



Datalogging and Monitoring

with Step by Step Examples

Content

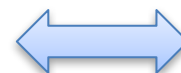
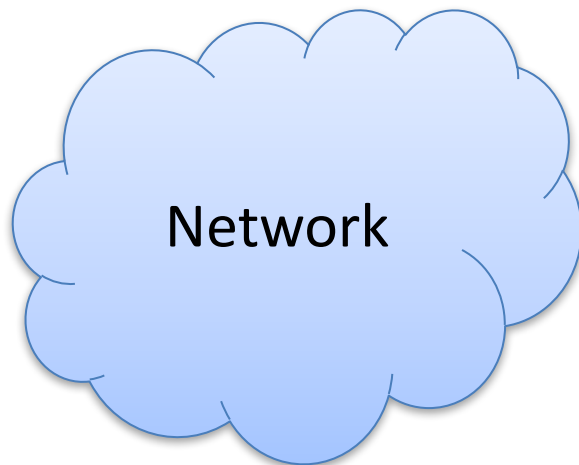
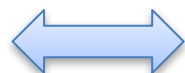
- Different Apps for Data Logging and Data Monitoring will be presented
- Here you find lots of examples in LabVIEW and Visual Studio/C#.
- The data is stored in SQL Server.
- Cloud solutions: Here you also find Microsoft Azure examples and Web API examples, etc.
- Web APIs, REST APIs or Web Services disconnect the logging from using the Database directly



System Overview



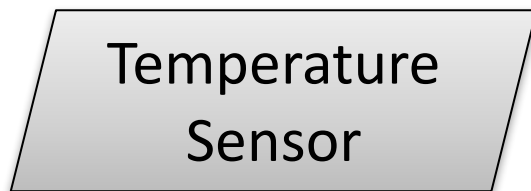
Logging App



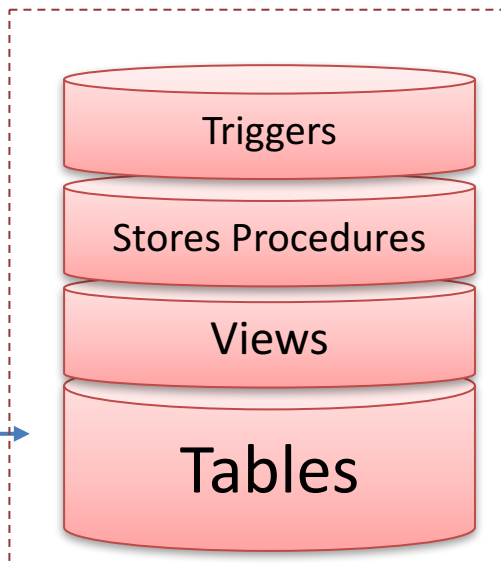
Monitoring App



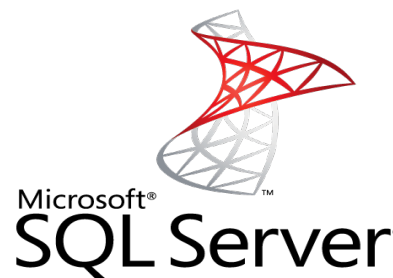
ASP.NET



TC-01 Thermocouple



erwin
Table Design





Database

Database

In this Example we will use the following simple Database:

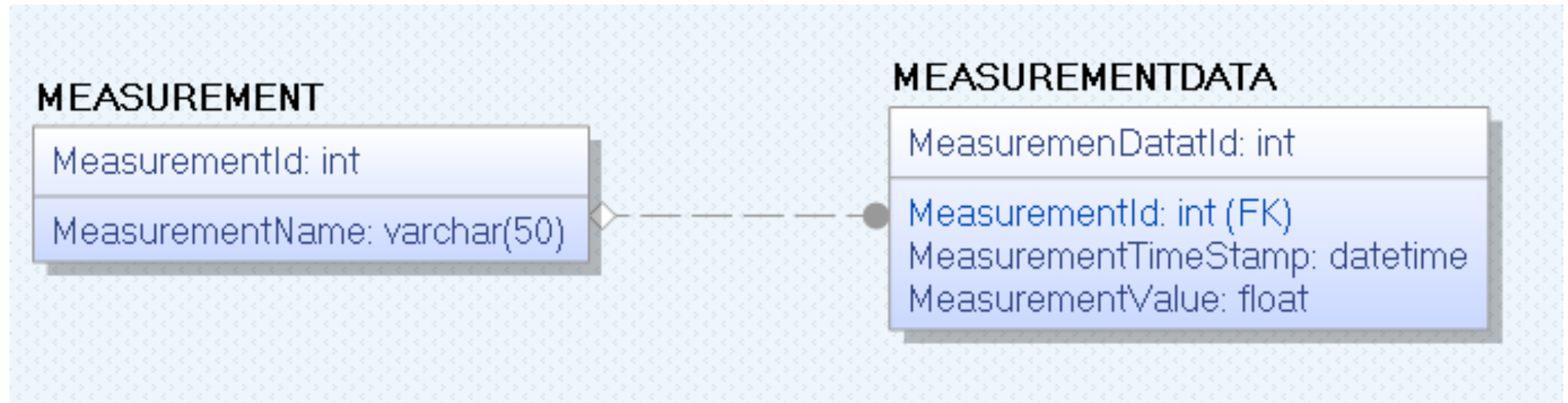


Table Script

```
CREATE TABLE [MEASUREMENT]
(
  [MeasurementId]    int NOT NULL IDENTITY ( 1,1 ) Primary Key,
  [MeasurementName]  varchar(50) NOT NULL UNIQUE
)
go

CREATE TABLE [MEASUREMENTDATA]
(
  [MeasurementDataId] int NOT NULL IDENTITY ( 1,1 ) Primary Key,
  [MeasurementId]    int NOT NULL Foreign Key REFERENCES
  MEASUREMENT(MeasurementId),
  [MeasurementTimeStamp] datetime NOT NULL ,
  [MeasurementValue] float NOT NULL
)
go
```

Stored Procedure

```
IF EXISTS (SELECT name
FROM sysobjects
WHERE name = 'SaveMeasurementData'
AND type = 'P')
DROP PROCEDURE SaveMeasurementData
GO
```

```
CREATE PROCEDURE SaveMeasurementData
@MeasurementName varchar(50),
@MeasurementValue float
AS
```

```
DECLARE
@MeasurementId int
```

```
if not exists (select * from MEASUREMENT where MeasurementName = @MeasurementName)
insert into MEASUREMENT (MeasurementName) values (@MeasurementName)
```

```
else
```

```
select @MeasurementId = MeasurementId from MEASUREMENT where MeasurementName = @MeasurementName
```

```
insert into MEASUREMENTDATA (MeasurementId, MeasurementValue, MeasurementTimeStamp)
values (@MeasurementId, @MeasurementValue, getdate())
```

```
GO
```

View

A View is used to collect
Data from multiple Tables

```
IF EXISTS (SELECT name
FROM sysobjects
WHERE name = 'GetMeasurementData'
AND type = 'V')
DROP VIEW GetMeasurementData
GO
```

```
CREATE VIEW GetMeasurementData
AS
```

```
SELECT
MEASUREMENTDATA.MeasurementDataId,
MEASUREMENT.MeasurementId,
MEASUREMENT.MeasurementName,
MEASUREMENTDATA.MeasurementTimeStamp,
MEASUREMENTDATA.MeasurementValue
```

```
FROM MEASUREMENTDATA
INNER JOIN MEASUREMENT ON
MEASUREMENTDATA.MeasurementId =
MEASUREMENT.MeasurementId
```

```
GO
```




Data Logging

Datalogging Example

erwin

Database Design
& Modelling

Create Stored Procedure
and View in SQL Server

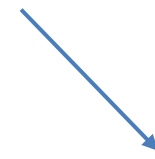
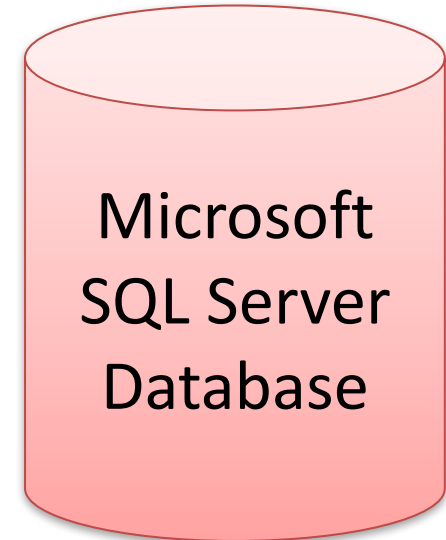
TC-01 Thermocouple



