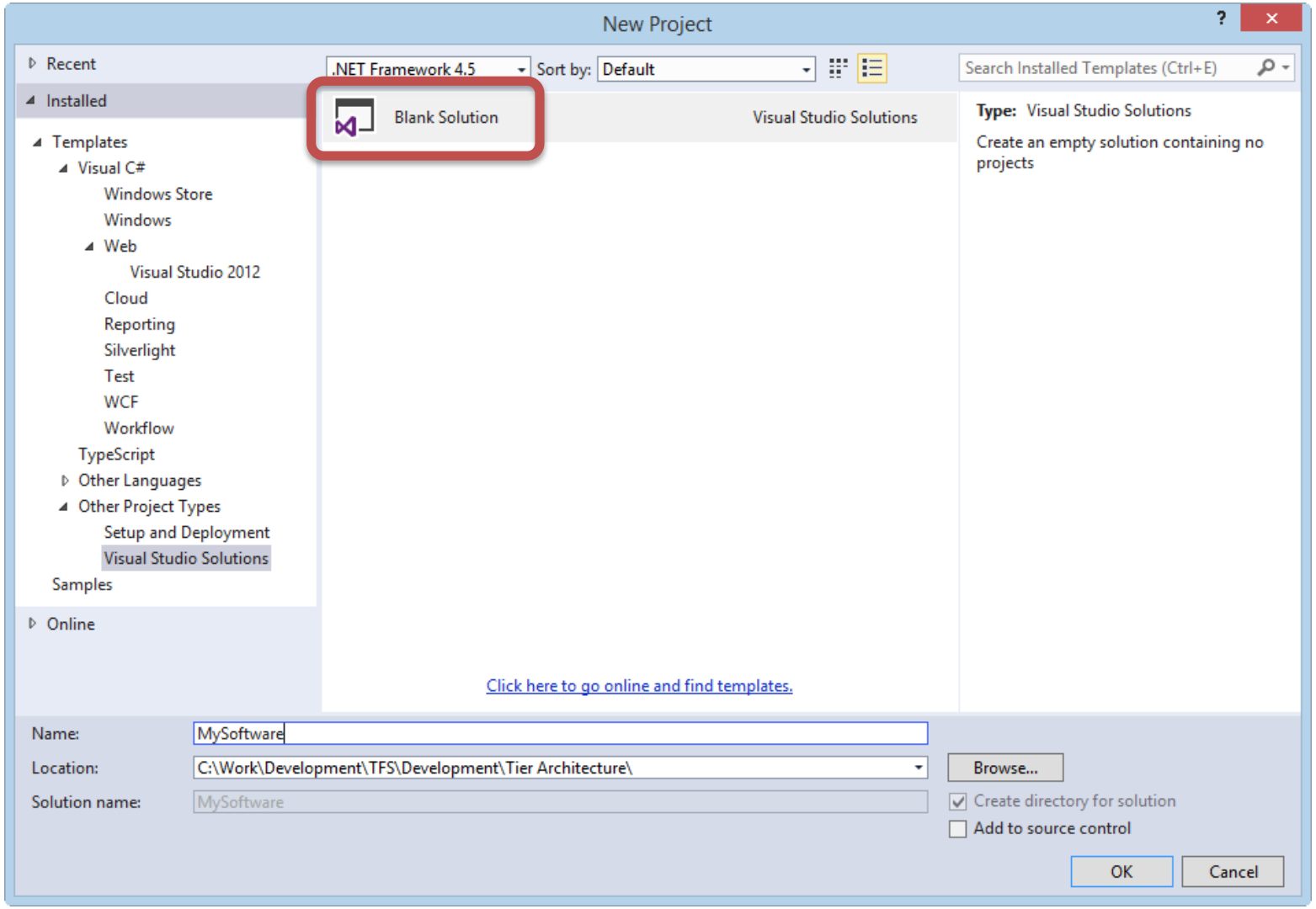
- All the Apps should/could share the same Logic Tier
- To make your Apps easier to maintain and extend
- etc.

Microsoft SQL Server

Data Tier

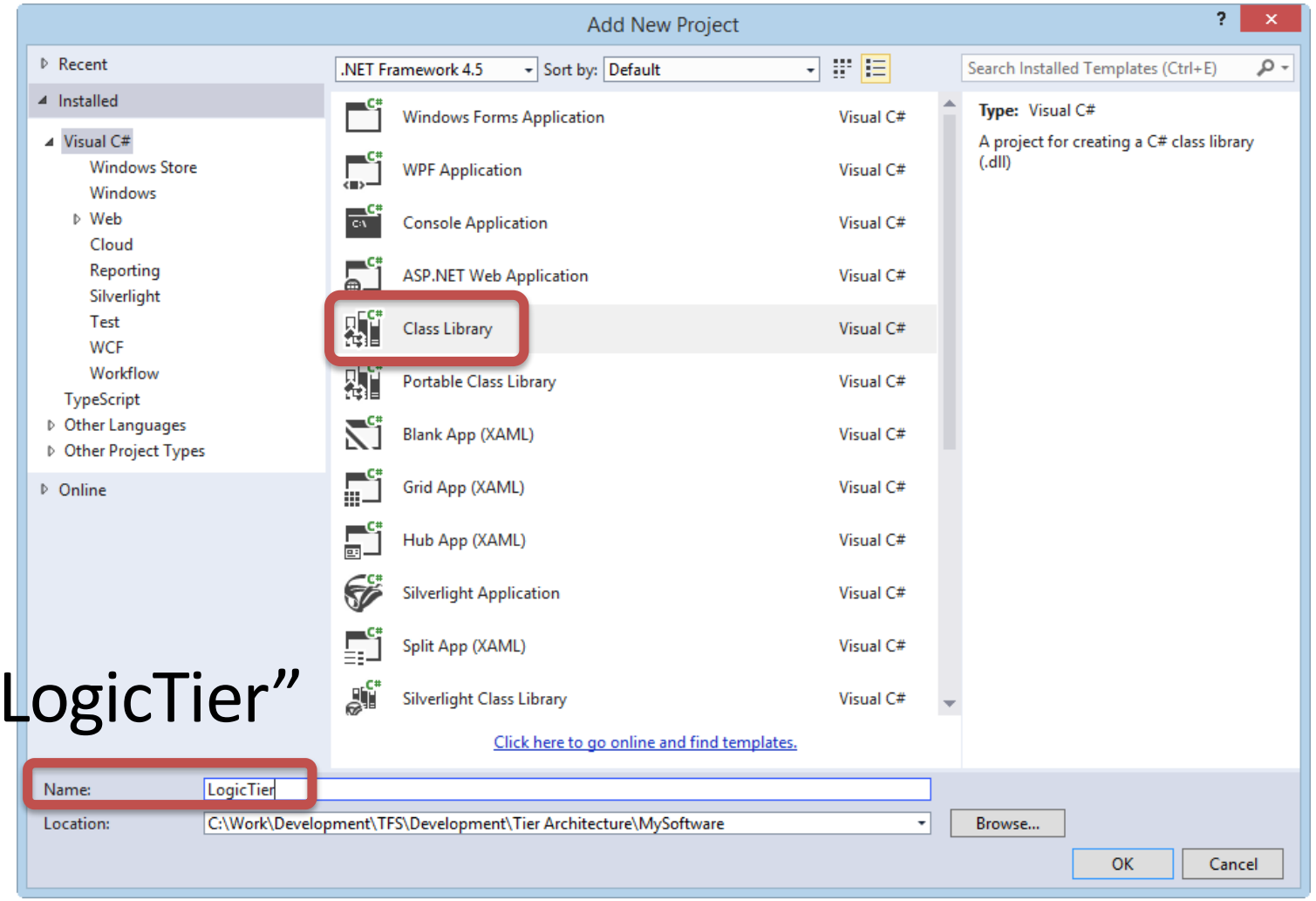
Database

Create an Empty (Blank) **Solution** in Visual Studio



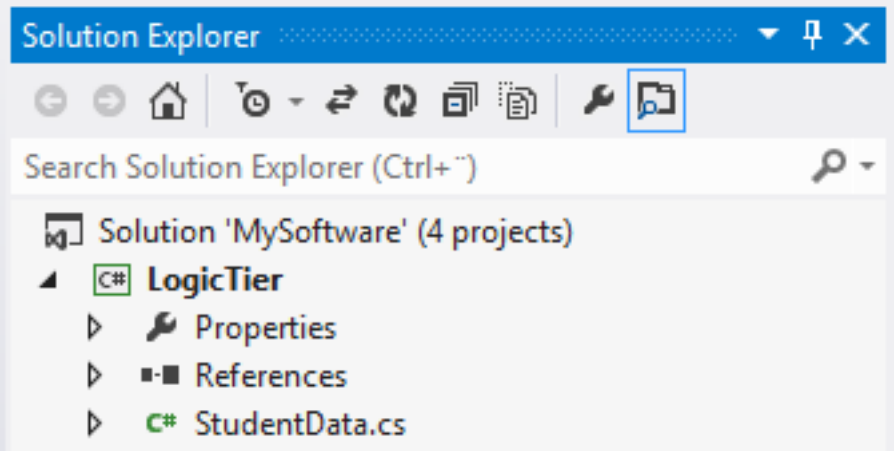
Add **Project** for Logic Tier (Data Access)

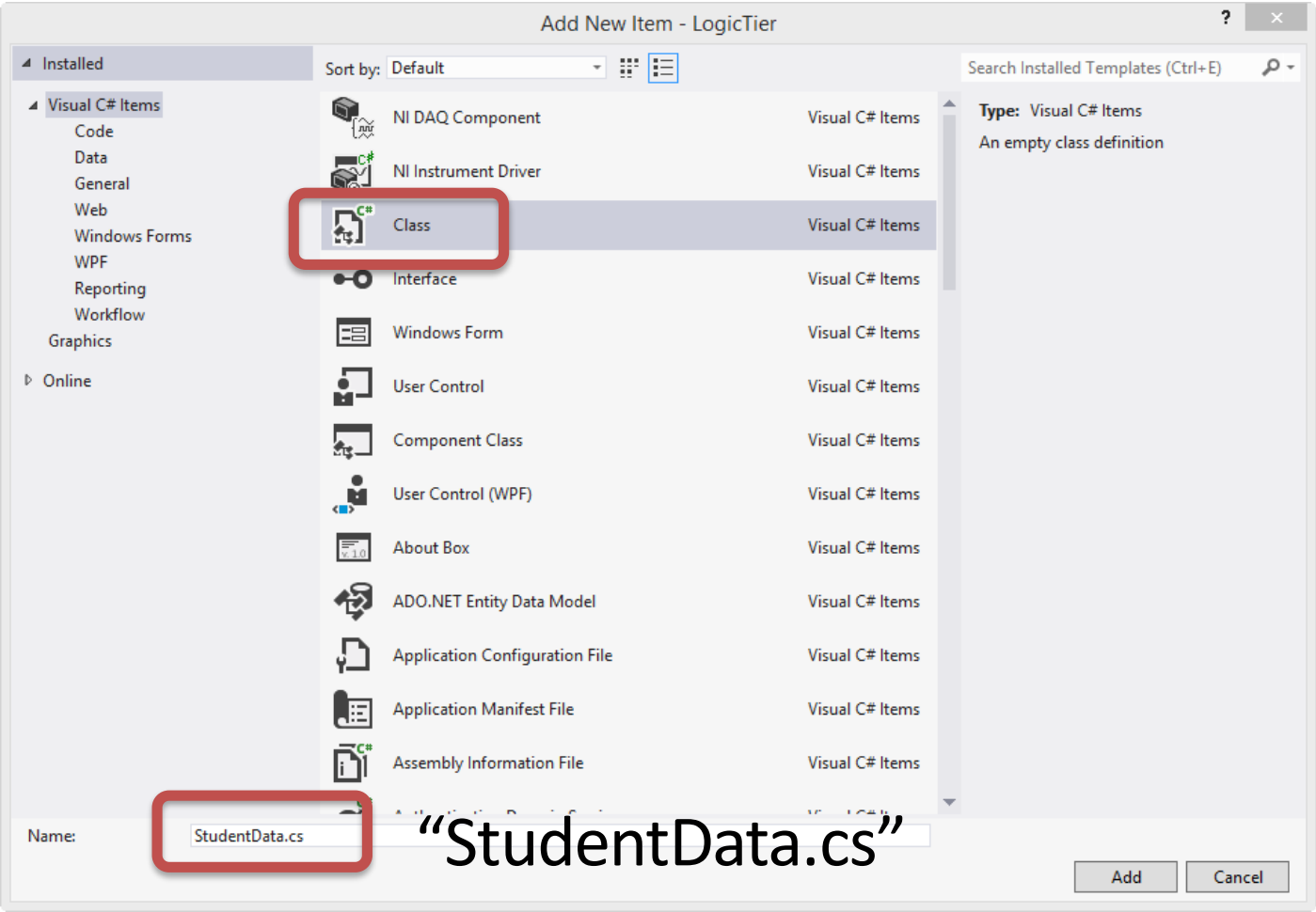
Select a "**Class Library**" Project



"LogicTier"

Add a New **Class** to the Project (“StudentData.cs”)





“StudentData.cs”

Create the **Code**, e.g., like this ("StudentData.cs"):

StudentData.cs

Tuc.School.LogicTier.StudentData | GetStudentDB(string connectionString)

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

using System.Data.SqlClient;
using System.Data.SqlTypes;
using System.Data;

namespace Tuc.School.LogicTier
{
    public class StudentData
    {

        public DataSet GetStudentDB(string connectionString)
        {

            string selectSQL = "select StudentName, StudentNumber, SchoolName, ClassName, Grade from StudentData order by StudentName";

            // Define the ADO.NET objects.
            SqlConnection con = new SqlConnection(connectionString);

            SqlDataAdapter da = new SqlDataAdapter(selectSQL, con);

            DataSet ds = new DataSet();
            da.Fill(ds);

            return ds;