DAQ



Data
Stored Procedure

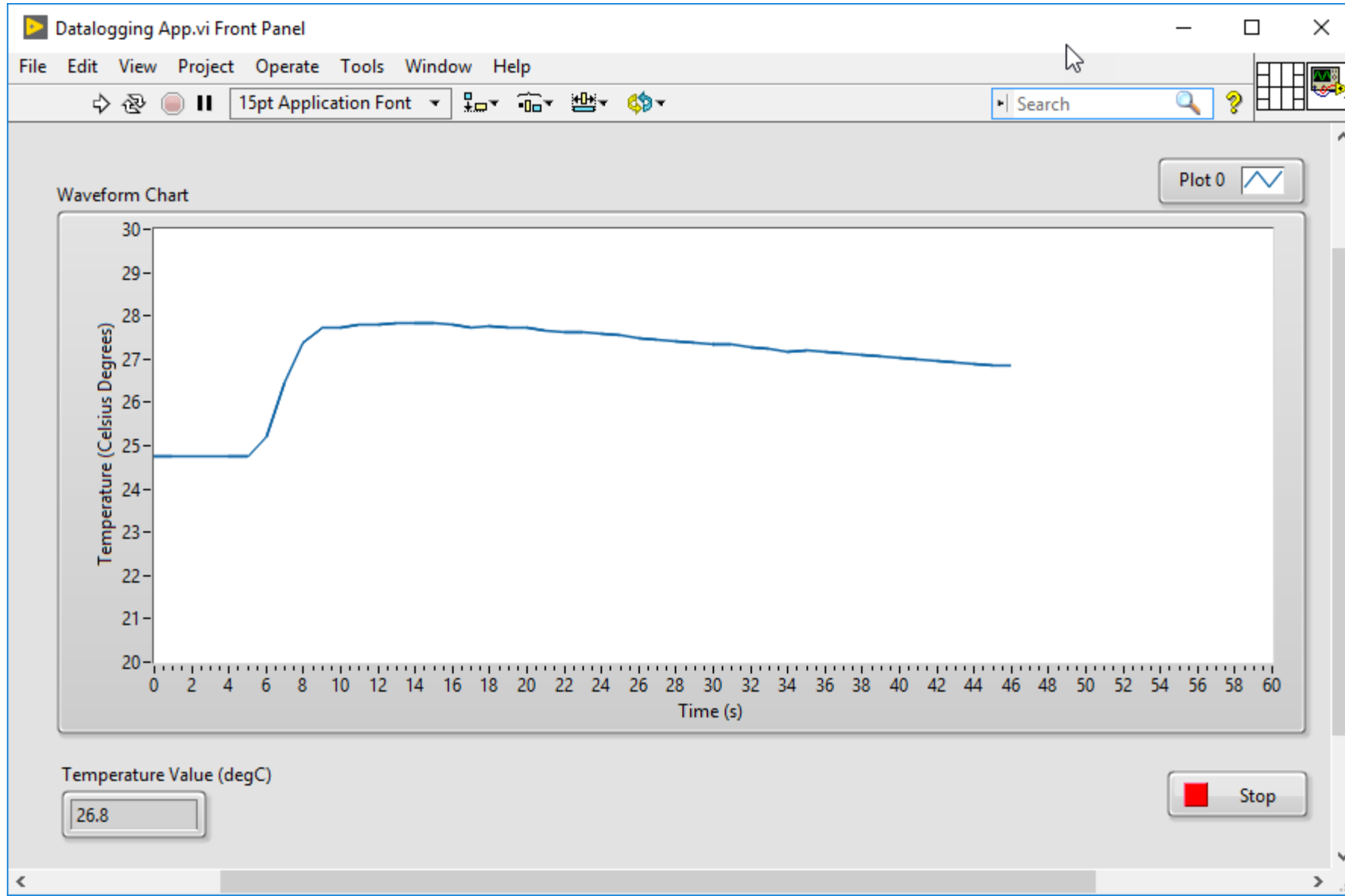




Data Logging

LabVIEW Example

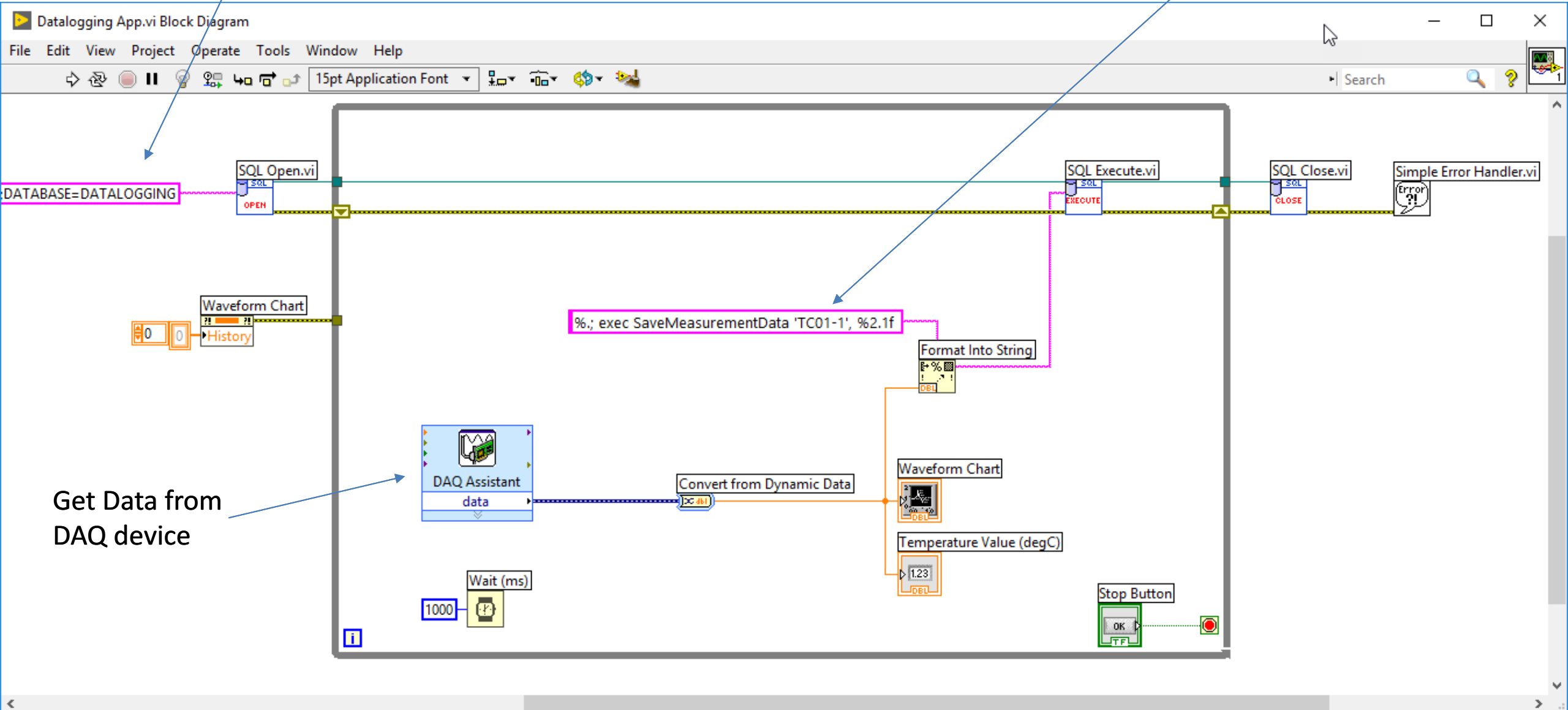
LabVIEW Example



LabVIEW Example

Connection string to the Database

Stored Procedure



Get Data from DAQ device



Data Logging

Visual Studio/C# Example

WinForm Example

User Interface Example

The screenshot shows a Windows application window titled "TC-01 Logging with Timer". The window has a standard Windows title bar with minimize, maximize, and close buttons. The main content area is light gray and contains four rows of input fields:

- Sensor Name:** A text box containing "TC01-1".
- Data Rate:** A text box containing "10" followed by the text "seconds" to its right.
- Measurement Value:** A text box containing "24.7".
- TimeStamp:** A text box containing "2017-08-28 13:53:51".

This is a simple Application retrieving Data from the Sensor.
The Data are then stored in a local SQL Server Database

Connection Sting in App.Config

```
App.config  # X
1  <?xml version="1.0" encoding="utf-8" ?>
2  <configuration>
3
4
5  <startup>
6      <supportedRuntime version="v4.0" sku=".NETFramework,Version=v4.6.1" />
7  </startup>
8
9  <connectionStrings>
10 <add name="DatabaseConnectionString" connectionString="Data Source=Y...;Initial Catalog=DATALOGGING;Trusted_Connection=True"
11     providerName="System.Data.SqlClient" />
12 </connectionStrings>
13
14
15 </configuration>
```

It is recommended that you store the Connection string in App.Config

Timer

```
public Form1()
{
    InitializeComponent();

    timer1.Interval = 10000;

    timer1.Start();
}

private void timer1_Tick(object sender, EventArgs e)
{

    GetSensorData();

    DateTime timeStamp = DateTime.Now;
    txtTimeStamp.Text = timeStamp.ToString();

    SaveMeasurementData();

}
```

Get Measurement Data from TC-01 Sensor

```
void GetSensorData()
{
    Task temperatureTask = new Task();

    AIChannel myAIChannel;

    myAIChannel = temperatureTask.AIChannels.CreateThermocoupleChannel(
        "Dev1/ai0",
        "Temperature",
        0,
        100,
        AIThermocoupleType.J,
        AITemperatureUnits.DegreesC
    );

    AnalogSingleChannelReader reader = new AnalogSingleChannelReader(temperatureTask.Stream);

    double analogDataIn = reader.ReadSingleSample();

    txtMeasurementValue.Text = analogDataIn.ToString("0.0");
}
```

Save Measurement Data to Database

```
void SaveMeasurementData()
{
    string sensorName;
    double measurementValue;

    sensorName = txtSensorName.Text;
    measurementValue = Convert.ToDouble(txtMeasurementValue.Text);

    try
    {
        using (SqlConnection con = new SqlConnection(connectionString))
        {
            SqlCommand cmd = new SqlCommand("SaveMeasurementData", con);
            cmd.CommandType = CommandType.StoredProcedure;

            cmd.Parameters.Add(new SqlParameter("@MeasurementName", sensorName));
            cmd.Parameters.Add(new SqlParameter("@MeasurementValue", measurementValue));

            con.Open();
            cmd.ExecuteNonQuery();
            con.Close();
        }
    }
    catch (Exception ex)
    {
        throw ex;
    }
}
```



Monitoring

Monitoring

- We will create some basic Web Applications using ASP.NET
- ASP.NET is a Web Framework for creating Web Pages
- ASP.NET is built on top of the .NET Framework
- You use Visual Studio and C#
- ASP.NET Web Forms are very similar to standard Win Forms that you are already familiar with.
- If you know ordinary WinForms, you also know ASP.NET WebForms!



ASP.NET

GridView Example

MeasurementData

MeasurementDataId	MeasurementTimeStamp	MeasurementValue
2	2017-08-28 10:22:57	24.3
3	2017-08-28 10:22:58	24.3
4	2017-08-28 10:22:59	24.3
5	2017-08-28 10:23:00	24.3
6	2017-08-28 10:23:01	24.3
7	2017-08-28 10:23:02	24.3
8	2017-08-28 10:23:03	24.3
9	2017-08-28 10:23:04	24.4
10	2017-08-28 10:23:05	24.3
11	2017-08-28 10:23:06	24.3
12	2017-08-28 10:23:07	24.3
13	2017-08-28 10:23:08	24.4
14	2017-08-28 10:23:09	24.3
15	2017-08-28 10:23:10	24.3
16	2017-08-28 10:23:11	24.4
17	2017-08-28 10:23:12	24.4
18	2017-08-28 10:23:13	24.3
19	2017-08-28 10:23:14	24.3
20	2017-08-28 10:23:15	24.4
21	2017-08-28 10:23:16	24.4
22	2017-08-28 10:23:17	24.4
23	2017-08-28 10:23:18	24.4
24	2017-08-28 10:23:19	24.4
25	2017-08-28 10:23:20	24.4
26	2017-08-28 10:23:21	24.4
27	2017-08-28 10:23:22	24.4
28	2017-08-28 10:23:23	24.4
29	2017-08-28 10:23:24	24.3
30	2017-08-28 10:23:25	24.4
31	2017-08-28 10:23:26	24.3
32	2017-08-28 10:23:27	24.4
33	2017-08-28 10:23:28	24.3

← GridView



Create New ASP.NET Application

ASP.NET Web Application

- Choose File -> New Project

New Project

Recent

Installed

Templates




- Visual C#
 - Windows Universal
 - Windows Classic Desktop
 - Web
 - .NET Core
 - .NET Standard
 - Cloud
 - Test
 - WCF
- Azure Data Lake
- Other Languages
- Other Project Types

Not finding what you are looking for?
[Open Visual Studio Installer](#)

Online

.NET Framework 4.5.2 Sort by: Default

Search Installed Templates (Ctrl+E)

	ASP.NET Web Application (.NET Framework)	Visual C#
	ASP.NET Core Web Application (.NET Core)	Visual C#
	ASP.NET Core Web Application (.NET Framework)	Visual C#

Type: Visual C#

Project templates for creating ASP.NET applications. You can create ASP.NET Web Forms, MVC, or Web API applications and add many other features in ASP.NET.

Name: WebApplication1

Location: c:\users\hansha\documents\visual studio 2017\Projects

Solution name: WebApplication1

Browse...

Create directory for solution

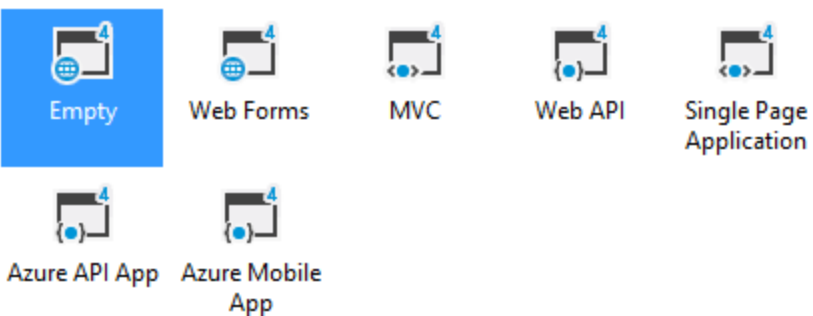
Add to Source Control

OK Cancel

New ASP.NET Web Application - GridView Example

? X

ASP.NET 4.5.2 Templates



An empty project template for creating ASP.NET applications. This template does not have any content in it.

[Learn more](#)

Change Authentication

Authentication: **No Authentication**

Add folders and core references for:

Web Forms MVC Web API

Add unit tests

Test project name:

OK

Cancel



Create a Web Form

The image shows a screenshot of the Visual Studio IDE. In the background, there is a 'Deploy to Azure' dialog box with the text 'Deploy to Azure' and 'Get started with Azure'. Below this, there is a 'New Item...' menu with a red box around it. The 'New Item...' menu is open, showing options like 'Existing Item...', 'New Scaffolded Item...', 'New Folder', 'Add ASP.NET Folder', 'Application Insights Telemetry...', 'Docker Support', 'REST API Client...', and 'New Azure WebJob Project'. The 'Add' option is highlighted in yellow. A secondary context menu is also open, showing options like 'Build', 'Rebuild', 'Clean', 'View', 'Analyze', 'Convert', 'Publish...', 'Overview', 'Scope to This', 'New Solution Explorer View', 'Add', 'Manage NuGet Packages...', 'Manage Bower Packages...', 'Set as StartUp Project', 'Debug', 'Initialize Interactive with Project', 'Source Control', 'Cut', and 'Paste'. The 'Add' option in this menu is also highlighted in yellow. In the top right corner, the 'Solution Explorer' is visible, showing a project named 'GridView Example'. Below the 'Solution Explorer', there is a 'Team Explorer' pane showing 'Project Properties' and 'Debugging' options.

id.

tion

ce:

Deploy to Azure

Get started with Azure

Publish your website to Azure

New Item... Ctrl+Shift+A

Existing Item... Shift+Alt+A

New Scaffolded Item...

New Folder

Add ASP.NET Folder

Application Insights Telemetry...

Docker Support

REST API Client...

New Azure WebJob Project

Build

Rebuild

Clean

View

Analyze

Convert

Publish...

Overview

Scope to This

New Solution Explorer View

Add

Manage NuGet Packages...

Manage Bower Packages...

Set as StartUp Project

Debug

Initialize Interactive with Project

Source Control

Cut Ctrl+X

Paste Ctrl+V

Solution Explorer

Search Solution Explorer (Ctrl+)

Solution 'GridView Example' (1 project)

GridView Example

Team Explorer

Project Properties

Debugging True

Authentication Enabled

Mode Integrated

Add New Item - GridView Example















Installed

Sort by: Default

Search Installed Templates (Ctrl+E)

- Visual C#
 - Code
 - Data
 - General
 - Web
 - General
 - Markup
 - Scripts
 - Web Forms
 - MVC
 - Razor
 - SignalR
 - Web API
 - Windows Forms
 - WPF
- ASP.NET Core
- SQL Server
- Storm Items

Online

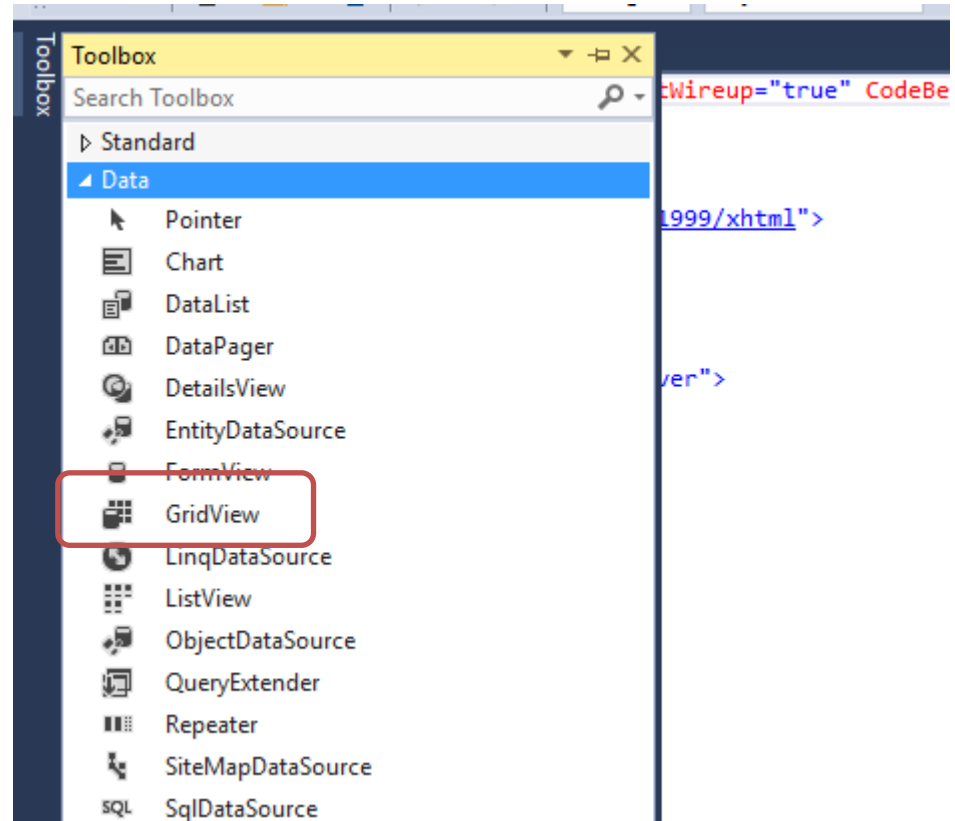
	HTML Page	Visual C#
	JavaScript File	Visual C#
	Style Sheet	Visual C#
	Web Form	Visual C#
	Web Form with Master Page	Visual C#
	MVC 5 View Page (Razor)	Visual C#
	MVC 5 View Page with Layout (Razor)	Visual C#
	Web API Controller Class (v2.1)	Visual C#
	SignalR Hub Class (v2)	Visual C#
	SignalR Persistent Connection Class (v2)	Visual C#
	ASP.NET Handler	Visual C#
	ASP.NET Module	Visual C#
	Browser File	Visual C#
	C# Class Library	Visual C#

Name: WebForm1.aspx

Add Cancel

Type: Visual C#
A form for Web Applications

Create GridView





Connection String

It is recommended that you store the Connection string in Web.Config

Create Connection String in Web Config

```
Web.config [X]
1 <?xml version="1.0" encoding="utf-8"?>
2 <!--
3 For more information on how to configure your ASP.NET application, please visit
4 https://go.microsoft.com/fwlink/?LinkId=169433
5 -->
6 <configuration>
7
8 <system.web>
9 <compilation debug="true" targetFramework="4.5.2"/>
10 <httpRuntime targetFramework="4.5.2"/>
11 </system.web>
12
13
14 <system.codedom>
15 <compilers>
16 <compiler language="c#;cs;csharp" extension=".cs"
17 type="Microsoft.CodeDom.Providers.DotNetCompilerPlatform.CSharpCodeProvider, Microsoft.CodeDom.Providers.DotNetCompilerPlatform, Version=1.0.3.0, Culture=neutral, PublicKeyToken=31bf3856ad364e35" warningLevel="4" compilerOptions="/langversion:6 /nowarn:1659;1699;1701"/>
18 <compiler language="vb;vbs;visualbasic;vbscript" extension=".vb"
19 type="Microsoft.CodeDom.Providers.DotNetCompilerPlatform.VBCodeProvider, Microsoft.CodeDom.Providers.DotNetCompilerPlatform, Version=1.0.3.0, Culture=neutral, PublicKeyToken=31bf3856ad364e35" warningLevel="4" compilerOptions="/langversion:14 /nowarn:41008 /define:_MYTYPE=\"Web\" /optionInfer+"/>
20 </compilers>
21 </system.codedom>
22
23
24
25
26 <connectionStrings>
27 <add name="DatabaseConnectionString_cloud" connectionString="Data Source=xxxxx;Initial Catalog=xxxxx;User Id=xxxx;Password=xxxx;Database=DATALOGGING;"
28 providerName="System.Data.SqlClient" />
29
30 <add name="DatabaseConnectionString" connectionString="Data Source=xxxxx;Initial Catalog=DATALOGGING;Trusted_Connection=True;"
31 providerName="System.Data.SqlClient" />
32 </connectionStrings>
33
34
35 </configuration>
```



Create Class

Create Class

```
using System;
using System.Collections.Generic;
using System.Data.SqlClient;
using System.Configuration;

namespace GridView_Example
{
    public class MeasurementData
    {

        public int MeasurementDataId { get; set; }
        public DateTime MeasurementTimeStamp { get; set; }
        public double MeasurementValue { get; set; }

        public List<MeasurementData> GetMeasurementData()
        {

            string connectionString = ConfigurationManager.ConnectionStrings["DatabaseConnectionString"].ConnectionString;

            List<MeasurementData> measurementDataList = new List<MeasurementData>();

            SqlConnection con = new SqlConnection(connectionString);

            string selectSQL = "select MeasurementDataId, MeasurementTimeStamp, MeasurementValue from GetMeasurementData where MeasurementName ='TC01-1'";

            con.Open();

            SqlCommand cmd = new SqlCommand(selectSQL, con);

            SqlDataReader dr = cmd.ExecuteReader();

            if (dr != null)
            {
                while (dr.Read())
                {
                    MeasurementData measurementData = new MeasurementData();

                    measurementData.MeasurementDataId = Convert.ToInt32(dr["MeasurementDataId"]);
                    measurementData.MeasurementTimeStamp = Convert.ToDateTime(dr["MeasurementTimeStamp"]);
                    measurementData.MeasurementValue = Convert.ToDouble(dr["MeasurementValue"]);

                    measurementDataList.Add(measurementData);
                }
            }

            con.Close();

            return measurementDataList;
        }
    }
}
```

```
WebForm1.aspx [X]
1
2 <%@ Page Language="C#" AutoEventWireup="true" CodeBehind="WebForm1.aspx.cs" Inherits="GridView_Example.WebForm1" %>
3
4 <!DOCTYPE html>
5
6 <html xmlns="http://www.w3.org/1999/xhtml">
7 <head runat="server">
8 <title></title>
9 </head>
10 <body>
11 <form id="form1" runat="server">
12 <div>
13
14 <h1>MeasurementData</h1>
15
16 <asp:GridView ID="gridViewMeasurementData" runat="server">
17 </asp:GridView>
18
19 </div>
20 </form>
21 </body>
22 </html>
23
```

Web Form

body

MeasurementData

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

You find the GridView in the Toolbox

Web Form Code

```
protected void Page_Load(object sender, EventArgs e)
```

```
{
```

```
    FillDataGrid();
```

```
}
```

```
private void FillDataGrid()
```

```
{
```

```
    List<MeasurementData> measurementList = new List<MeasurementData>();
```

```
    MeasurementData measurementData = new MeasurementData();
```

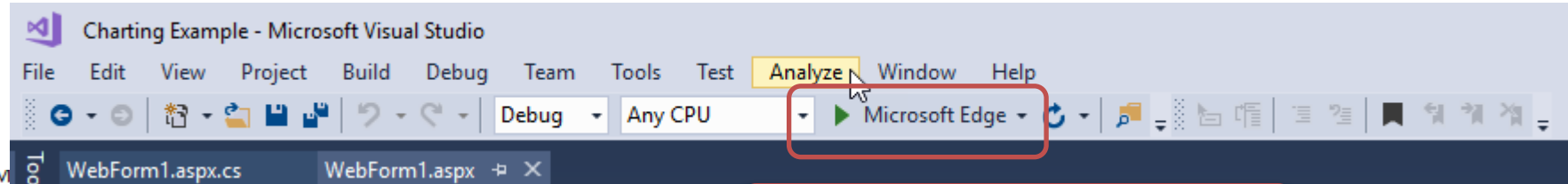
```
    measurementList = measurementData.GetMeasurementData();
```

```
    gridViewMeasurementData.DataSource = measurementList;
```

```
    gridViewMeasurementData.DataBind();
```

```
}
```

Run your Application



Click the green arrow or F5 in order to start the application

Make sure to set your WebForm as Start Page

A screenshot of the context menu for a file named 'WebForm1.aspx' in Visual Studio. The menu items include 'Open', 'Open With...', 'View Code' (F7), 'View Designer' (Shift+F7), 'View Markup', 'View Component Designer', 'View Code Gen File', 'View in Browser (Microsoft Edge)', 'Browse With...', 'Set As Start Page' (highlighted in yellow), 'Scope to This', 'New Solution Explorer View', 'Exclude From Project', 'Cut' (Ctrl+X), 'Copy' (Ctrl+C), 'Delete' (Del), 'Rename', and 'Properties' (Alt+Enter). A red box highlights the 'Set As Start Page' option.

A screenshot of the Team Explorer window showing the file properties for 'WebForm1.aspx'.

Property	Value
Content	~/WebForm1.aspx
Output Directory	Do not copy
Namespace	WebForm1.aspx
Path	C:\Users\har...