        }

    }
}
```

Create your own Namespace

A View that collects data from several tables

Improvements: Use Try... Catch ...

You should test the SQL Query in the **SQL Server Management Studio** first

SQLQuery1.sql - macwin8.SCHOOL (MACWIN8\Hans-Petter (52))* - Microsoft SQL Server Management Studio

File Edit View Query Project Debug Tools Window Help

New Query | SCHOOL | Execute | Debug

Object Explorer

Connect

- macwin8 (SQL Server 11.0.2100 - MACWIN8\Har
  - Databases
    - System Databases
    - SCHOOL
      - Database Diagrams
      - Tables
        - System Tables
        - FileTables
        - dbo.CLASS
        - dbo.COURSE
        - dbo.GRADE
        - dbo.SCHOOL
        - dbo.STUDENT
        - dbo.STUDENT_COURSE
        - dbo.TEACHER
        - dbo.TEACHER_COURSE
      - Views
        - System Views
        - dbo.CourseData
        - dbo.StudentData
      - Synonyms
      - Programmability
      - Service Broker
      - Storage
      - Security
  - Security
  - Server Objects
  - Replication

SQLQuery1.sql - ma...8\Hans-Petter (52))*

```
select StudentName, StudentNumber, SchoolName, ClassName, Grade from StudentData order by StudentName
```

100 %

Results | Messages

	StudentName	StudentNumber	SchoolName	ClassName	Grade
1	Barak Obama	3333	TUC	SCE2	0
2	Jens Stoltenberg	2222	TUC	SCE1	5
3	John Cleese	1111	TUC	SCE1	4
4	Kurt Nilsen	4444	TUC	SCE2	3

Query executed successfully. | macwin8 (11.0 RTM) | MACWIN8\Hans-Petter (52) | SCHOOL | 00:00:03 | 4 rows

Ready | Ln 1 | Col 52 | Ch 52 | INS

Code (“StudentData.cs”):

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

namespace Tuc.School.LogicTier
{
    public class StudentData
    {
        public DataSet GetStudentDB(string connectionString)
        {
            string selectSQL = "select StudentName, StudentNumber, SchoolName, ClassName,
                   Grade from StudentData order by StudentName";

            // Define the ADO.NET objects.
            SqlConnection con = new SqlConnection(connectionString);

            SqlDataAdapter da = new SqlDataAdapter(selectSQL, con);

            DataSet ds = new DataSet();
            da.Fill(ds);

            return ds;

        }
    }
}
```

Create a proper name for the **Assembly** (.dll File)

Right-click on the Project in the Solution Explorer and select Properties

LogicTier

Application
Build
Build Events
Debug
Resources
Services
Settings
Reference Paths
Signing
Code Analysis

Configuration: N/A Platform: N/A

Assembly name: Tuc.School.LogicTier
Default namespace: Tuc.School.LogicTier

Target framework: .NET Framework 4.5
Output type: Class Library

Startup object: (Not set)
Assembly Information...

Resources
Specify how application resources will be managed:

Icon and manifest
A manifest determines specific settings for an application. To embed a custom manifest, first add it to your project and then select it from the list below.
Icon: (Default Icon)
Manifest: Embed manifest with default settings

Resource file:

Then Build your Project (hopefully with no errors)

Debug

File Home Share View

« TFS ▸ Development ▸ Tier Architecture ▸ MySoftware ▸ LogicTier ▸ bin ▸

Favorites
Desktop
Downloads
Recent places

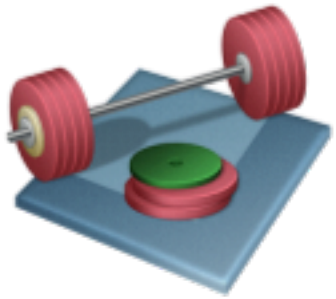
Name Date modifi

Tuc.School.LogicTier.dll 2014-02-26 4
Tuc.School.LogicTier.pdb 2014-02-26 4

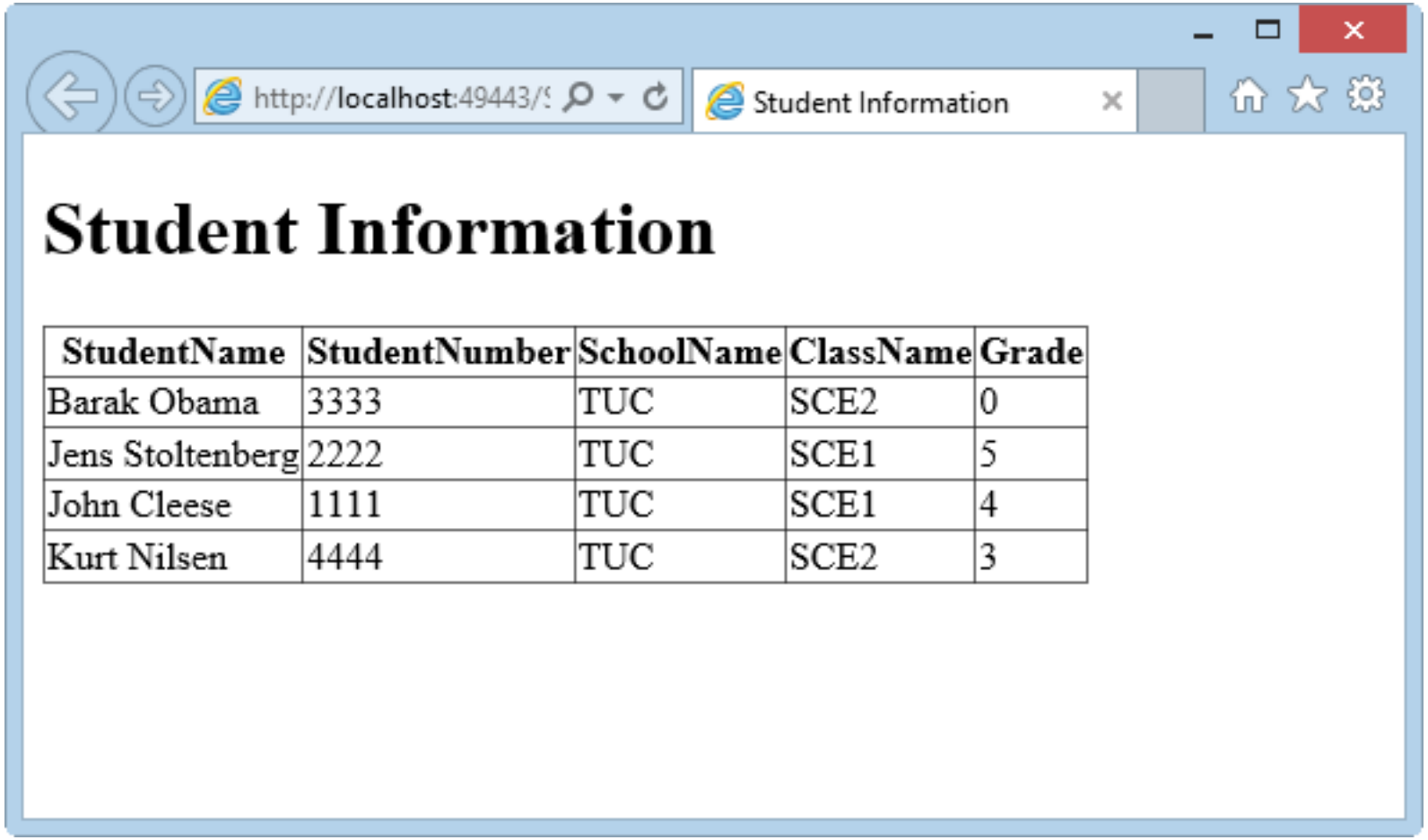
This will be the Assembly for your Logic Tier, that can be imported and used in other projects.
Create once – use it many times!!

You are finished with the Exercise

Presentation Layer
Web App: ASP.NET WebForms



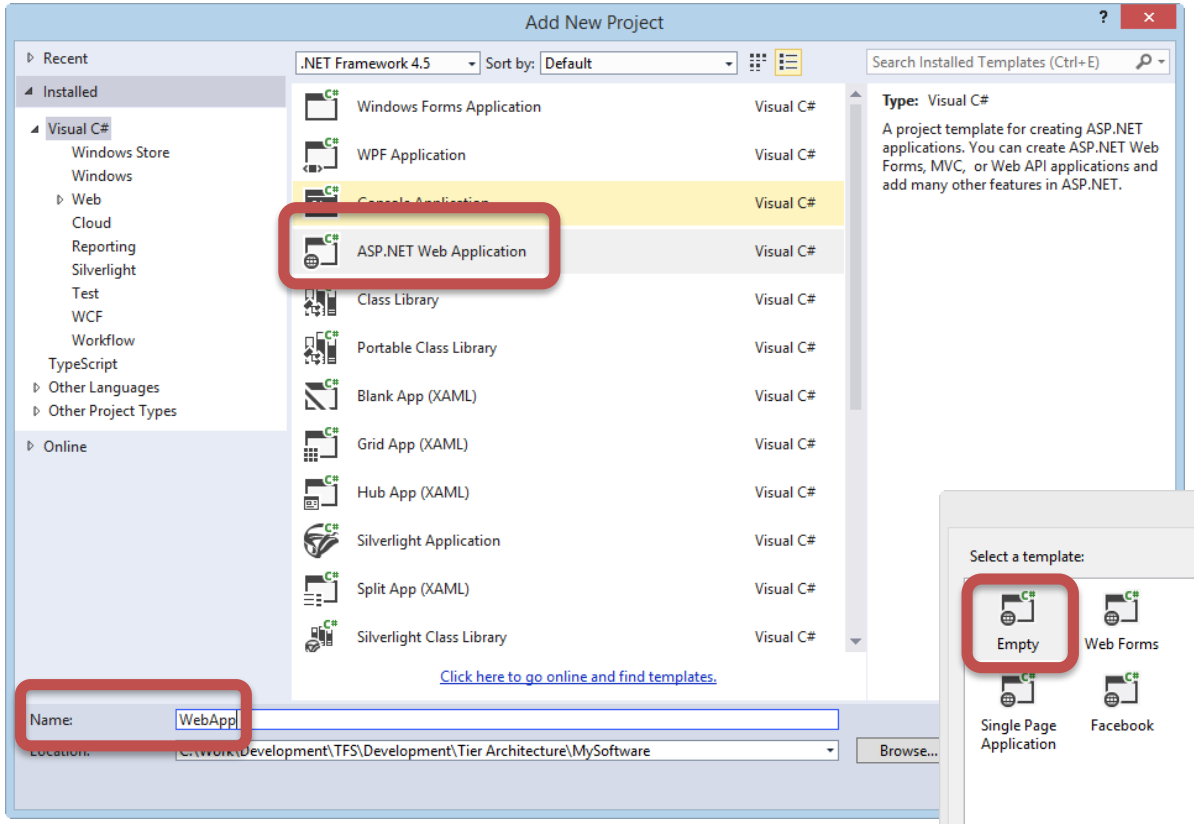
We will create a WebForm like this where the data comes from our Logic Tier

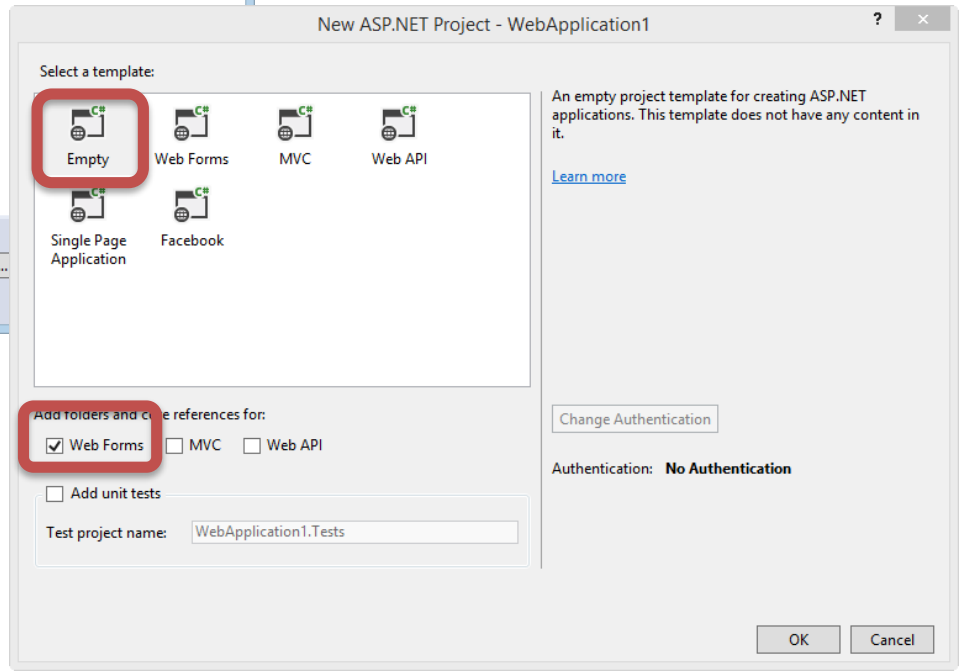


Student Information

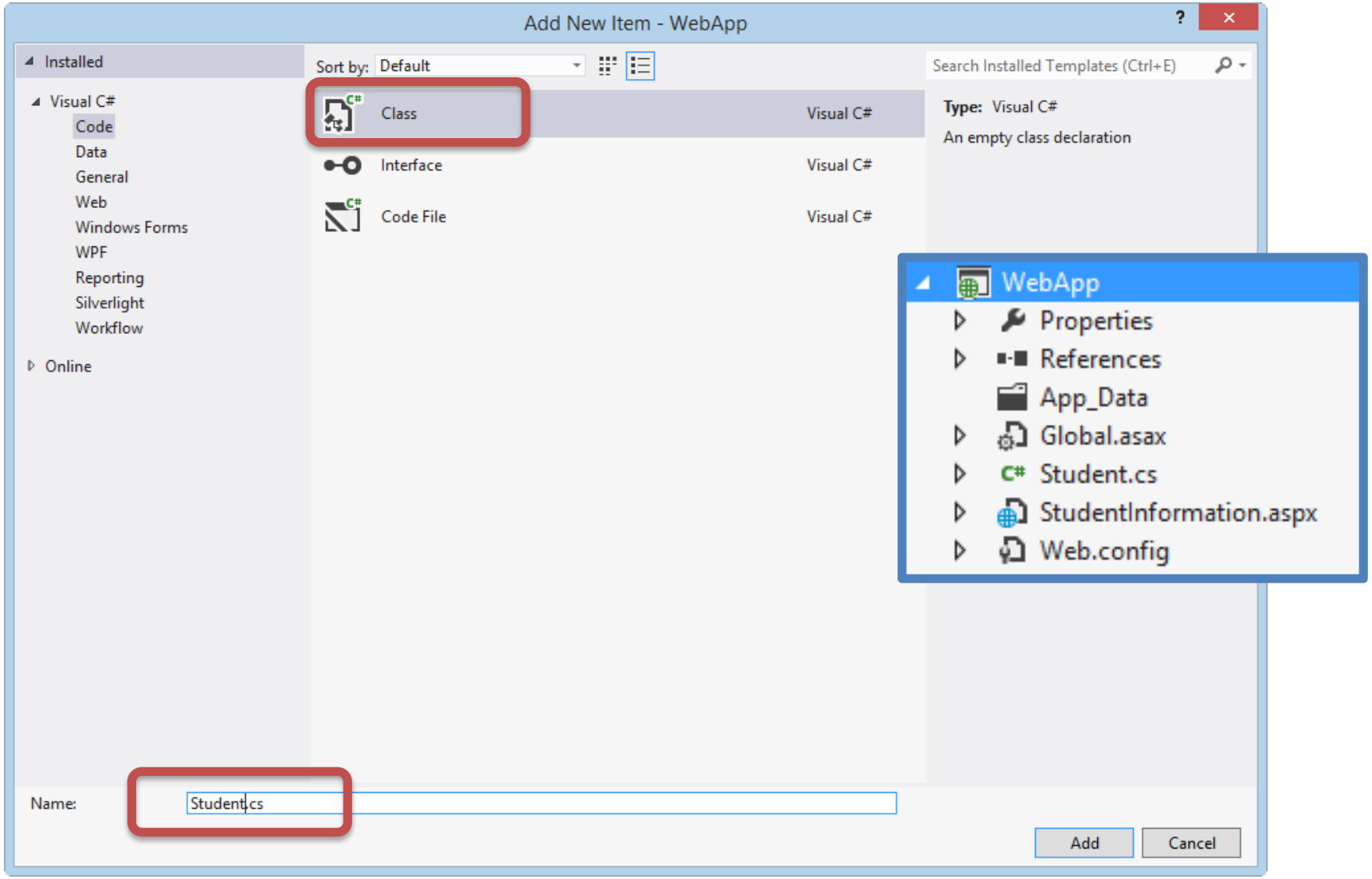
StudentName	StudentNumber	SchoolName	ClassName	Grade
Barak Obama	3333	TUC	SCE2	0
Jens Stoltenberg	2222	TUC	SCE1	5
John Cleese	1111	TUC	SCE1	4
Kurt Nilsen	4444	TUC	SCE2	3

Add Project for Presentation Tier (**ASP.NET WebForm**)





Add a New **Class** (“Student.cs”)



Add Code ("Student.cs")

Note! This is our Logic Tier

Add a Reference to the Assembly in the Logic Tier

Student.cs

Tuc.School.WebApp.Student

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

using Tuc.School.LogicTier;

namespace Tuc.School.WebApp
{
    public class Student
    {

        public DataSet GetStudent(
        {
            StudentData studentDat

            return studentData.Get

        }


    }
}
```

References

Log

Mic

Syst

Syst

Syst

Syst

System.Data

Add Reference...

Add Service Reference...

Manage NuGet Packages...

Scope to This

New Solution Explorer View

Reference Manager - WebApp

Assemblies

Solution

Projects

COM

Browse

Search Solution (Ctrl+E)

Name	Path
LogicTier	C:\Work\Development\TFS

Name:
LogicTier

Browse... OK Cancel

Code for “Student.cs”

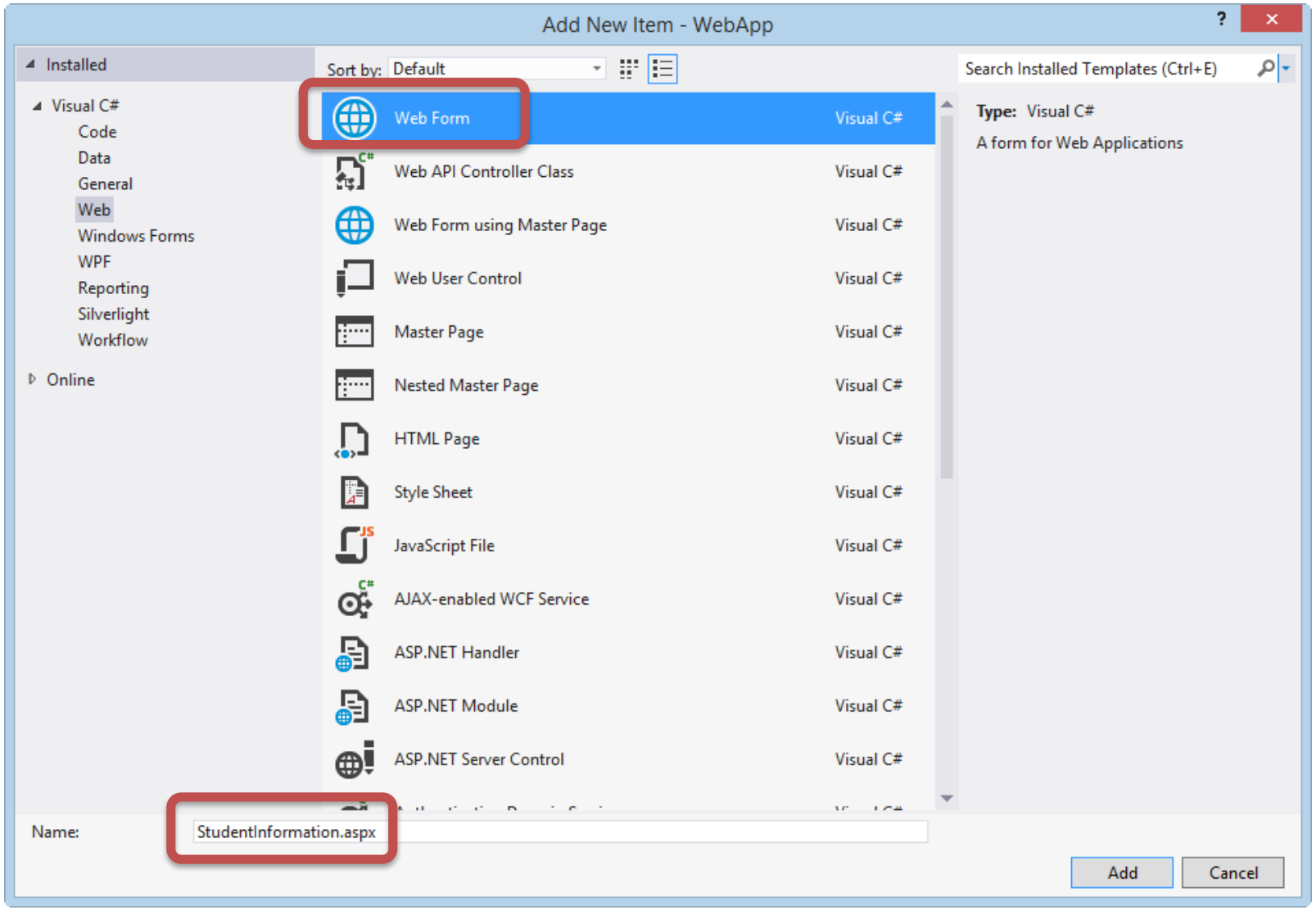
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;

using System.Data;

using Tuc.School.LogicTier;


namespace Tuc.School.WebApp
{

    public class Student
    {

        public DataSet GetStudent(string connectionString)
        {

            StudentData studentData = new StudentData();

            return studentData.GetStudentDB(connectionString);

        }
    }
}
```

Since we are using the DataSet Class

Our Logic Tier

Get Dat from our Logic Tier

Add a New **WebForm** (StudentInformation.aspx)



Create WebForm Page (“**StudentInformation.aspx**”)

StudentInformation.aspx

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="StudentInformation.aspx.cs" Inherits="WebApp.StudentInformation" %>

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title>Student Information</title>
</head>
<body>
    <form id="form1" runat="server">
    <div>

        <h1>Student Information</h1>

    </div>
        <asp:GridView ID="gridStudentData" runat="server">
        </asp:GridView>
    </form>
</body>
</html>
```

100 %

body

Student Information

Column0	Column1	Column2
abc	abc	abc
abc	abc	abc
abc	abc	abc
abc	abc	abc
abc	abc	abc

GridView (Drag & Drop from Toolbox or create code in .aspx file)

HTML /ASPX Code (“**StudentInformation.aspx**”)

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="StudentInformation.aspx.cs" Inherits="WebApp.StudentInformation" %>

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title>Student Information</title>
</head>
<body>
    <form id="form1" runat="server">
    <div>

        <h1>Student Information</h1>

    </div>
        <asp:GridView ID="gridStudentData" runat="server">
        </asp:GridView>
    </form>
</body>
</html>
```

Code behind for the Web Form (“**StudentInformation.aspx.cs**”)

```
using System.Web.Configuration;
using Tuc.School.WebApp;

namespace WebApp
{
    public partial class StudentInformation : System.Web.UI.Page
    {

        private string connectionString =
           WebConfigurationManager.ConnectionStrings["SCHOOLConnectionString"].ConnectionString;

        protected void Page_Load(object sender, EventArgs e)
        {
            if (!IsPostBack)
            {
                FillStudentGrid();
            }
        }

        private void FillStudentGrid()
        {
            DataSet ds = new DataSet();

            Student studentList = new Student();

            ds = studentList.GetStudent(connectionString);

            gridStudentData.DataSource = ds;

            gridStudentData.DataBind();
        }
    }
}
```

Note!

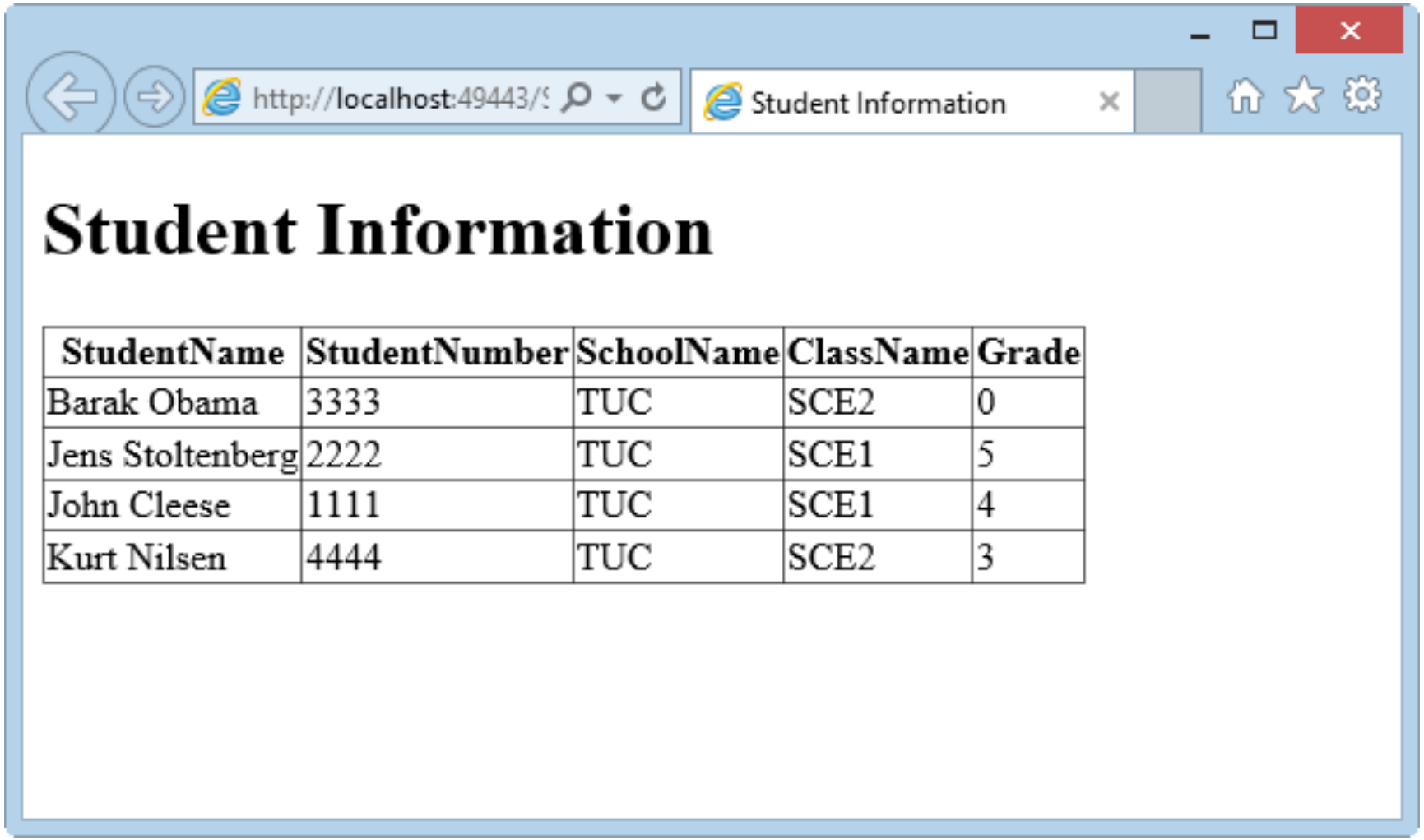
Web.Config

Store the “ConnectionString” for your Database in “**Web.Config**”

```
<?xml version="1.0" encoding="utf-8"?>
<!--
  For more information on how to configure your ASP.NET application, please visit
  http://go.microsoft.com/fwlink/?LinkId=169433
  -->



<configuration>

  <connectionStrings>
    <add name="SCHOOLConnectionString" connectionString="Data Source=macwin8;Initial
Catalog=SCHOOL;Persist Security Info=True;User ID=sa;Password=xxxxxx"
      providerName="System.Data.SqlClient" />
  </connectionStrings>

  <system.web>
    <compilation debug="true" targetFramework="4.5" />
    <httpRuntime targetFramework="4.5" />
  </system.web>
</configuration>
```

Then you can easly switch Databsse without changing the Code!!