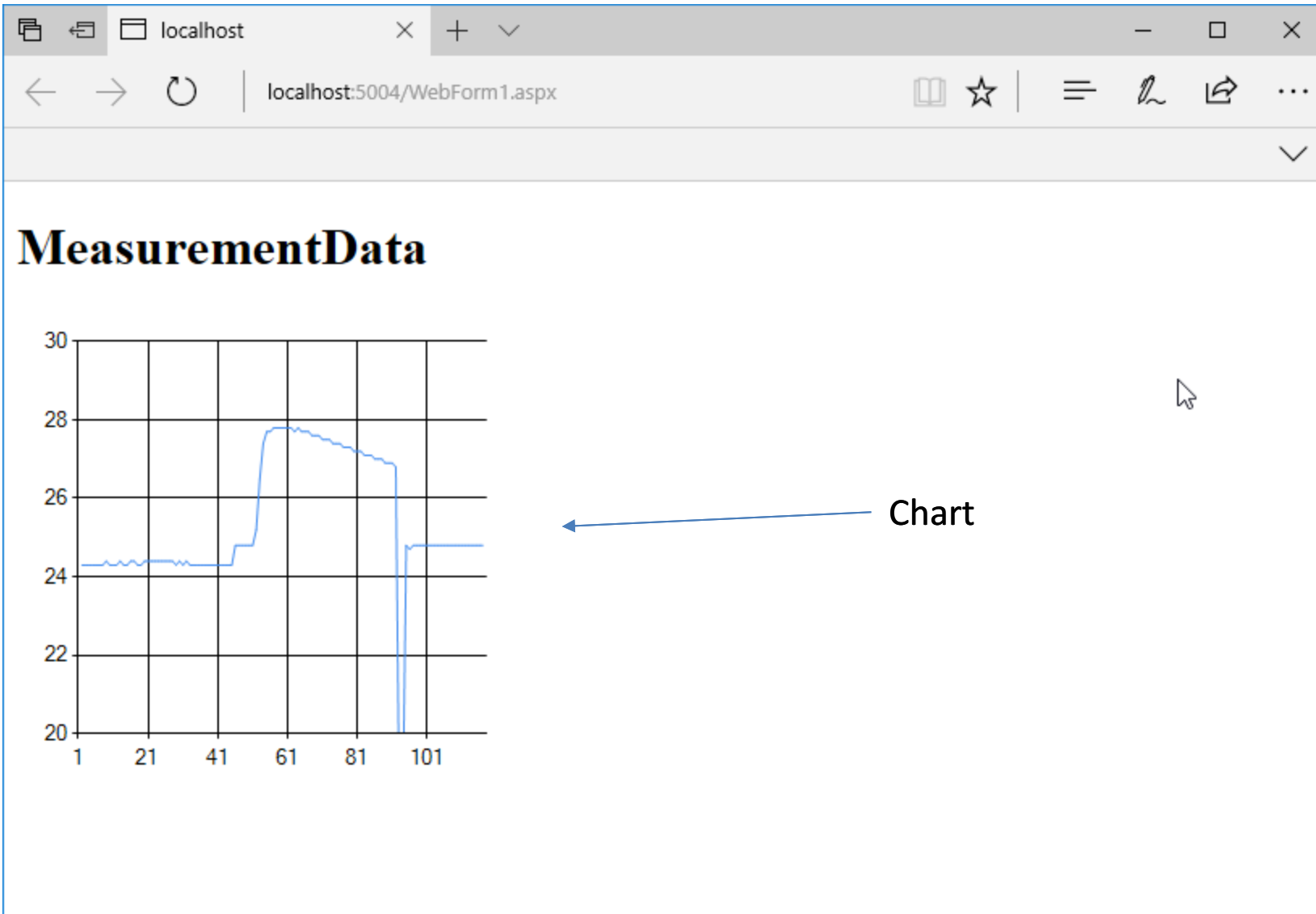
MeasurementData

MeasurementDataId	MeasurementTimeStamp	MeasurementValue
2	2017-08-28 10:22:57	24.3
3	2017-08-28 10:22:58	24.3
4	2017-08-28 10:22:59	24.3
5	2017-08-28 10:23:00	24.3
6	2017-08-28 10:23:01	24.3
7	2017-08-28 10:23:02	24.3
8	2017-08-28 10:23:03	24.3
9	2017-08-28 10:23:04	24.4
10	2017-08-28 10:23:05	24.3
11	2017-08-28 10:23:06	24.3
12	2017-08-28 10:23:07	24.3
13	2017-08-28 10:23:08	24.4
14	2017-08-28 10:23:09	24.3
15	2017-08-28 10:23:10	24.3
16	2017-08-28 10:23:11	24.4
17	2017-08-28 10:23:12	24.4
18	2017-08-28 10:23:13	24.3
19	2017-08-28 10:23:14	24.3
20	2017-08-28 10:23:15	24.4
21	2017-08-28 10:23:16	24.4
22	2017-08-28 10:23:17	24.4
23	2017-08-28 10:23:18	24.4
24	2017-08-28 10:23:19	24.4
25	2017-08-28 10:23:20	24.4
26	2017-08-28 10:23:21	24.4
27	2017-08-28 10:23:22	24.4
28	2017-08-28 10:23:23	24.4
29	2017-08-28 10:23:24	24.3
30	2017-08-28 10:23:25	24.4
31	2017-08-28 10:23:26	24.3
32	2017-08-28 10:23:27	24.4
33	2017-08-28 10:23:28	24.3



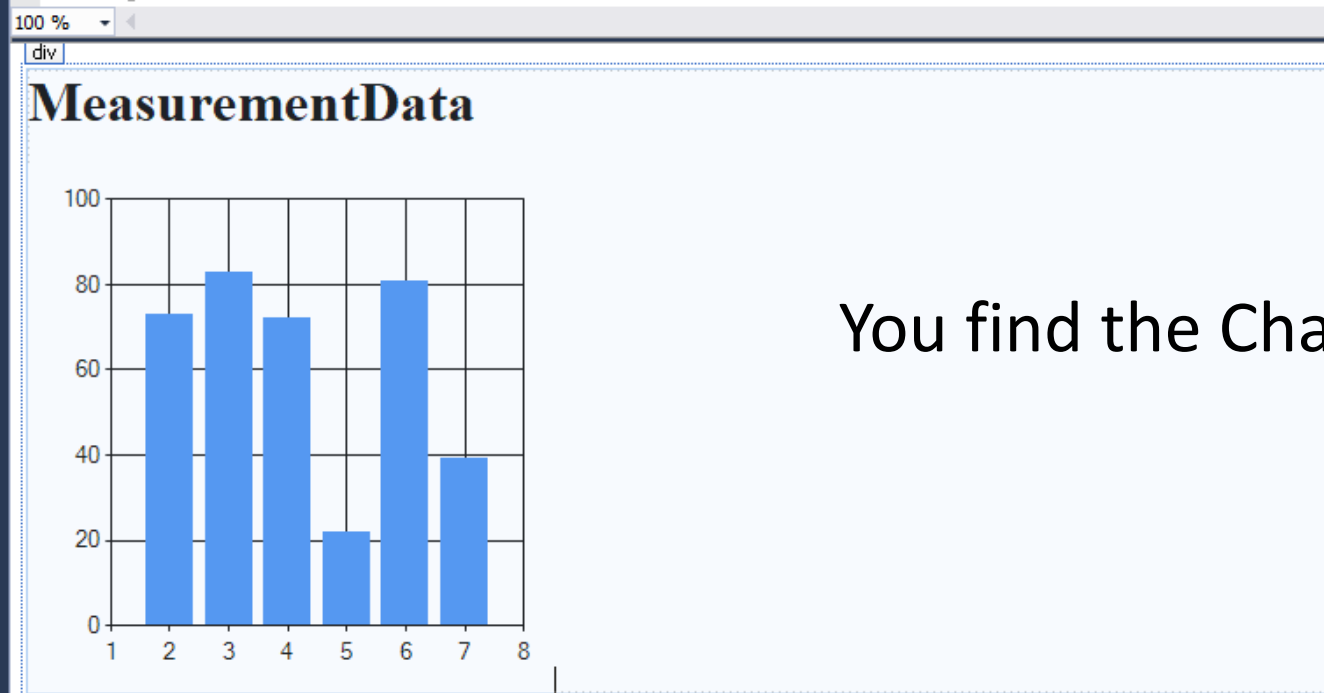
ASP.NET

Charting Example



Web Form

```
WebForm1.aspx
8 <head runat="server">
9   <title></title>
10 </head>
11 <body>
12   <form id="form1" runat="server">
13     <div>
14       <h1>MeasurementData</h1>
15
16       <asp:Chart ID="chartMeasurementData" runat="server">
17         <series>
18           <asp:Series Name="Series1">
19             </asp:Series>
20         </series>
21         <chartareas>
22           <asp:ChartArea Name="ChartArea1">
23             </asp:ChartArea>
24         </chartareas>
25       </asp:Chart>
26
27     </div>
28   </form>
29 </body>
30 </html>
```



You find the Chart in the Toolbox

Web Form Code

```
protected void Page_Load(object sender, EventArgs e)
```

```
{  
    FillChart();  
}
```

```
private void FillChart()
```

```
{
```

```
    chartMeasurementData.Series.Clear();  
    chartMeasurementData.Series.Add("MeasurementData");  
    chartMeasurementData.Series["MeasurementData"].ChartType = SeriesChartType.Line;
```

```
    ChartArea area = chartMeasurementData.ChartAreas[0];  
    area.AxisY.Minimum = 20;  
    area.AxisY.Maximum = 30;
```

```
    List<MeasurementData> measurementList = new List<MeasurementData>();  
    MeasurementData measurementData = new MeasurementData();
```

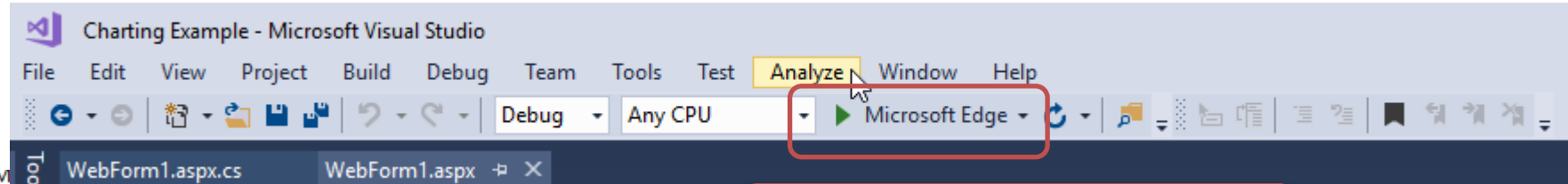
```
    measurementList = measurementData.GetMeasurementData();
```

```
    foreach (MeasurementData data in measurementList)
```

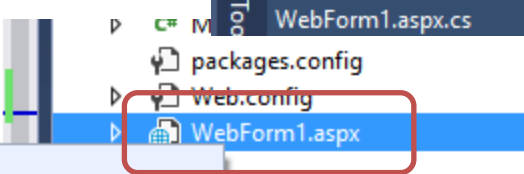
```
    {  
        chartMeasurementData.Series["MeasurementData"].Points.AddXY(data.MeasurementDataId, data.MeasurementValue);  
    }
```

```
}
```

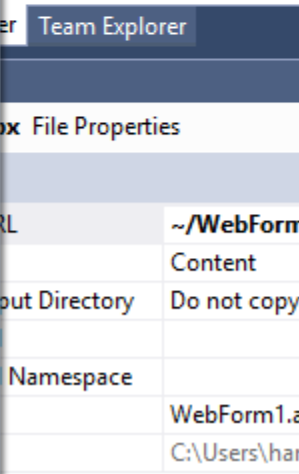
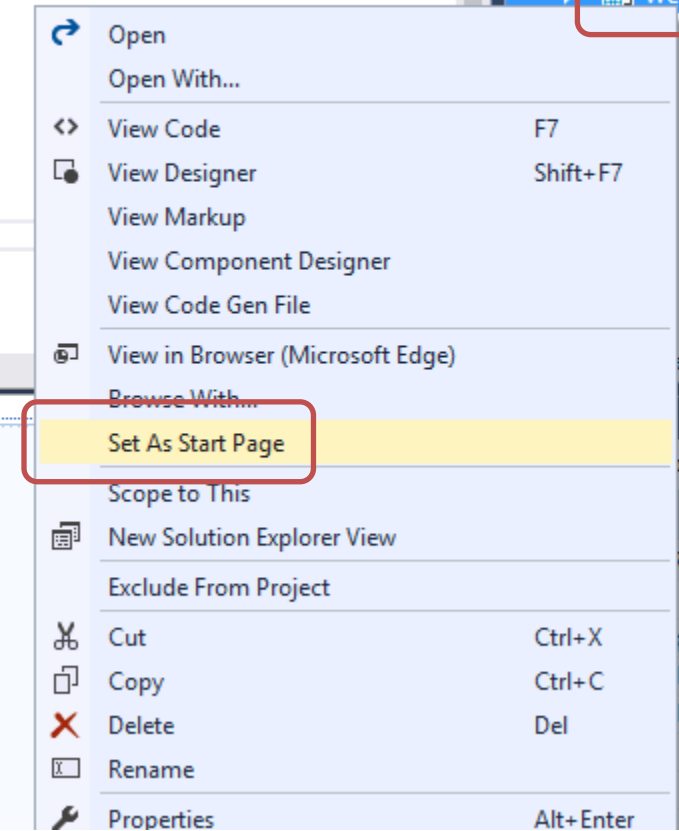
Run your Application



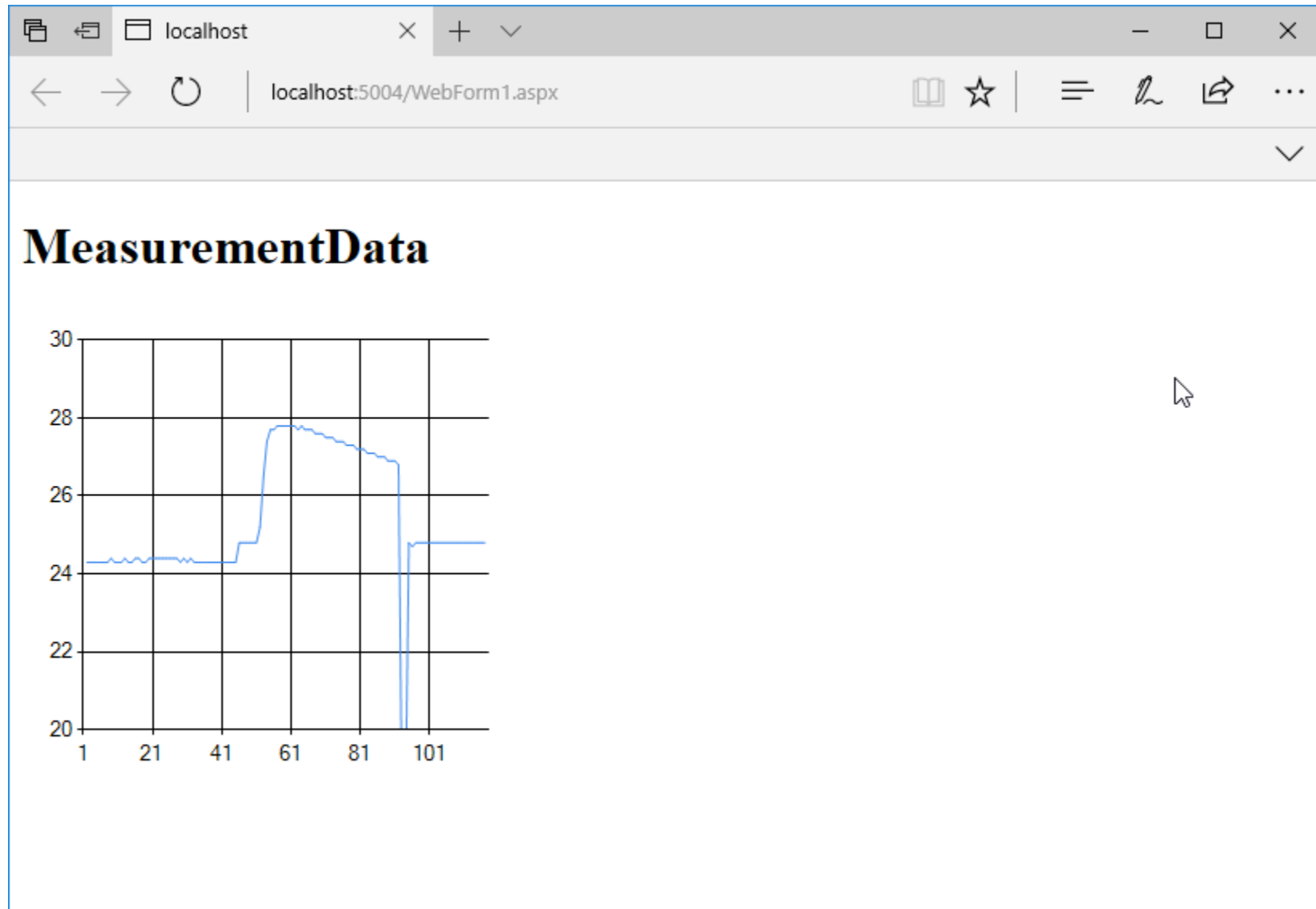
Click the green arrow or F5 in order to start the application



Make sure to set your WebForm as Start Page



Final Results





ASP.NET

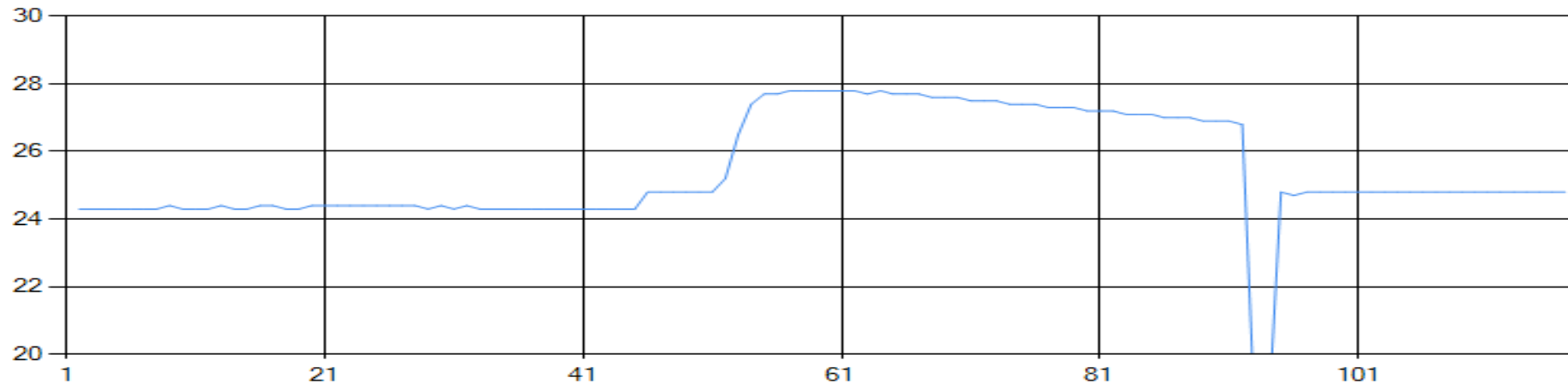
Charting and GridView Example

Monitoring App

- We combine the GridView and Charting Examples

Monitoring App

Charting



Measurement Data

MeasurementDataId	MeasurementTimeStamp	MeasurementValue
2	2017-08-28 10:22:57	24.3
3	2017-08-28 10:22:58	24.3
4	2017-08-28 10:22:59	24.3
5	2017-08-28 10:23:00	24.3
6	2017-08-28 10:23:01	24.3
7	2017-08-28 10:23:02	24.3
8	2017-08-28 10:23:03	24.3
9	2017-08-28 10:23:04	24.4
10	2017-08-28 10:23:05	24.3
11	2017-08-28 10:23:06	24.3
12	2017-08-28 10:23:07	24.3
13	2017-08-28 10:23:08	24.4

In this Example both the Data and the Web App are on my local computer



Cloud-based Data logging

The Cloud

- We have successfully created a local Datalogging and Monitoring System
- The next step is to store the Measurement Data into the Cloud instead of a local Database
- Necessary Steps:
 - Create a Microsoft Azure account
 - Goto the Azure Portal <https://portal.azure.com>
 - Create a Microsoft Azure SQL Server Database and put your Tables, Stored Procedures and Views into the Azure SQL Server Database
 - Change the Connection String for your local Logging App

System Overview

 Visual Studio

Logging
App

Monitoring
App

The Cloud
(Microsoft Azure)

 Visual Studio

ASP.NET

Temperature
Sensor

TC-01 Thermocouple


Table Design

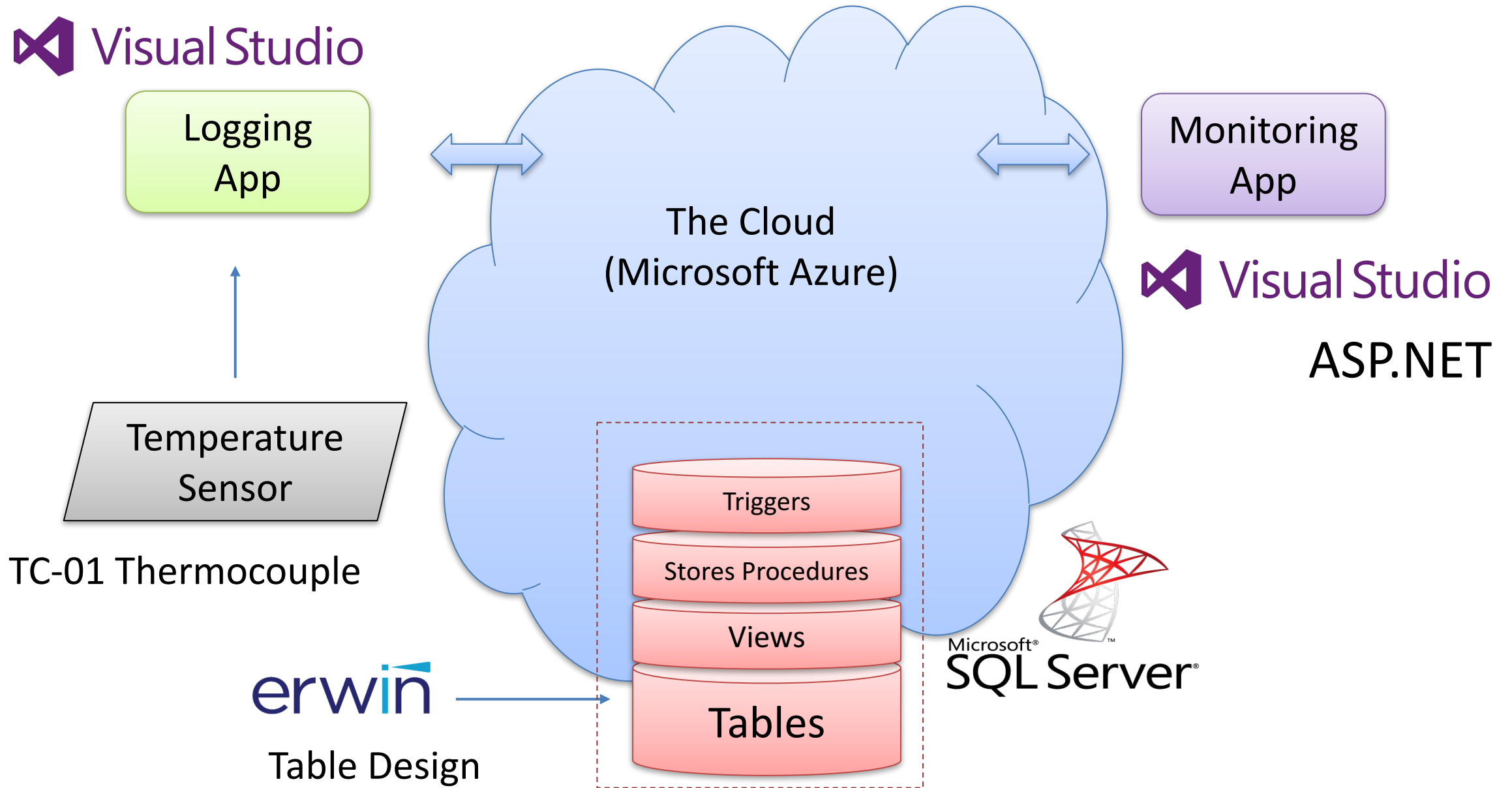
Triggers

Stores Procedures

Views

Tables

Microsoft
SQL Server





Microsoft Azure

Microsoft Azure SQL Database

We need to do the following

- Create Microsoft Azure SQL Server and Database
- Get Connection string
- Give access in Firewall
- Connect to the Database from local SQL Server Management Studio

Microsoft Azure SQL Database

Microsoft Azure SQL databases

hans.p.halvorsen@us...
HANSPHALVORSENUSN (DEF...)

SQL databases
hansphalvorsenusn (Default Directory)

+ Add Columns Refresh

Subscriptions: Microsoft Imagine

Filter by name... All resource groups All locations No grouping

4 items

NAME	STATUS	REPLICATION ROLE	SERVER	PRICING TIER	LOCATION	SUBSCRIPTION
...	Online	None	...	Free	West Central US	Microsoft Imagine
...	Online	None	halvorsen	Free	North Europe	Microsoft Imagine
...	Online	None	...	Free	West Europe	Microsoft Imagine
MEASUREMENTDATA	Online	None	halvorsen	Free	South Central US	Microsoft Imagine

Connection String

The screenshot shows the Microsoft Azure portal interface. At the top, the breadcrumb navigation reads "Microsoft Azure > SQL databases > MEASUREMENTDATA". Below this, the "SQL databases" section for "hansphalvorsenusn (Default Directory)" is visible, with a list of databases including "MEASUREMENTDATA". The "MEASUREMENTDATA" database is selected, and the "Essentials" pane on the right shows its status as "Online" in the "South Central US" region. A "Connection strings" link is highlighted, and a tooltip or dropdown menu is shown with the option "Show database connection strings" selected. An inset window in the foreground displays the "Database connection strings" page for "MEASUREMENTDATA", with the "ADO.NET" tab selected. It shows the ADO.NET (SQL authentication) connection string: `Server=tcp:halvorsendata.database.windows.net,1433;Initial Catalog=MEASUREMENTDATA;Persist Security Info=False;User ID={your_username};Password={your_password};MultipleActiveResultSets=False;Encrypt=True;TrustServerCertificate=False;Connection Timeout=30;` and a link to "Download ADO.NET driver for SQL server".

Firewall

The screenshot displays the Microsoft Azure portal interface for a SQL database. The breadcrumb navigation at the top shows 'Microsoft Azure' > 'SQL databases' > 'MEASUREMENTDATA'. The main header includes 'SQL databases' (hansphalvorsenun (Default Directory)) and 'MEASUREMENTDATA' (SQL database). A search bar is present in the top right corner.