Test your Web App

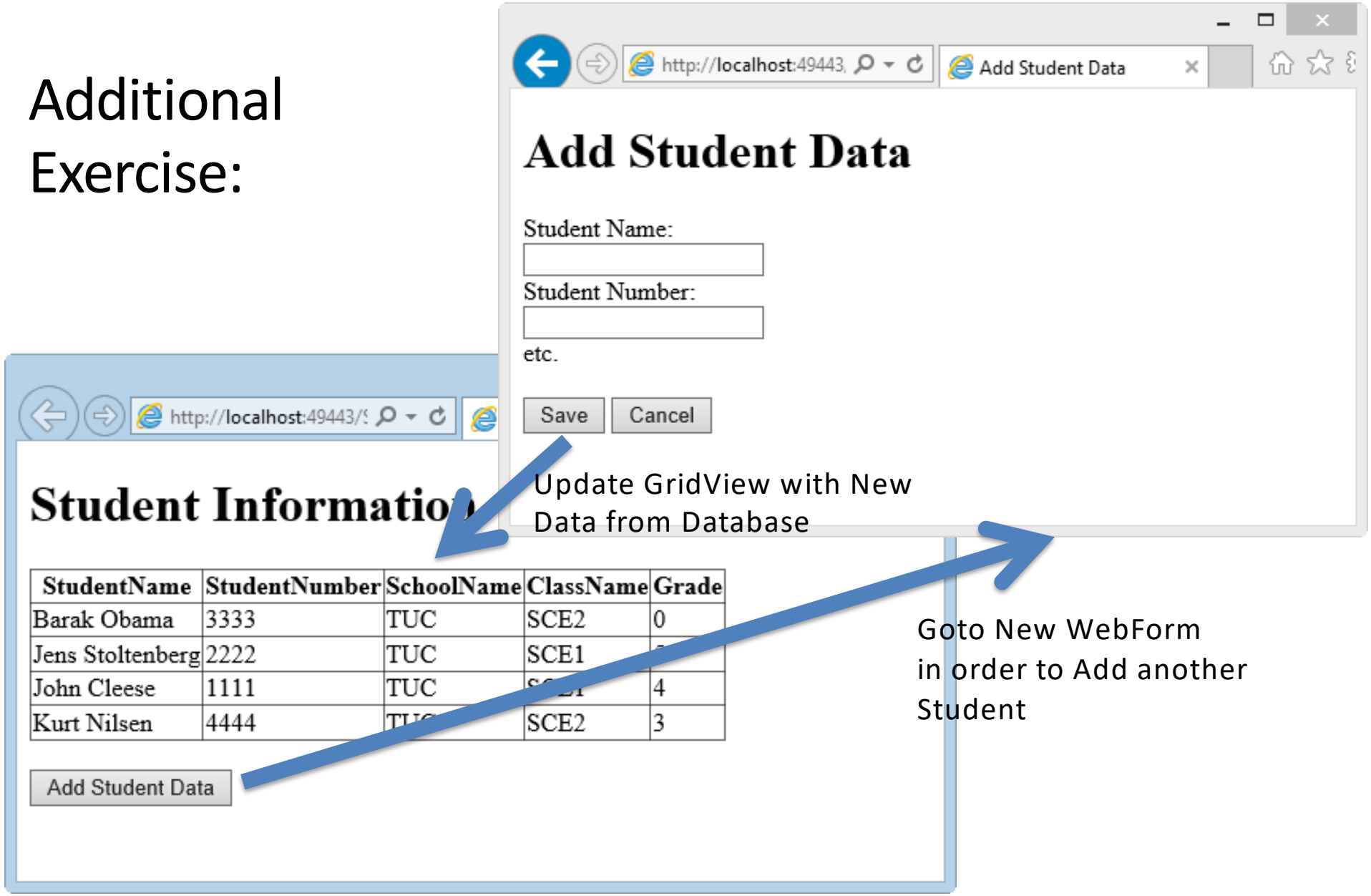


Student Information

StudentName	StudentNumber	SchoolName	ClassName	Grade
Barak Obama	3333	TUC	SCE2	0
Jens Stoltenberg	2222	TUC	SCE1	5
John Cleese	1111	TUC	SCE1	4
Kurt Nilsen	4444	TUC	SCE2	3

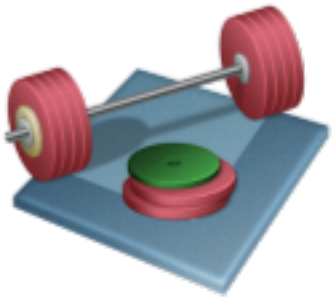
Note! We have used a “View” in order to get data from several tables

Additional Exercise:

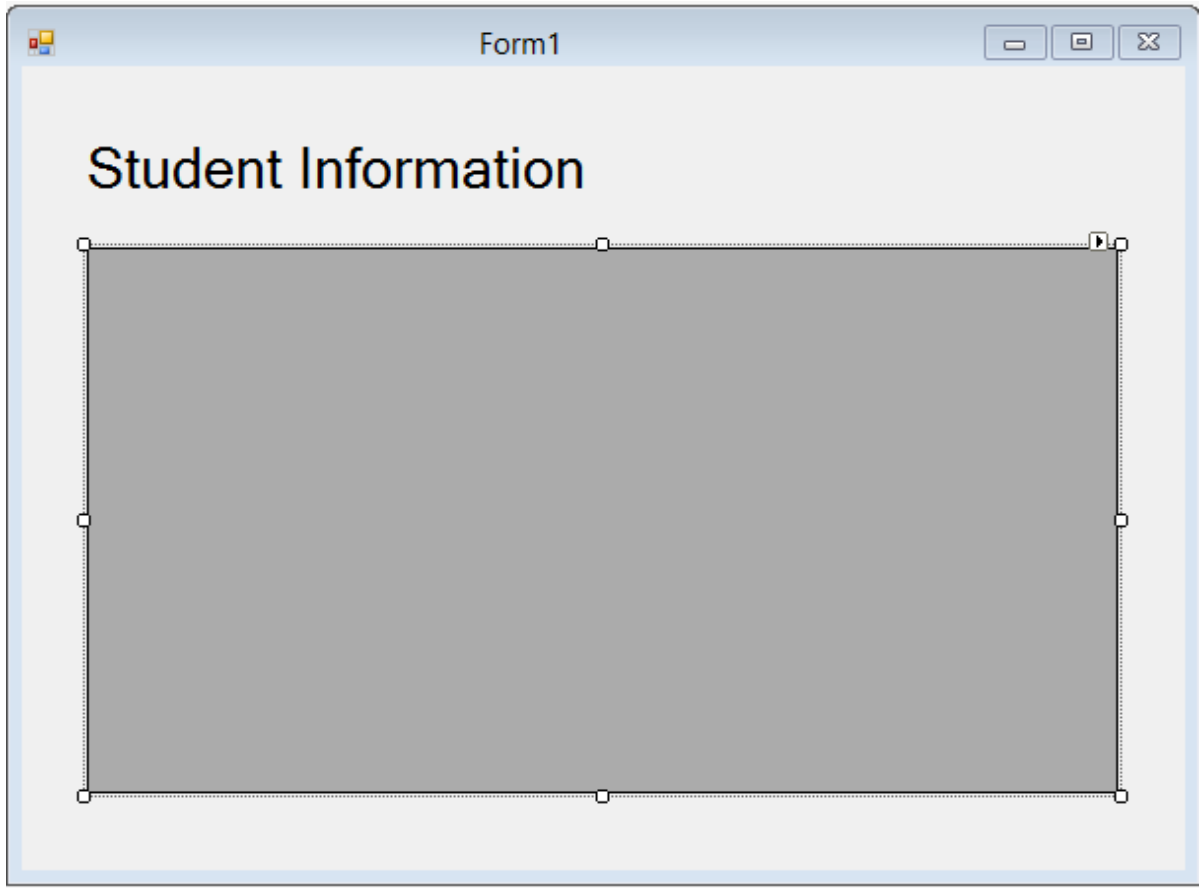


You are finished with the Exercise

Presentation Layer
Desktop App: WinForms



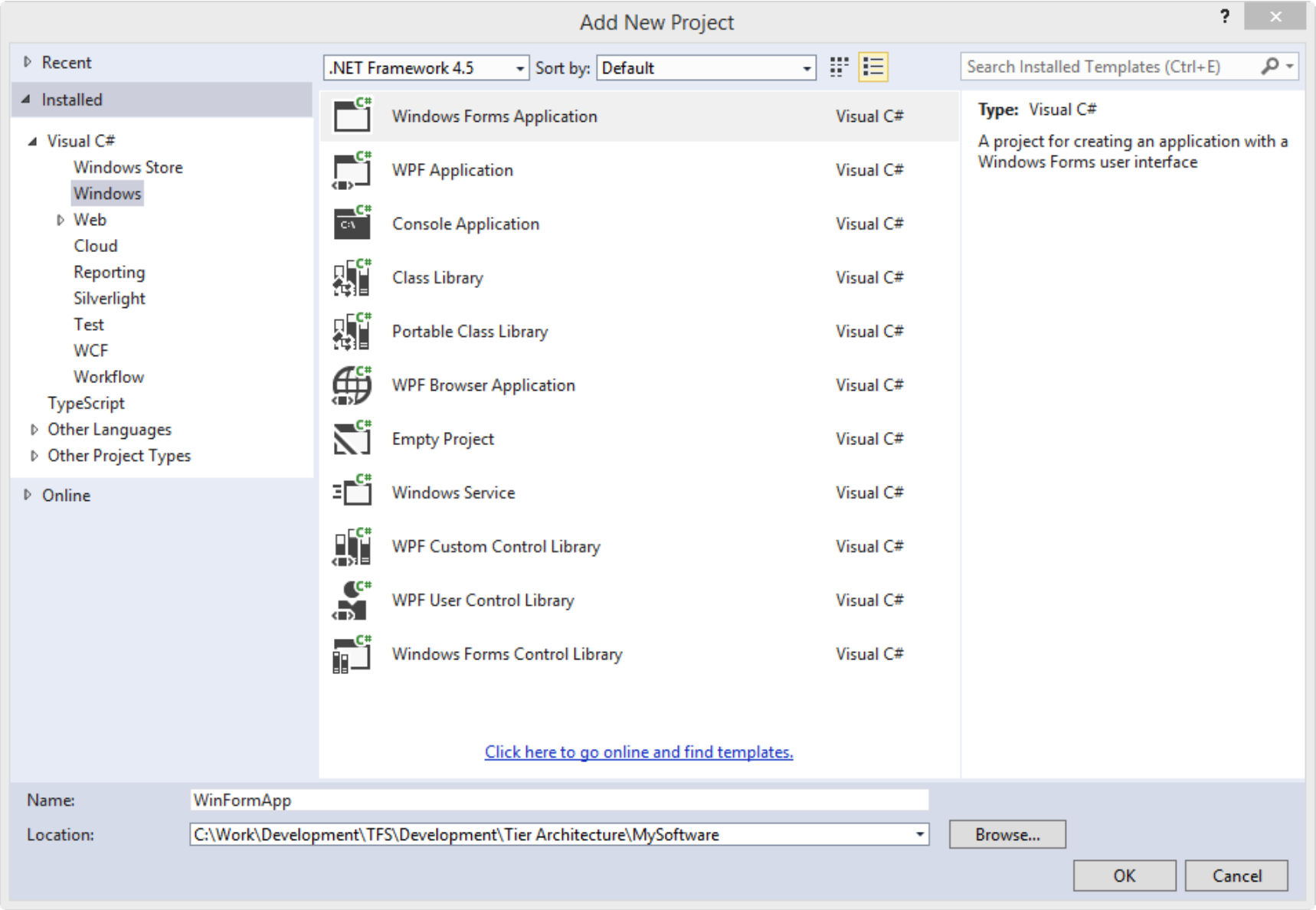
Part A: Without Web Services (we assume the App will be used only in the local LAN (or local on the same computer where the database is located) and that we have direct access to the Database)



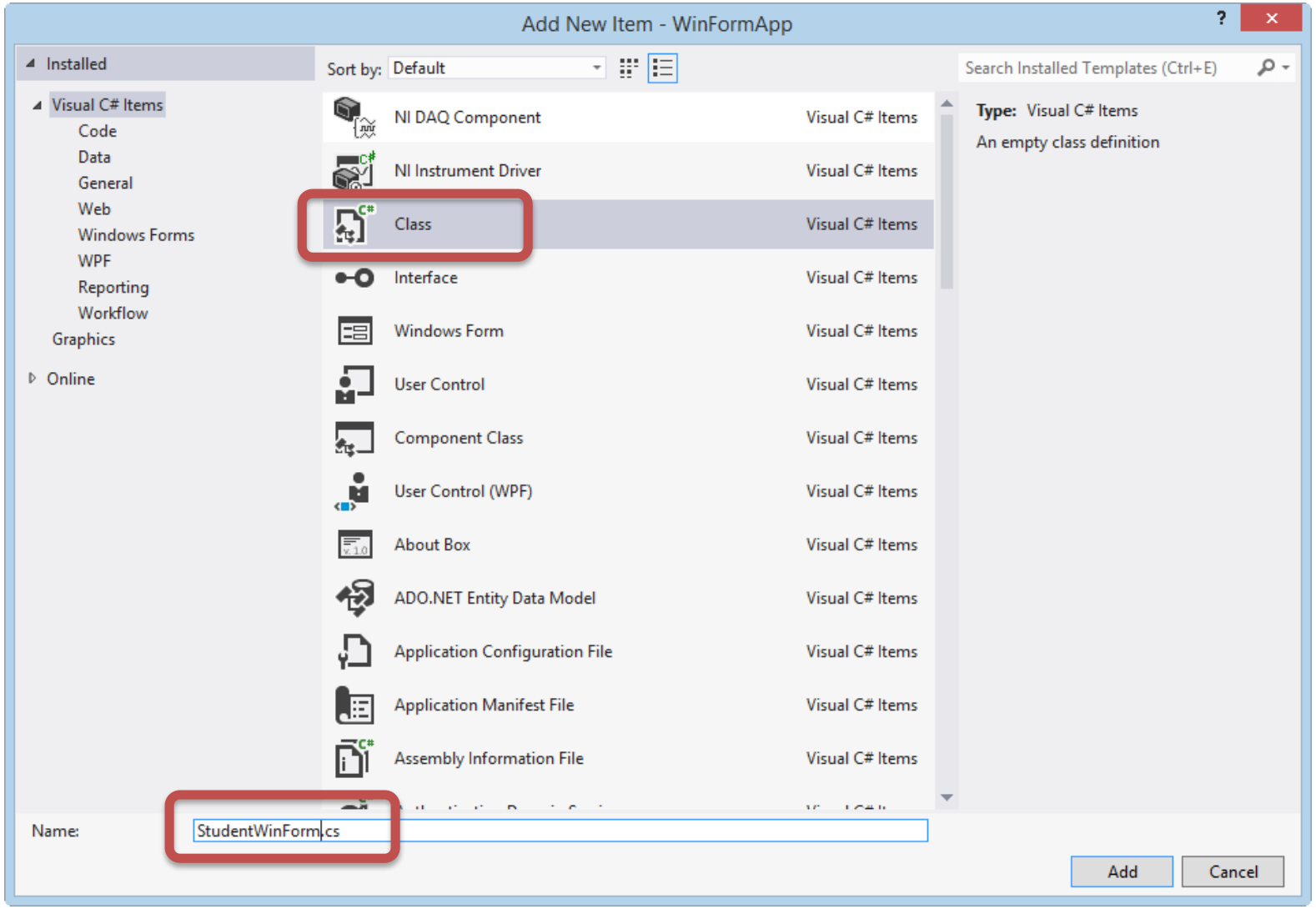
Label

DataGridView

Add a **WinForm Project**



Add a New **Class** (“StudentWinForm.cs”)