The left sidebar contains navigation icons for home, subscriptions, resource groups, SQL databases, and other services. The main content area is divided into three sections:

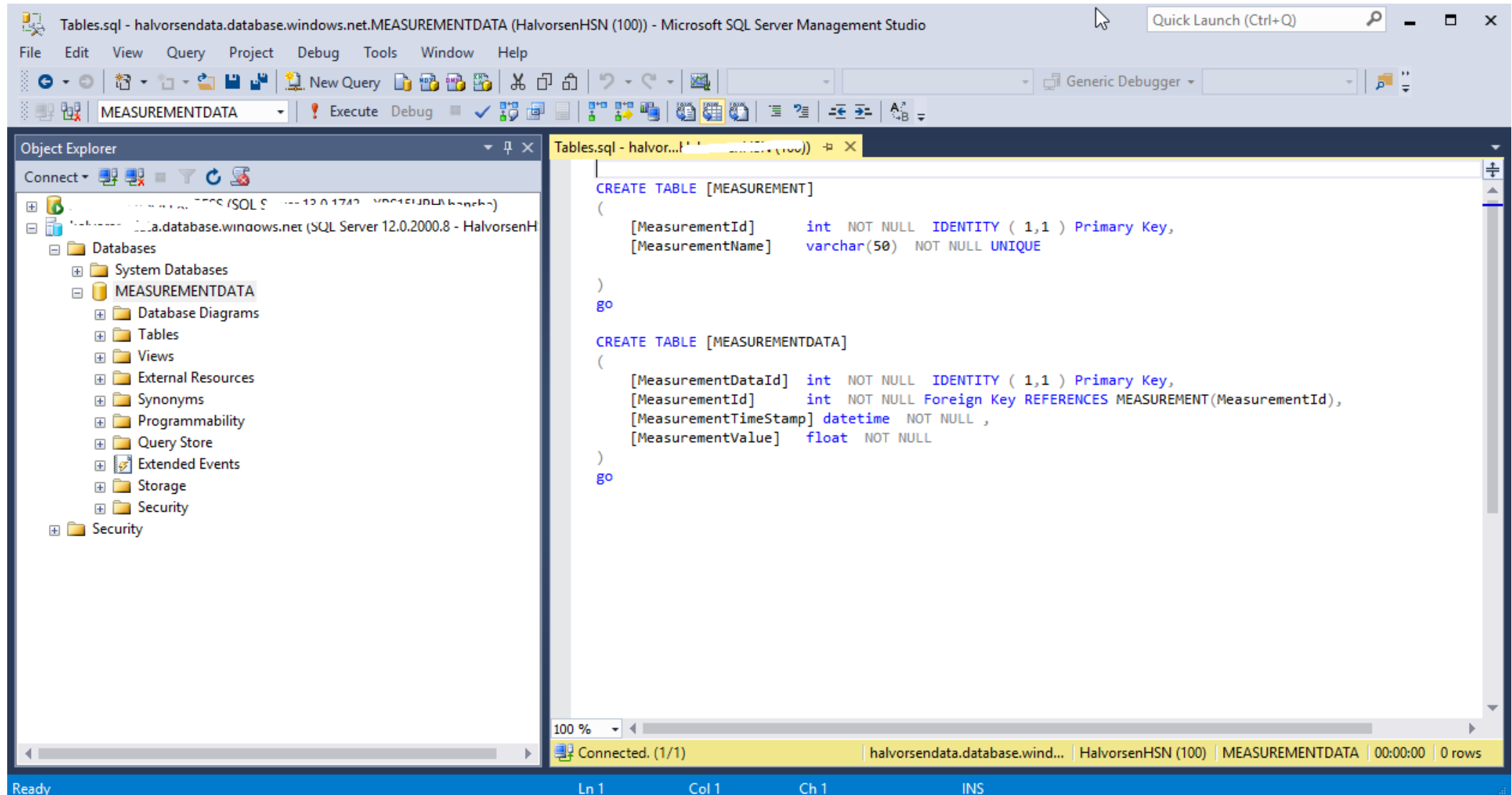
- Subscriptions:** Microsoft Imagine. A filter box is available. Below it, a list of 4 items is shown with a 'NAME' dropdown. The items are: BOOKDB, library, measurementsystem, and MEASUREMENTDATA (highlighted).
- Essentials:** A list of actions: Overview (selected), Activity log, Tags, and Diagnose and solve problems.
- Settings:** Quick start and Pricing tier (scale DTUs).

The right-hand pane shows the 'Tools' menu with 'Set server firewall' highlighted in a red box. Below this, the 'Essentials' section displays the following details:

Resource group	Server name
halvorsen	halvorsendata.database.wii
Status	Connection strings
Online	Show database connection
Location	Pricing tier
South Central US	Free (5 DTUs)
Subscription name	Geo-Replication role
Microsoft Imagine	Not available
Subscription ID	
75ec469f-c646-4c44-b48a-f4711f5d62c4	

The 'Monitoring' section at the bottom shows 'DTU percentage'.

Insert Tables, View and Stored Procedure from Script



The screenshot displays the Microsoft SQL Server Management Studio interface. The title bar indicates the current file is 'Tables.sql' connected to the 'MEASUREMENTDATA' database on the 'HalvorsenHSN (100)' server. The Object Explorer on the left shows the database structure, with 'MEASUREMENTDATA' expanded to show 'Tables'. The main query window contains the following SQL script:

```
CREATE TABLE [MEASUREMENT]
(
    [MeasurementId]    int NOT NULL IDENTITY ( 1,1 ) Primary Key,
    [MeasurementName] varchar(50) NOT NULL UNIQUE
)
go

CREATE TABLE [MEASUREMENTDATA]
(
    [MeasurementDataId] int NOT NULL IDENTITY ( 1,1 ) Primary Key,
    [MeasurementId]    int NOT NULL Foreign Key REFERENCES MEASUREMENT(MeasurementId),
    [MeasurementTimeStamp] datetime NOT NULL ,
    [MeasurementValue] float NOT NULL
)
go
```

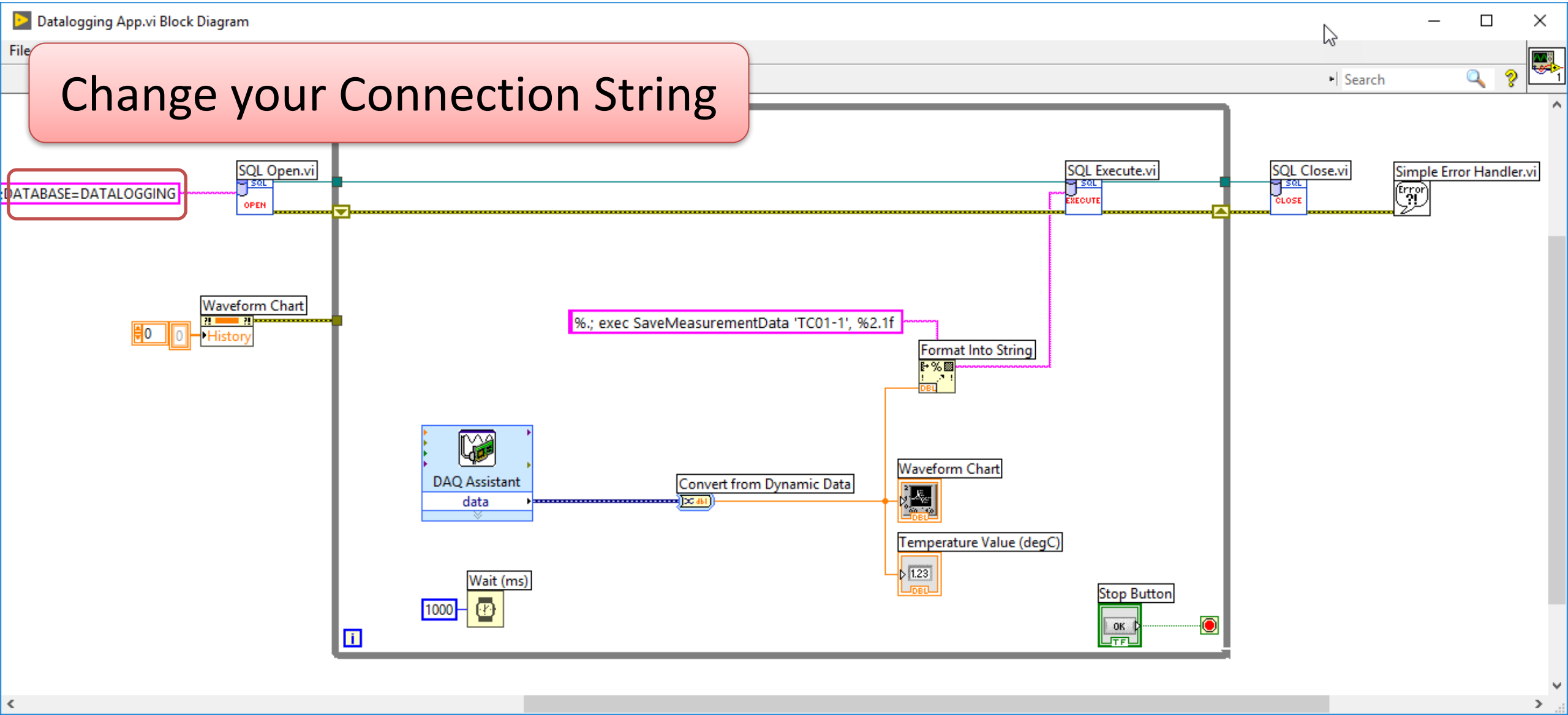
The status bar at the bottom shows the connection is active and the current query is at line 1, column 1.



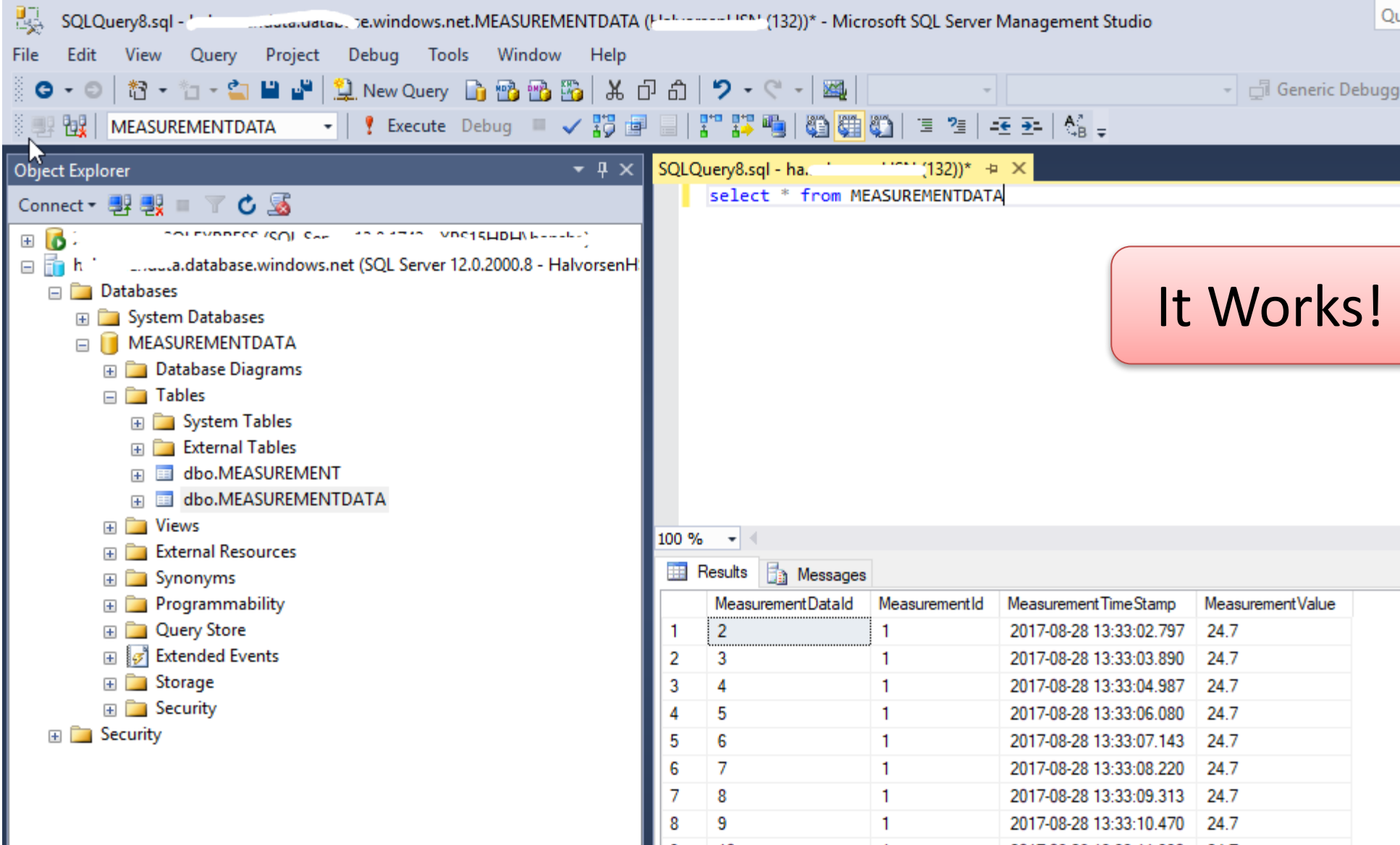
Cloud Data Logging

LabVIEW Example

LabVIEW Example



Check if Data are stored in the Cloud



The screenshot displays the Microsoft SQL Server Management Studio interface. The top menu bar includes File, Edit, View, Query, Project, Debug, Tools, Window, and Help. The toolbar contains various icons for file operations and query execution. The Object Explorer on the left shows the server structure, with the 'MEASUREMENTDATA' database expanded to show its tables, including 'dbo.MEASUREMENTDATA'. The central query editor contains the SQL statement: `select * from MEASUREMENTDATA`. The Results pane at the bottom shows a table with the following data:

	MeasurementDataId	MeasurementId	MeasurementTimeStamp	MeasurementValue
1	2	1	2017-08-28 13:33:02.797	24.7
2	3	1	2017-08-28 13:33:03.890	24.7
3	4	1	2017-08-28 13:33:04.987	24.7
4	5	1	2017-08-28 13:33:06.080	24.7
5	6	1	2017-08-28 13:33:07.143	24.7
6	7	1	2017-08-28 13:33:08.220	24.7
7	8	1	2017-08-28 13:33:09.313	24.7
8	9	1	2017-08-28 13:33:10.470	24.7
9	10	1	2017-08-28 13:33:11.930	24.7

A red callout box with the text "It Works!" is positioned to the right of the query editor.



Cloud Monitoring

Cloud Monitoring

- Example 1:
 - We just change the Connection string for our local Web Monitoring App
 - ..
- Example 2:
 - We Deploy the Web Monitoring App so it is hosted in the Cloud (Microsoft Azure) as well



Cloud Monitoring

Example 1

Change Connection String

- We only need to change the Connection String in Web.config

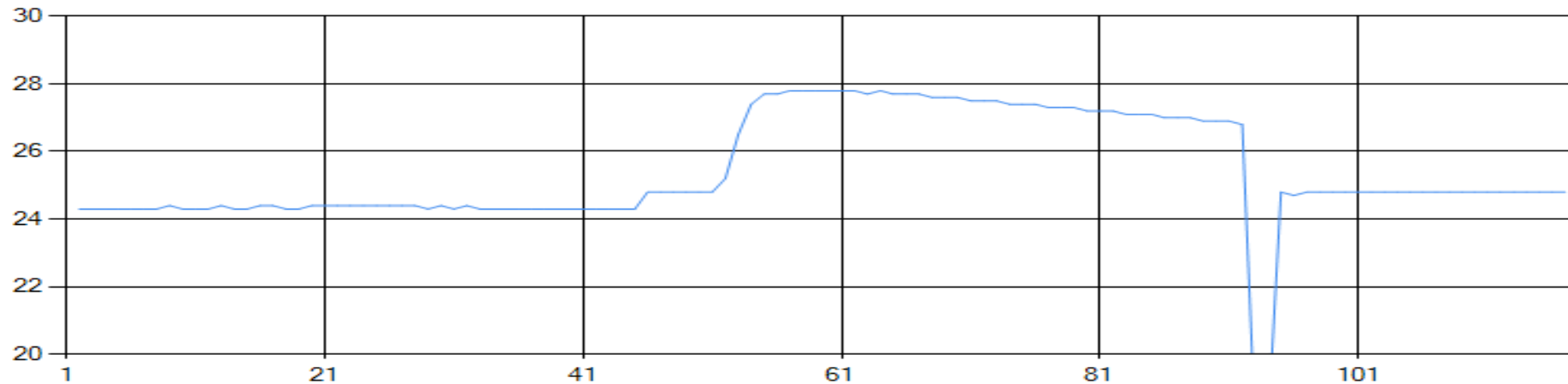
```
<connectionStrings>
```

```
  <add name="DatabaseConnectionString_cloud" connectionString="DATA  
SOURCE=xxx.database.windows.net;UID=xxx;PWD=xxx;DATABASE=xxx"  
providerName="System.Data.SqlClient"/>
```

```
</connectionStrings>
```

Monitoring App

Charting



Measurement Data

MeasurementDataId	MeasurementTimeStamp	MeasurementValue
2	2017-08-28 10:22:57	24.3
3	2017-08-28 10:22:58	24.3
4	2017-08-28 10:22:59	24.3
5	2017-08-28 10:23:00	24.3
6	2017-08-28 10:23:01	24.3
7	2017-08-28 10:23:02	24.3
8	2017-08-28 10:23:03	24.3
9	2017-08-28 10:23:04	24.4
10	2017-08-28 10:23:05	24.3
11	2017-08-28 10:23:06	24.3
12	2017-08-28 10:23:07	24.3
13	2017-08-28 10:23:08	24.4

In this Example we run the Web App locally, but we get the Data from the Cloud (Microsoft Azure)



Cloud Monitoring

Example 2

Cloud Monitoring

- In addition to the SQL Server Database we will also deploy, or install the Web Application as well, in the Cloud (Microsoft Azure)
- In order to deploy or host the Web Application in Microsoft Azure, we need to create a “Web App” using the “App Service” feature in Microsoft Azure

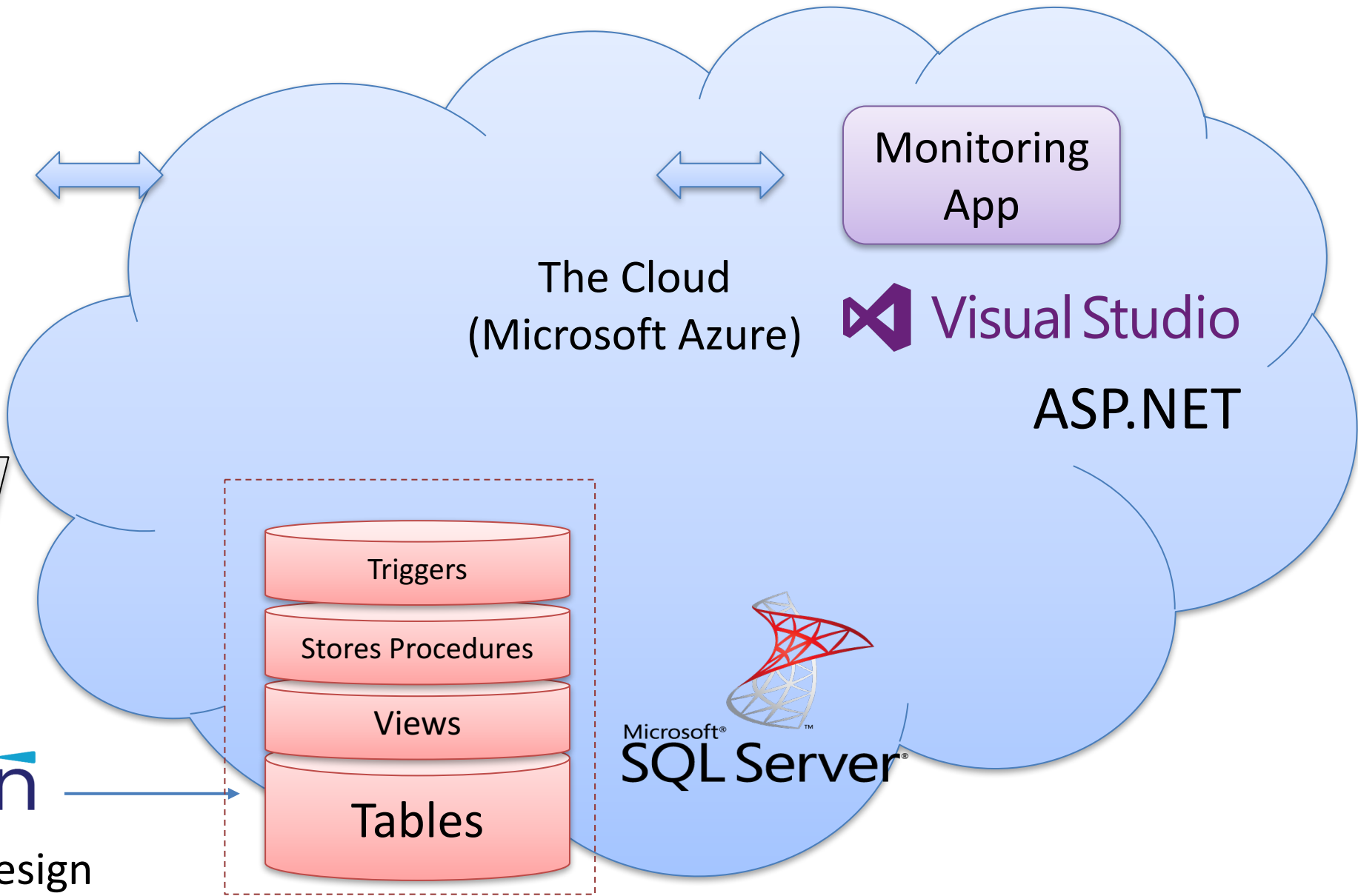


System Overview



Logging App

Monitoring App



Temperature Sensor

TC-01 Thermocouple

erwin
Table Design

Microsoft Azure – App Service

Microsoft Azure App Services

App Services (Default Directory)

+ Add Columns Refresh

Subscriptions: Microsoft Imagine

Filter by name... All resource groups All locations No grouping

4 items

NAME	STATUS	APP TYPE	APP SERVICE PLAN	LOCATION	SUBSCRIPTION
...	Running	Web app	ServicePlan9f7d4df...	South Central US	Microsoft Imagine
...	Running	Web app	ServicePlan9f7d4df...	South Central US	Microsoft Imagine
...	Running	Web app	ServicePlan9f7d4df...	South Central US	Microsoft Imagine
...	Running	Web app	ServicePlan9f7d4df...	South Central US	Microsoft Imagine

Web Apps

After clicking “Add”, select “Web App”



Web App

Microsoft Azure App Services > datamonitoringapp

App Services
hansphalvorsenusn (Default Directory)

datamonitoringapp
App Service

Search resources

Search (Ctrl+)

Subscriptions: Microsoft Imagine

Filter by name...

5 items

NAME

- datamonitoringapp

Essentials

Resource group (change)
datamonitoringapp

Status
Running

Location
South Central US

Subscription (change)
Microsoft Imagine

Subscription ID
7511f5d2c...

URL
<http://datamonitoringapp.azurewebsites.net>

App Service plan/pricing tier
Free

FTP/deployment username
No FTP/deployment user set

FTP hostname
ftp.azurewebsites.net

FTPS hostname
ftps.azurewebsites.net

Http 5xx

Data In

Data Out

HTTP SERVER ERRORS 0

DATA IN 0 B

Then you get a URL like this: <http://datamonitoringapp.azurewebsites.net>

Default Documents

datamonitoringapp - Application settings
App Service

Search (Ctrl+/)

Save Discard

Here you can configure the name for your start page.

Connection strings

No results

Name	Value	SQL Database	Slot setting	...
------	-------	--------------	--------------	-----

Default documents

Default documents	...
Default.htm	...
Default.html	...
Default.asp	...
index.htm	...
index.html	...
iisstart.htm	...
default.aspx	...
index.php	...

I have changed my start page from "WebForm1.aspx" to "Default.aspx" in Visual Studio.
Then I removed all Default documents in the list except "Default.aspx"
Remember to click "Save" afterwards.

DEPLOYMENT

- Quickstart
- Deployment credentials
- Deployment slots
- Deployment options
- Continuous Delivery (Preview)

SETTINGS

- Application settings
- Authentication / Authorization
- Backups
- Custom domains

Handler mappings

No results

Publish

The image shows a screenshot of the Visual Studio IDE. In the top-left corner, the 'Solution Explorer' window displays a project named 'Data Monitoring' with files like 'MeasurementData.cs', 'packages.config', 'Web.config', and 'WebForm1.aspx'. A context menu is open over the project, with the 'Publish...' option highlighted in yellow and a red box around it. Below the menu, the 'Data Monitoring' project window is active, showing a 'Publish' tab selected in the left-hand navigation pane. The main area of the window displays the 'Publish' wizard. The title is 'Publish' with the subtitle 'Publish your app to Azure or another host. [Learn more](#)'. There are three main options represented by icons: 'Microsoft Azure App Service' (highlighted with a red box), 'IIS, FTP, etc', and 'Folder'. Below these options, there are two radio buttons: 'Create New' (selected) and 'Select Existing' (highlighted with a red box). A 'Publish' button is located at the bottom right of the wizard area.