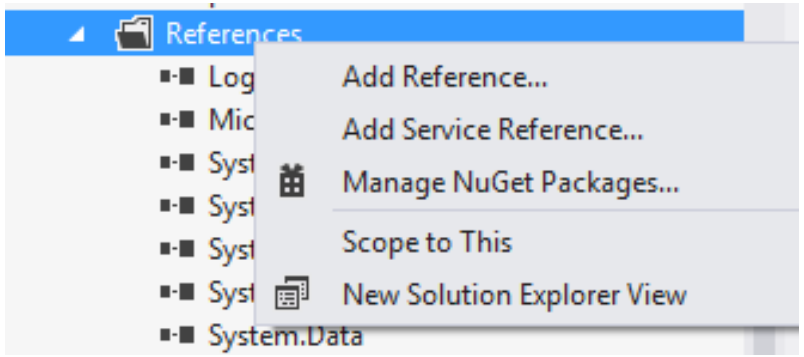
Add **Code** in Class

StudentWinForm.cs

Tuc.School.WinFormApp.StudentWinForm

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

using System.Data;

using Tuc.School.LogicTier;

namespace Tuc.School.WinFormApp
{
    class StudentWinForm
    {

        public DataSet GetStudent(string connectionString)
        {
            StudentData studentData = new StudentData();

            return studentData.GetStudentDB(connectionString);
        }

    }
}
```

Add a Reference to the Assembly in the Logic Tier



Code for Class “StudentWinForm.cs”

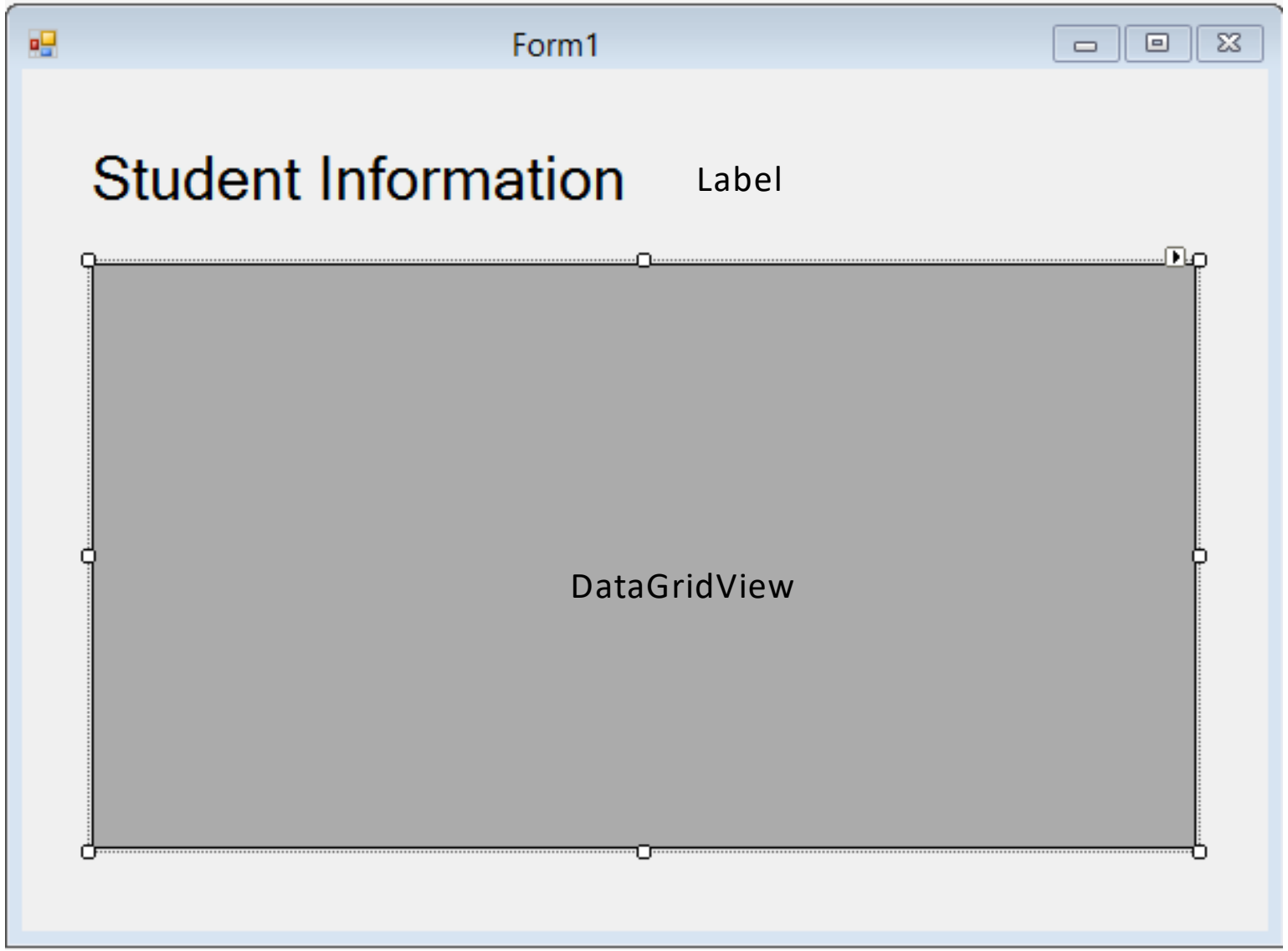
```
using System.Data;

using Tuc.School.LogicTier;

namespace Tuc.School.WinFormApp
{
    class StudentWinForm
    {

        public DataSet GetStudent(string connectionString)
        {
            StudentData studentData = new StudentData();

            return studentData.GetStudentDB(connectionString);
        }

    }
}
```

Since we are using the DataSet Class

Reference to our Logic Tier

Our Database Method in our Logic Tier

Create Form



Create Form Code

Form1.cs | Form1.cs [Design]

WinFormApp.Form1 | connectionString

```
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 System.Configuration;

using Tuc.School.WinFormApp;


namespace WinFormApp
{
    public partial class Form1 : Form
    {

        private string connectionString = ConfigurationManager.ConnectionStrings["SCHOOLConnectionString"].ConnectionString;

        public Form1()
        {
            InitializeComponent();
        }

        private void Form1_Load(object sender, EventArgs e)
        {
            FillStudentGrid();

        }


        private void FillStudentGrid()
        {

            DataSet ds = new DataSet();

            StudentWinForm studentList = new StudentWinForm();

            ds = studentList.GetStudent(connectionString);

            dataGridViewStudentInformation.DataSource = ds.Tables[0];

        }
    }
}
```

WinForm Code

```
using System.Configuration;
using Tuc.School.WinFormApp;

namespace WinFormApp
{
    public partial class Form1 : Form
    {

        private string connectionString =
            ConfigurationManager.ConnectionStrings["SCHOOLConnectionString"].ConnectionString;

        public Form1()
        {
            InitializeComponent();
        }

        private void Form1_Load(object sender, EventArgs e)
        {
            FillStudentGrid();

        }

        private void FillStudentGrid()
        {

            DataSet ds = new DataSet();

            StudentWinForm studentList = new StudentWinForm();

            ds = studentList.GetStudent(connectionString);

            dataGridViewStudentInformation.DataSource = ds.Tables[0];

        }
    }
}
```

Note!

ConnectionString is stored in App.config

Note! Add System.Configuration Reference

ConfigurationManagerImport - Microsoft Visual Studio (Administrator)

Form1.cs*

ConfigurationManagerImport.Form1

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

using System.Configuration;

namespace ConfigurationManagerImport
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();

            string conStr = ConfigurationManager.ConnectionStrings["connectionString"].ConnectionString;
        }
    }
}
```

3

class System.Configuration.ConfigurationManager
Provides access to configuration files for client applications. This class cannot be inherited.

4

Add Reference

.NET | COM | Projects | Browse | Recent

Filtered to: .NET Framework 4 Client Profile

Component Name
System.AddIn.Contract
System.AddIn
System.ComponentModel.Composition
System.ComponentModel.DataAnnotations
System.Configuration
System.Configuration.Install
System.Core
System.Data.DataSetExtensions
System.Data
System.Data.Entity

2

OK | Cancel

Solution Explorer

Solution 'ConfigurationManagerImport' (1 project)
ConfigurationManagerImport
Properties
References
Add Reference...
Add Service Reference...
Add Library Package Reference...
System.Data
System.Data.DataSetExtensions
System.Deployment
System.Drawing
System.Windows.Forms
System.Xml
System.Xml.Linq
App.config
Form1.cs
Program.cs

1

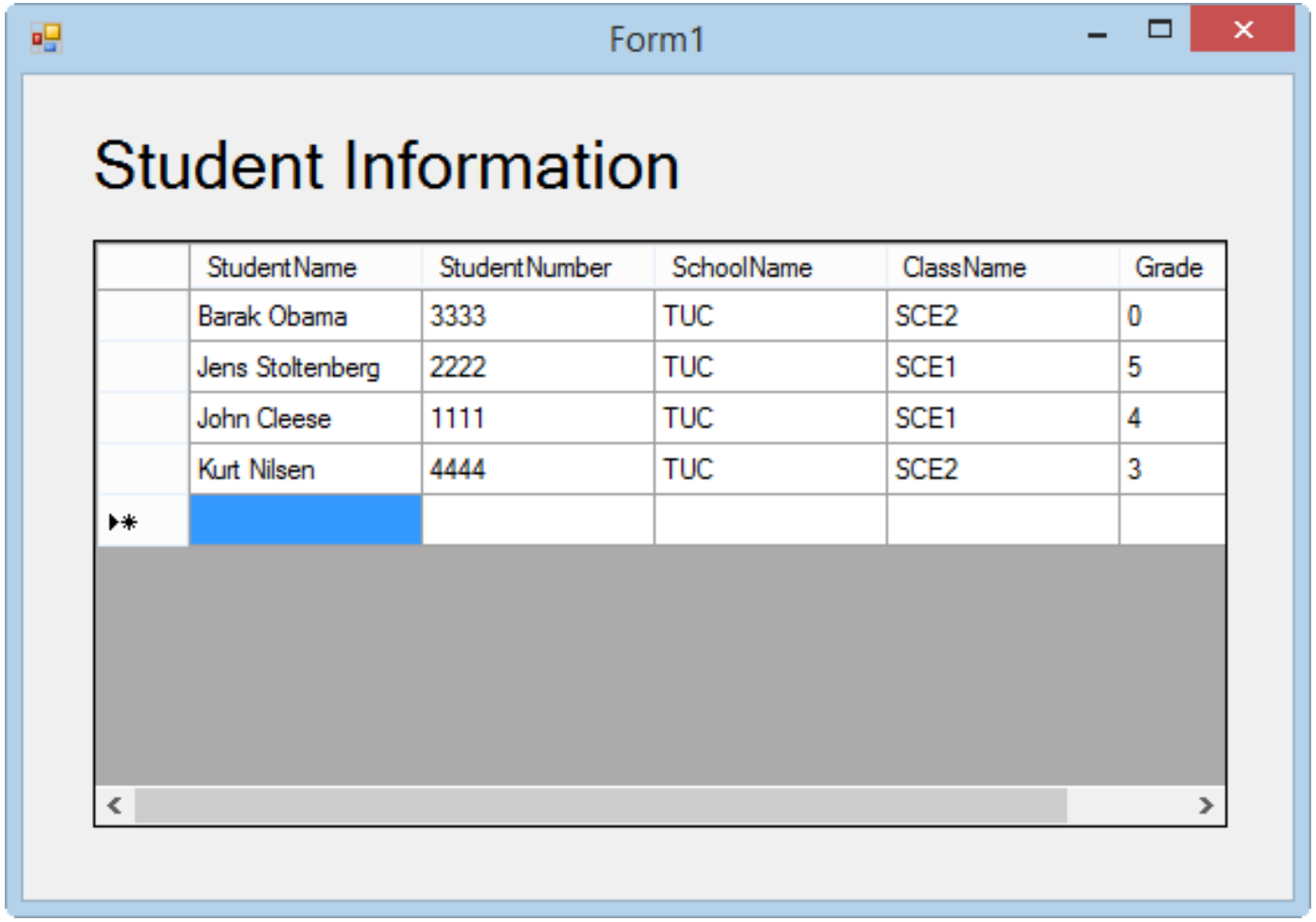
Create DB ConnectionString in **App.config**

```
<?xml version="1.0" encoding="utf-8" ?>
<configuration>

    <startup>
        <supportedRuntime version="v4.0" sku=".NETFramework,Version=v4.5" />
    </startup>


  <connectionStrings>
    <add name="SCHOOLConnectionString" connectionString="Data Source=macwin8;Initial Catalog=SCHOOL;Persist Security
Info=True;User ID=sa;Password=xxxxxx"
      providerName="System.Data.SqlClient" />
  </connectionStrings>


</configuration>
```

App.config

```
<?xml version="1.0" encoding="utf-8" ?>
<configuration>


    <startup>
        <supportedRuntime version="v4.0" sku=".NETFramework,Version=v4.5" />
    </startup>


  <connectionStrings>
    <add name="SCHOOLConnectionString" connectionString="Data Source=macwin8;Initial Catalog=SCHOOL;Persist Security Info=True;User ID=sa;Passwor
      providerName="System.Data.SqlClient" />
  </connectionStrings>



</configuration>
```

Test it

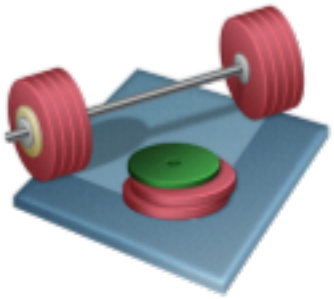


	StudentName	StudentNumber	SchoolName	ClassName	Grade
	Barak Obama	3333	TUC	SCE2	0
	Jens Stoltenberg	2222	TUC	SCE1	5
	John Cleese	1111	TUC	SCE1	4
	Kurt Nilsen	4444	TUC	SCE2	3
▸*					

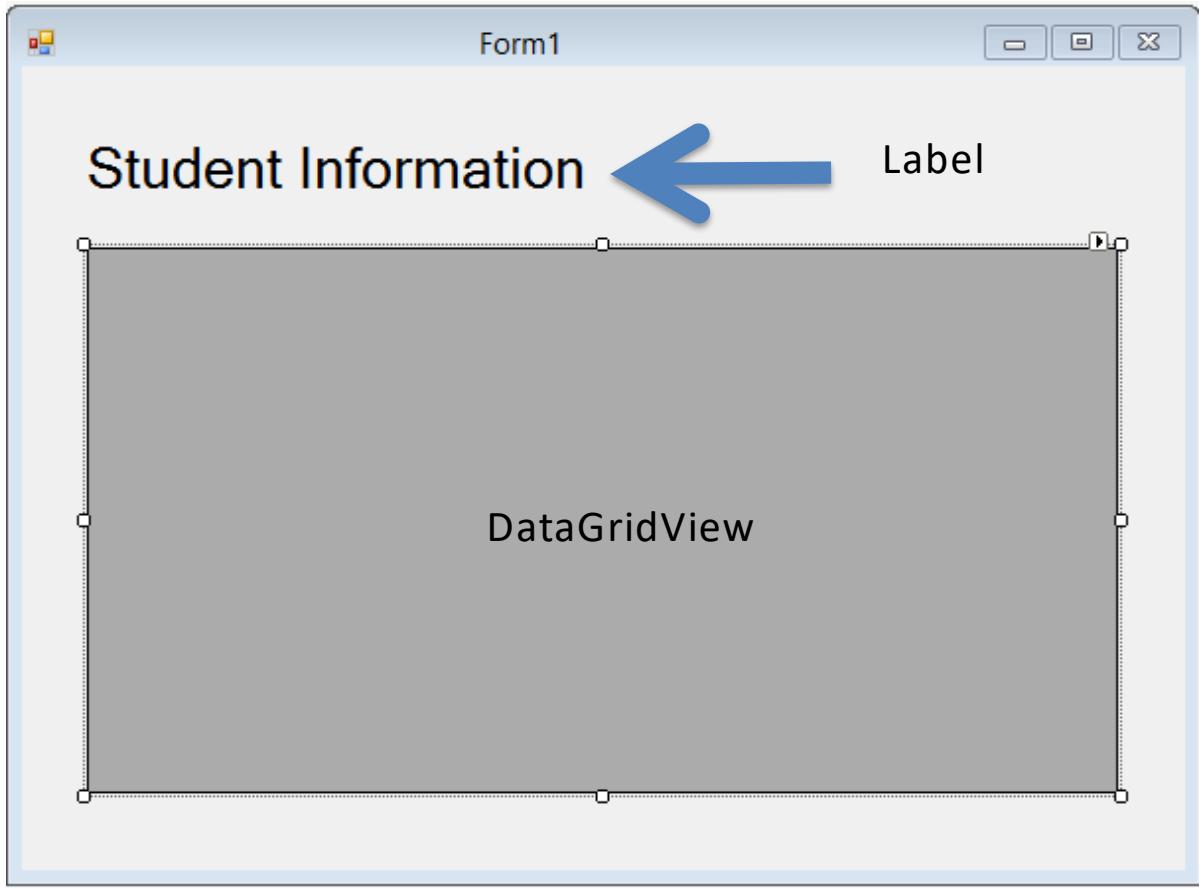
It works!!!

You are finished with the Exercise

Presentation Layer
Desktop App: WinForms



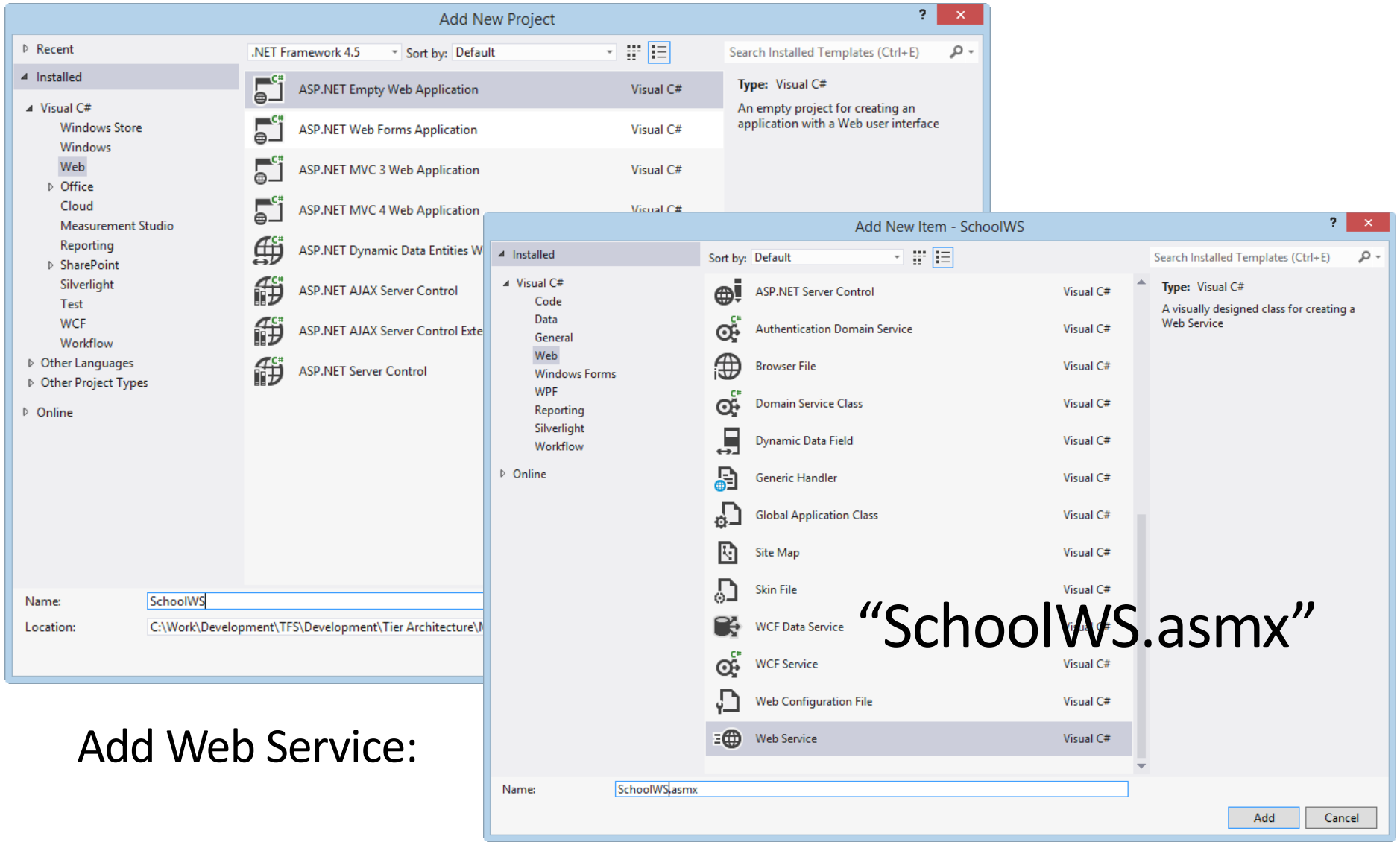
Part B: Using **Web Services** (we assume the The App should be used on Internet outside the Firewall)



Step 1: Create Web Service

"SchoolWS"

Create an ASP.NET Project:



"SchoolWS.asmx"

Add Web Service:

SchoolWS.asmx.cs

SchoolWS.SchoolWS | connectionString

Web Service Code

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

using System.Data;

using System.Web.Configuration;

using Tuc.School.LogicTier;

namespace SchoolWS
{
    /// <summary>
    /// Summary description for SchoolWS
    /// </summary>
    [WebService(Namespace = "http://tempuri.org/")]
    [WebServiceBinding(ConformsTo = WsiProfiles.BasicProfile1_1)]
    [System.ComponentModel.ToolboxItem(false)]
    // 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 SchoolWS : System.Web.Services.WebService
    {

        private string connectionString = WebConfigurationManager.ConnectionStrings["SCHOOLConnectionString"].ConnectionString;

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

        [WebMethod]
        public DataSet GetStudent()
        {
            StudentData studentData = new StudentData();

            return studentData.GetStudentDB(connectionString);
        }
    }
}
```

Database ConnectionString is located in Web.config

Web Service Method

Database ConnectionString is located in **Web.config**

Web.config | SchoolWS.asmx.cs | StudentInformation2

```xml
<?xml version="1.0"?>

<!--
  For more information on how to configure your ASP.NET application, please visit
  http://go.microsoft.com/fwlink/?LinkId=169433
  -->

<configuration>
    <system.web>
      <compilation debug="true" targetFramework="4.5" />
      <httpRuntime targetFramework="4.5" />
    </system.web>



  <connectionStrings>
    <add name="SCHOOLConnectionString" connectionString="Data Source=macwin8;Initial Catalog=SCHOOL;Persist Security Info=True;User ID=sa;Password=
      providerName="System.Data.SqlClient" />
  </connectionStrings>


</configuration>
```

Test Web Service

http://localhost:49564/S

SchoolWS Web Service

SchoolWS

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

- GetStudent
- HelloWorld

Click to Test the Web Service Method we created

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/":

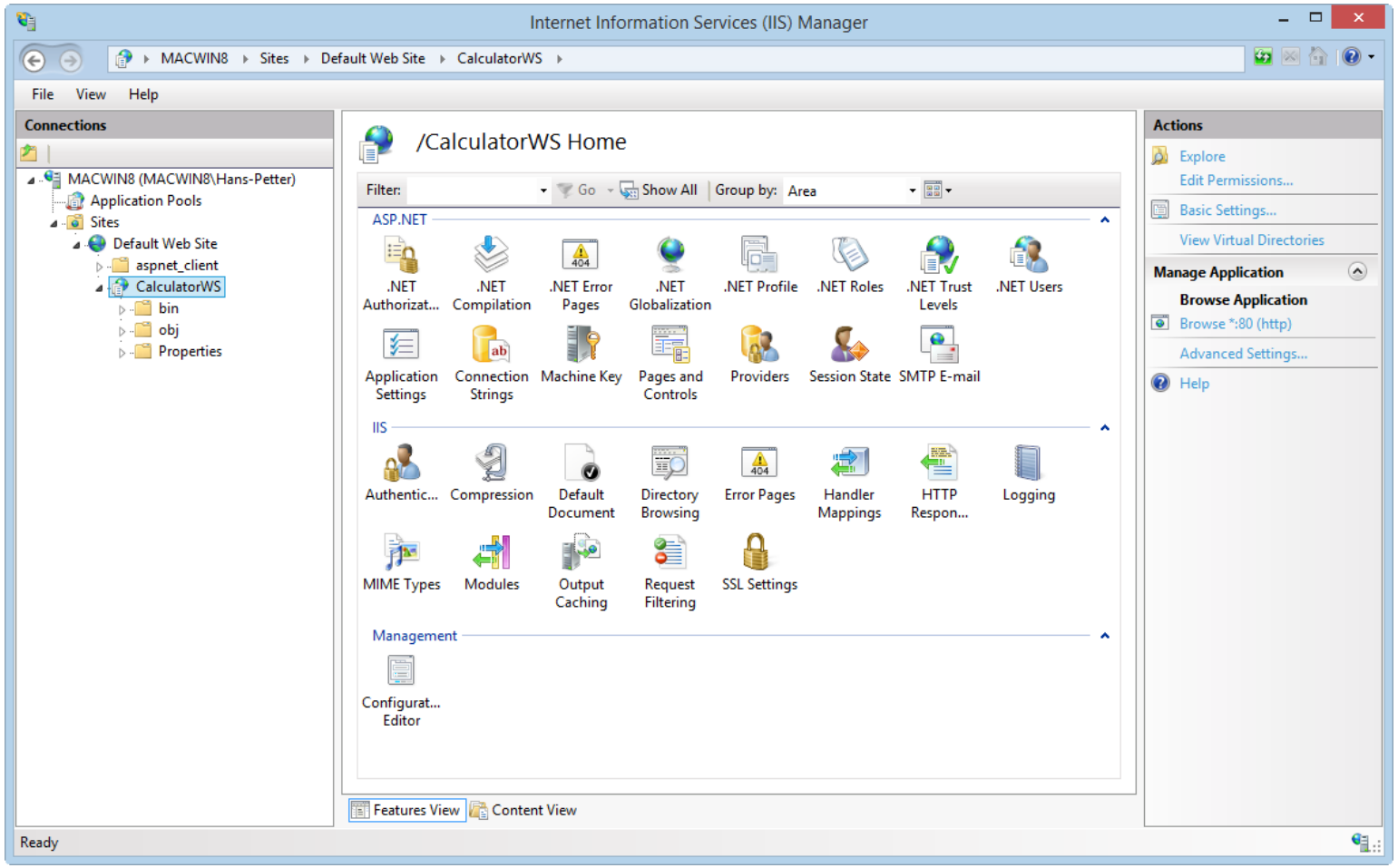
C#

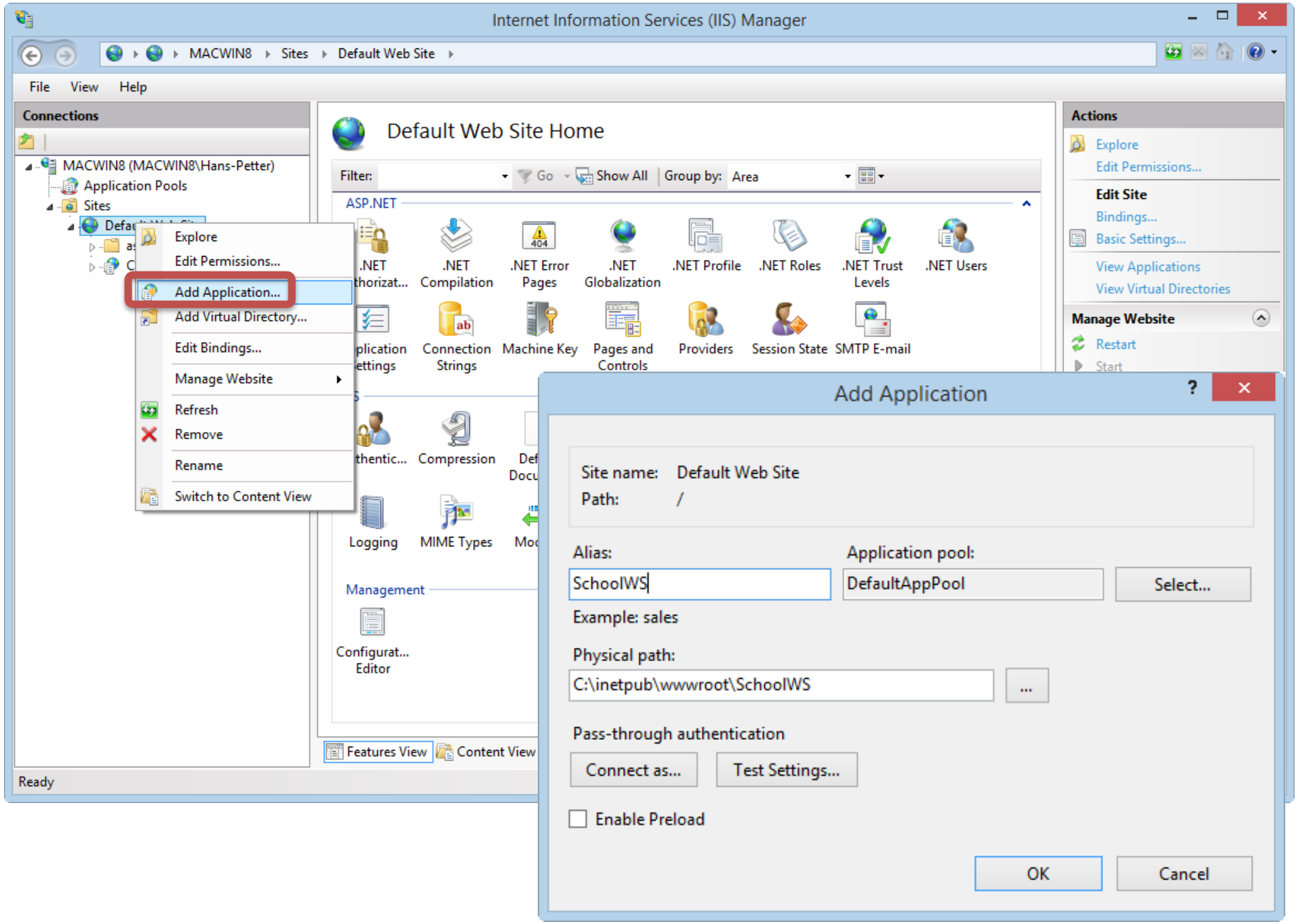
```
[WebService(Namespace="http://microsoft.com/webservices/")]
public class MyWebService {
    // implementation
}
```

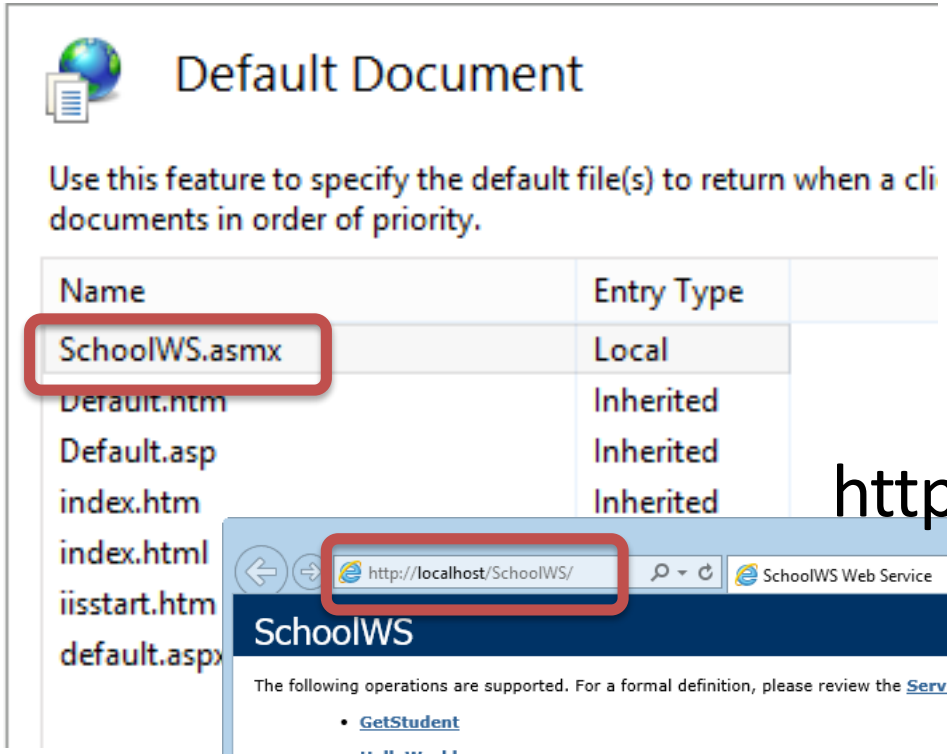
It Works!!

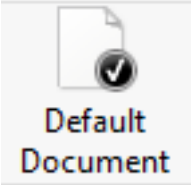
Deploy/Publish Web Service to **IIS**

Copy Web Service Files (Project) to default IIS Directory: C:\inetpub\wwwroot



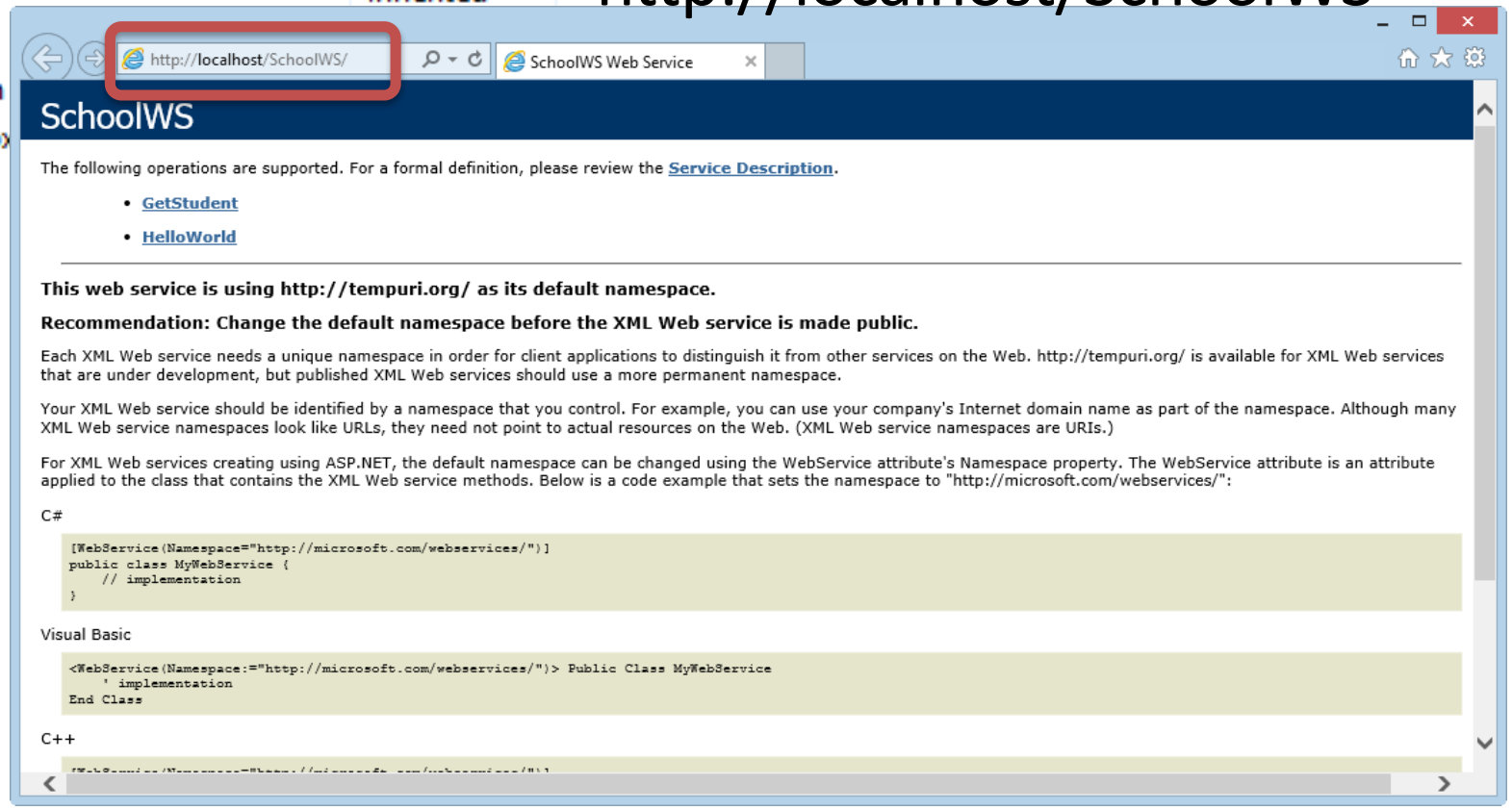

Internet Information Services (IIS) Manager
MACWIN8 ▸ Sites ▸ Default Web Site ▸
File View Help
Connections
MACWIN8 (MACWIN8\Hans-Petter)
Application Pools
Sites
Explore
Edit Permissions...
Add Application...
Add Virtual Directory...
Edit Bindings...
Manage Website
Refresh
Remove
Rename
Switch to Content View
Default Web Site Home
Filter:
Go
Show All
Group by: Area
ASP.NET
.NET Compilation
.NET Error Pages
.NET Globalization
.NET Profile
.NET Roles
.NET Trust Levels
.NET Users
Connection Strings
Machine Key
Pages and Controls
Providers
Session State
SMTP E-mail
Compression
Logging
MIME Types
Management
Configurat... Editor
Features View
Content View
Ready
Actions
Explore
Edit Permissions...
Edit Site
Bindings...
Basic Settings...
View Applications
View Virtual Directories
Manage Website
Restart
Start
Add Application
Site name: Default Web Site
Path: /
Alias:
SchoolWS
Application pool:
DefaultAppPool
Select...
Example: sales
Physical path:
C:\inetpub\wwwroot\SchoolWS
...
Pass-through authentication
Connect as...
Test Settings...
Enable Preload
OK
Cancel





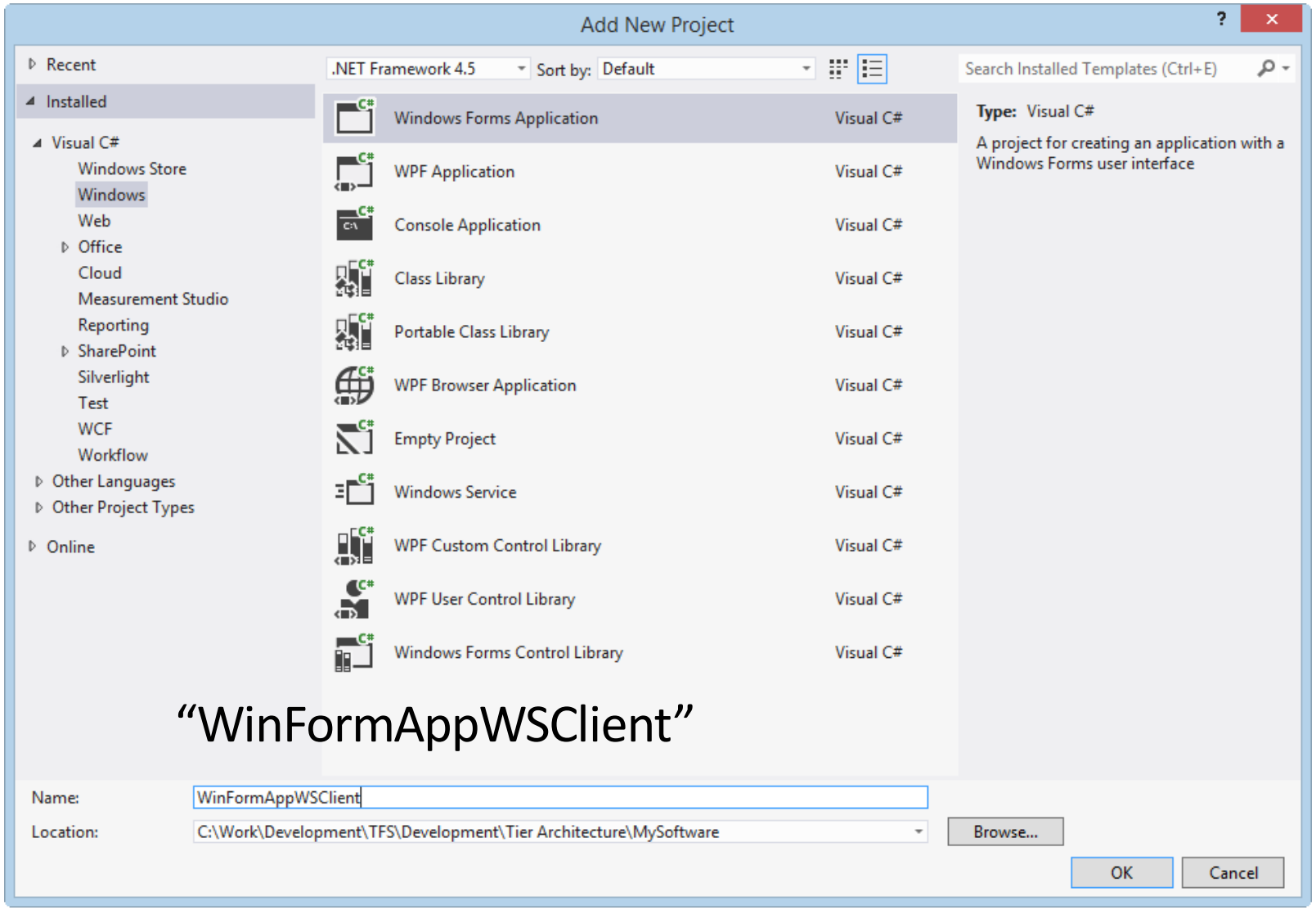
Test if WS working:

http://localhost/SchoolWS

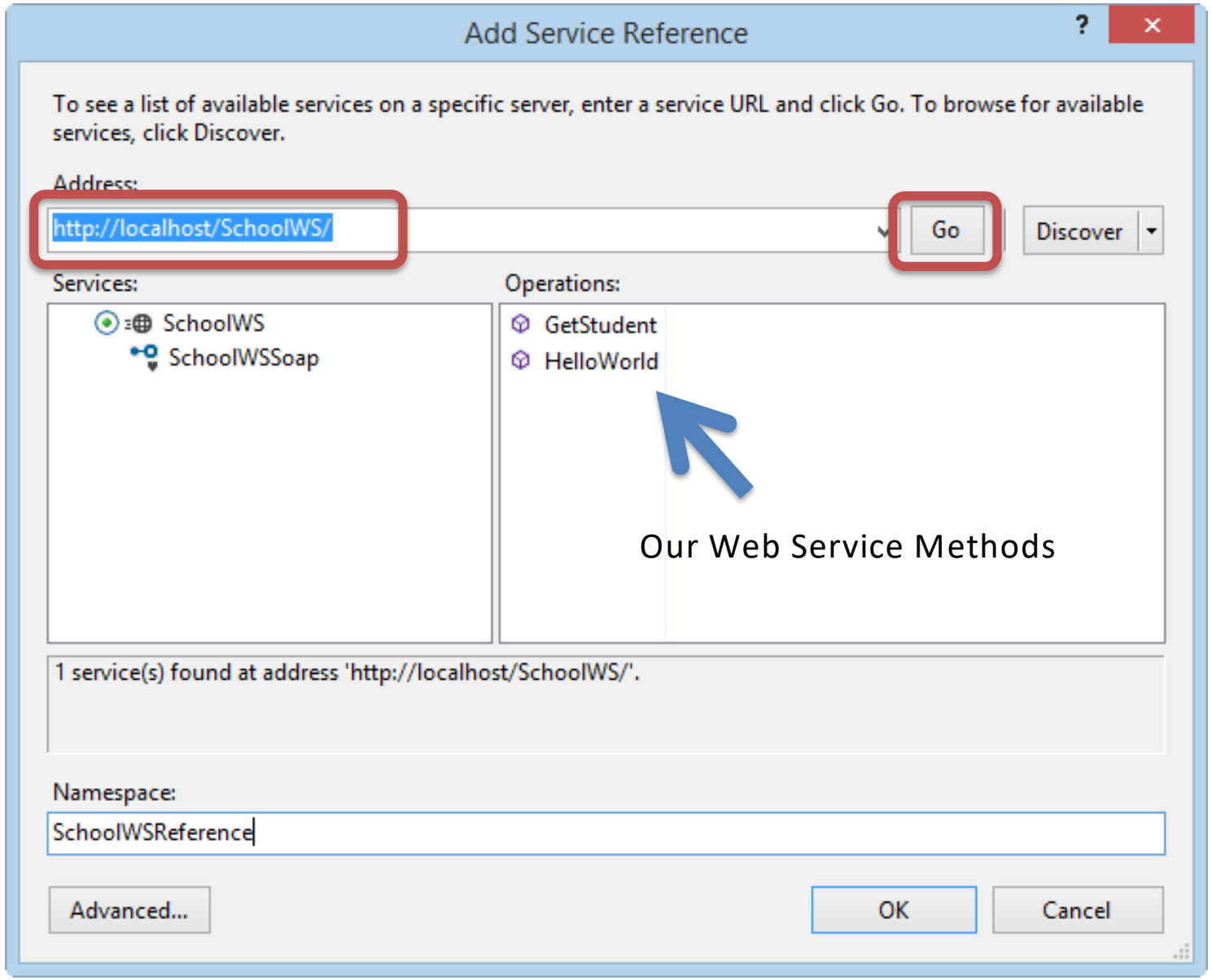


Step 2: Use Web Service in WinForm

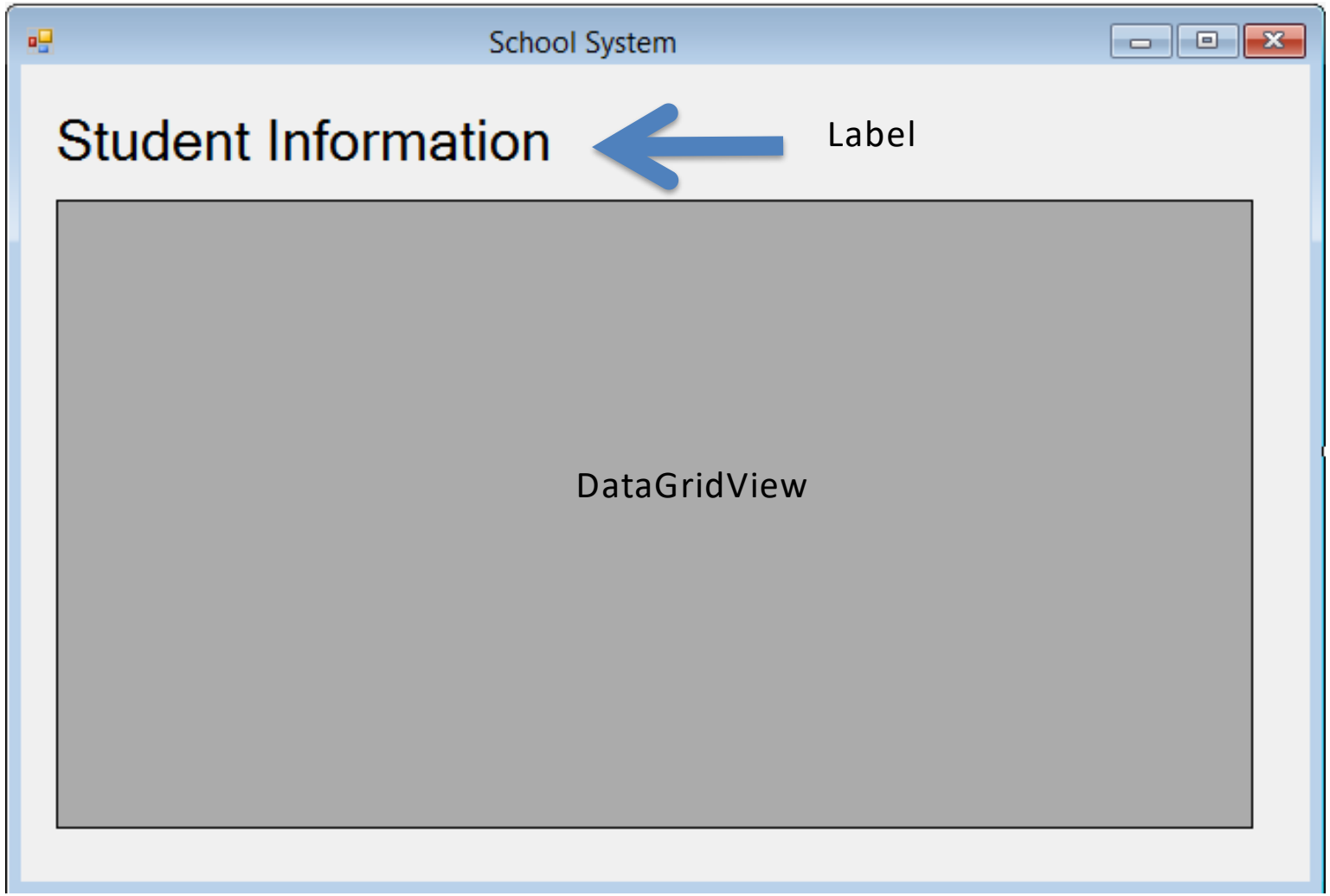
Create New WinForm Project:



Add Web Service Reference



Create **GUI**



FormWSClient.cs | FormWSClient.cs [Design]

WinFormAppWSClient.FormWSClient | FormWSClient_Load(object sende

Create 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 WinFormAppWSClient
{
    public partial class FormWSClient : Form
    {

        public FormWSClient()
        {
            InitializeComponent();
        }


        private void FormWSClient_Load(object sender, EventArgs e)
        {

        FillStudentGrid();

        }


        private void FillStudentGrid()
        {

            DataSet ds = new DataSet();

            SchoolWSReference.SchoolWSSoapClient schoolWs = new SchoolWSReference.SchoolWSSoapClient();

            ds = schoolWs.GetStudent();

            dataGridViewStudentInformation.DataSource = ds.Tables[0];


        }
```

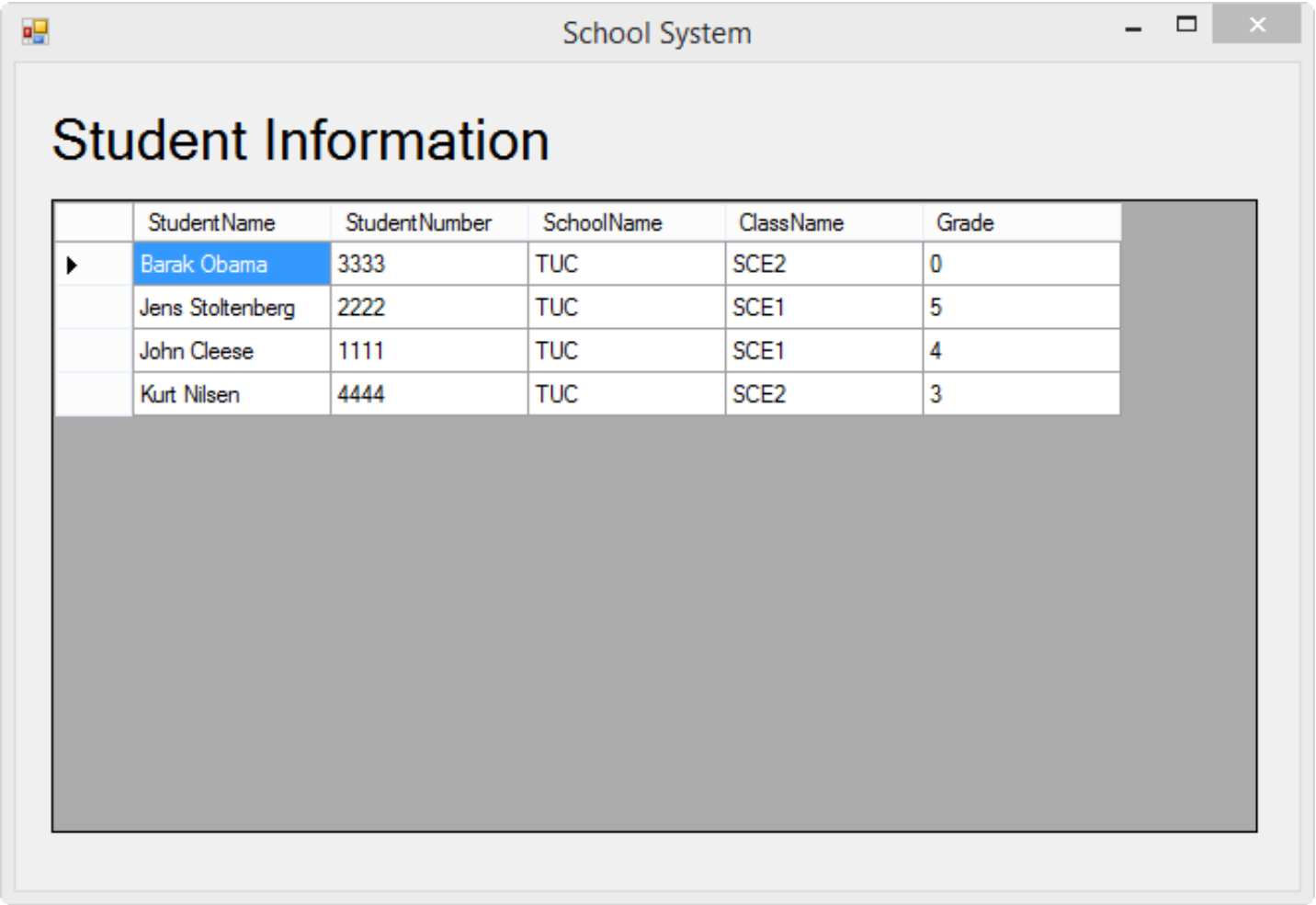
WinForm Code

```
using System.Windows.Forms;


namespace WinFormAppWSClient
{
    public partial class FormWSClient : Form
    {

        public FormWSClient()
        {
            InitializeComponent();
        }

        private void FormWSClient_Load(object sender, EventArgs e)
        {

        FillStudentGrid();
        }

        private void FillStudentGrid()
        {

            DataSet ds = new DataSet();

            SchoolWSReference.SchoolWSSoapClient schoolWs = new
SchoolWSReference.SchoolWSSoapClient();

            ds = schoolWs.GetStudent();

            dataGridViewStudentInformation.DataSource = ds.Tables[0];

        }
}
```

Call the Web Service method

Fill GridView

Test it:



StudentName	StudentNumber	SchoolName	ClassName	Grade
Barak Obama	3333	TUC	SCE2	0
Jens Stoltenberg	2222	TUC	SCE1	5
John Cleese	1111	TUC	SCE1	4
Kurt Nilsen	4444	TUC	SCE2	3

It works!!!

You are finished with the Exercise

Hans-Petter Halvorsen

University of South-Eastern Norway

www.usn.no

E-mail: hans.p.halvorsen@usn.no

Web: https://www.halvorsen.blog