Publish

App Service
Host your web and mobile applications, REST APIs, and more in Azure

Microsoft account
hans.p.halvorsen@usn.no

Subscription
Microsoft Imagine

View
Resource Group

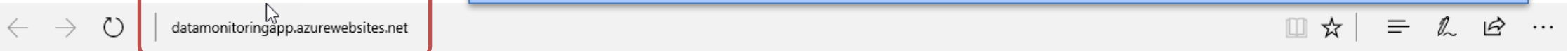
Search

- ▶ bookexamplehalvorsen
- ▶ datamonitoringapp
 - ▶ datamonitoringapp
- ▶ halvorsen

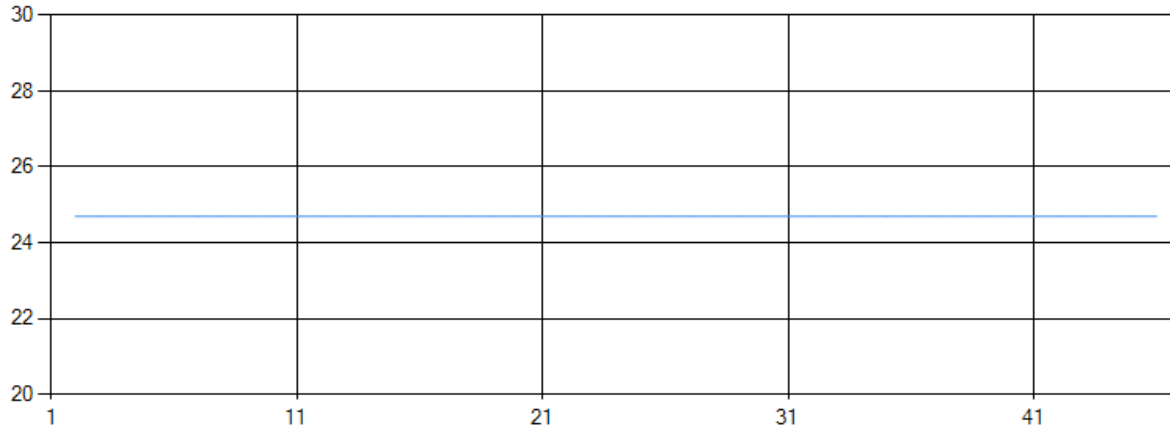
Select your Web App from the list

OK Cancel

Your Web Application is successfully hosted in the Cloud (Microsoft Azure)



Charting



Measurement Data

MeasurementDataId	MeasurementTimeStamp	MeasurementValue
2	8/28/2017 1:33:02 PM	24.7
3	8/28/2017 1:33:03 PM	24.7
4	8/28/2017 1:33:04 PM	24.7
5	8/28/2017 1:33:06 PM	24.7
6	8/28/2017 1:33:07 PM	24.7
7	8/28/2017 1:33:08 PM	24.7
8	8/28/2017 1:33:09 PM	24.7
9	8/28/2017 1:33:10 PM	24.7
10	8/28/2017 1:33:11 PM	24.7
11	8/28/2017 1:33:13 PM	24.7
12	8/28/2017 1:33:14 PM	24.7
13	8/28/2017 1:33:15 PM	24.7

In this Example we run the Web App in the Cloud, and we get the Data from the Cloud (Microsoft Azure)

Errors? Possible Solutions

```
</handlers>
</system.webServer>
<system.web>
  <customErrors mode="Off" />
</system.web>
<httpHandlers>
  <add path="ChartImg.axd" verb="GET" validate="false" />
</httpHandlers>
```

Turn on more descriptive error messages.
Set customErrors mode="Off" in your Web.config File

```
<configuration>
  <appSettings>
    <add key="ChartImageHandler" value="storage=file;timeout=20;dir=c:\TempImageFiles\" />
  </appSettings>
  <system.webServer>
    <validation validateIntegratedModeConfiguration="false" />
    <handlers>
      <remove name="ChartImageHandler" />
    </handlers>
  </system.webServer>
</configuration>
```

```
-->
<configuration>
  <appSettings>
    <add key="ChartImageHandler" value="storage=file;timeout=20;" />
  </appSettings>
  <system.webServer>
    <validation validateIntegratedModeConfiguration="false" />
    <handlers>
```

Remove this part, because this directory
do not exists on the Server



Data Logging

Web API

Web API

- We will improve our Logging App
- Instead of connecting directly to the Database from the Logging App we will create a “Web API” that is hosted in Microsoft Azure.
- The Advantage with this solution is that we don't need to give access to the client from the Firewall in Microsoft Azure.
- Web APIs, REST APIs or Web Services (Dear child has many names 😊) uses HTTP and are therefore Internet-friendly

System Overview

 Visual Studio

Logging App

Web API

Monitoring App

 Visual Studio

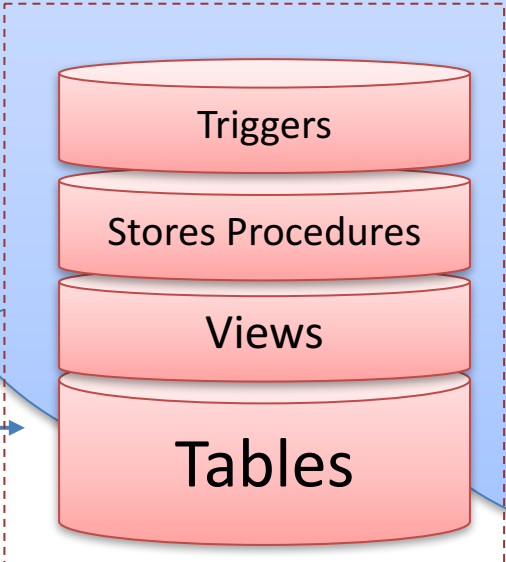
ASP.NET

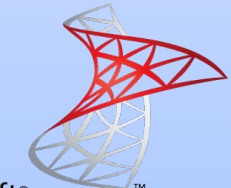
The Cloud
(Microsoft Azure)

Temperature Sensor

TC-01 Thermocouple


Table Design

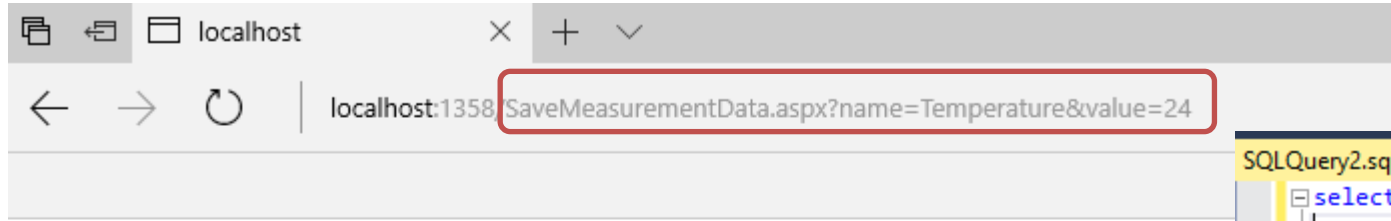



Microsoft
SQL Server

ASP.NET Web API

- We create a simple Web API that we use to store the data instead of communicating directly to the database
- The Web API is created as a simple ASP.NET Web Form Application
- We deploy the Web API the same way we deploy ordinary ASP.NET Applications

Web API Example



Save Measurement Data

Measurement Name:

Measurement Value:

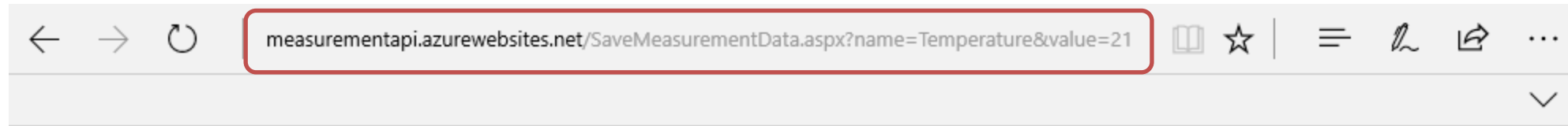
We test the Web API, and we see that data is stored in the Database

```
select * from MEASUREMENT
select * from MEASUREMENTDATA where MeasurementId=2
```

MeasurementId	MeasurementName
1	TC01-1
2	Temperature

MeasurementDataId	MeasurementId	MeasurementTimeStamp	MeasurementValue
119	2	2017-08-30 09:51:35.923	21
120	2	2017-08-30 10:09:38.887	22
121	2	2017-08-30 10:15:54.380	23
122	2	2017-08-30 10:30:28.060	24

We Deploy the Web API to Azure



Save Measurement Data

Measurement Name:

Measurement Value:

Note! Make sure to update Connection string in Web.config



Data Logging

LabVIEW Example

We Modify the Datalogging App

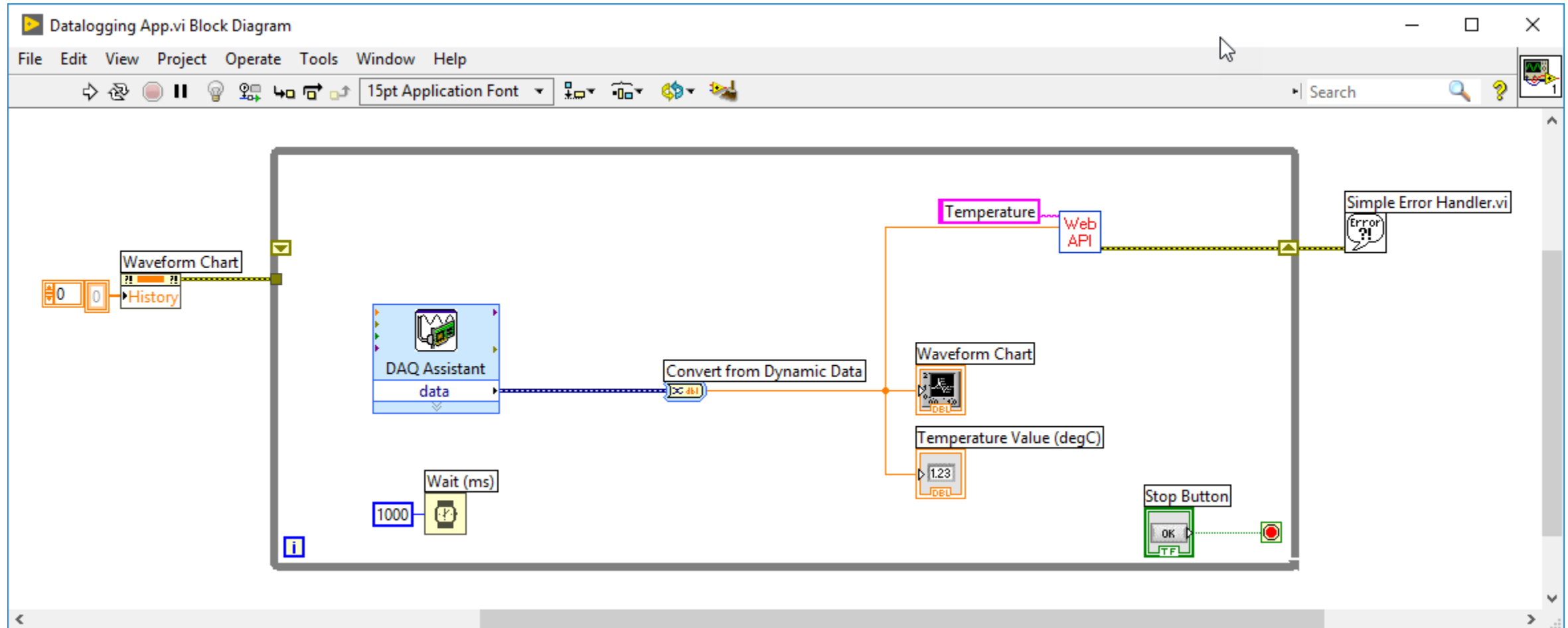
The screenshot displays the LabVIEW interface for a "Datalogging App.vi Front Panel". The main window features a menu bar (File, Edit, View, Project, Operate, Tools, Window, Help) and a toolbar with various icons. A "Waveform Chart" is visible, plotting "Temperature (Celsius Degrees)" on the y-axis (ranging from 20 to 30) against an unlabeled x-axis (ranging from 0 to 10). A blue line on the chart shows a constant temperature of approximately 24.1 degrees Celsius. Below the chart is a "Temperature Value (degC)" display showing "24.1".

Overlaid on the main window is a "Probe Watch Window". This window contains a table with the following data:

Probe(s)	Value	Last Update
Web API.vi		
[3] Probe	"http://measurement.	2017-08-30 11:02:24

To the right of the table is a "Probe Display" area showing the URL: `http://measurementapi.azurewebsites.net/SaveMeasurementData.aspx?name=Temperature&value=24.13`.

Datalogging App using the Web API



Web API SubVI

Web API.vi Front Panel

File Edit View Project Operate Tools Window Help

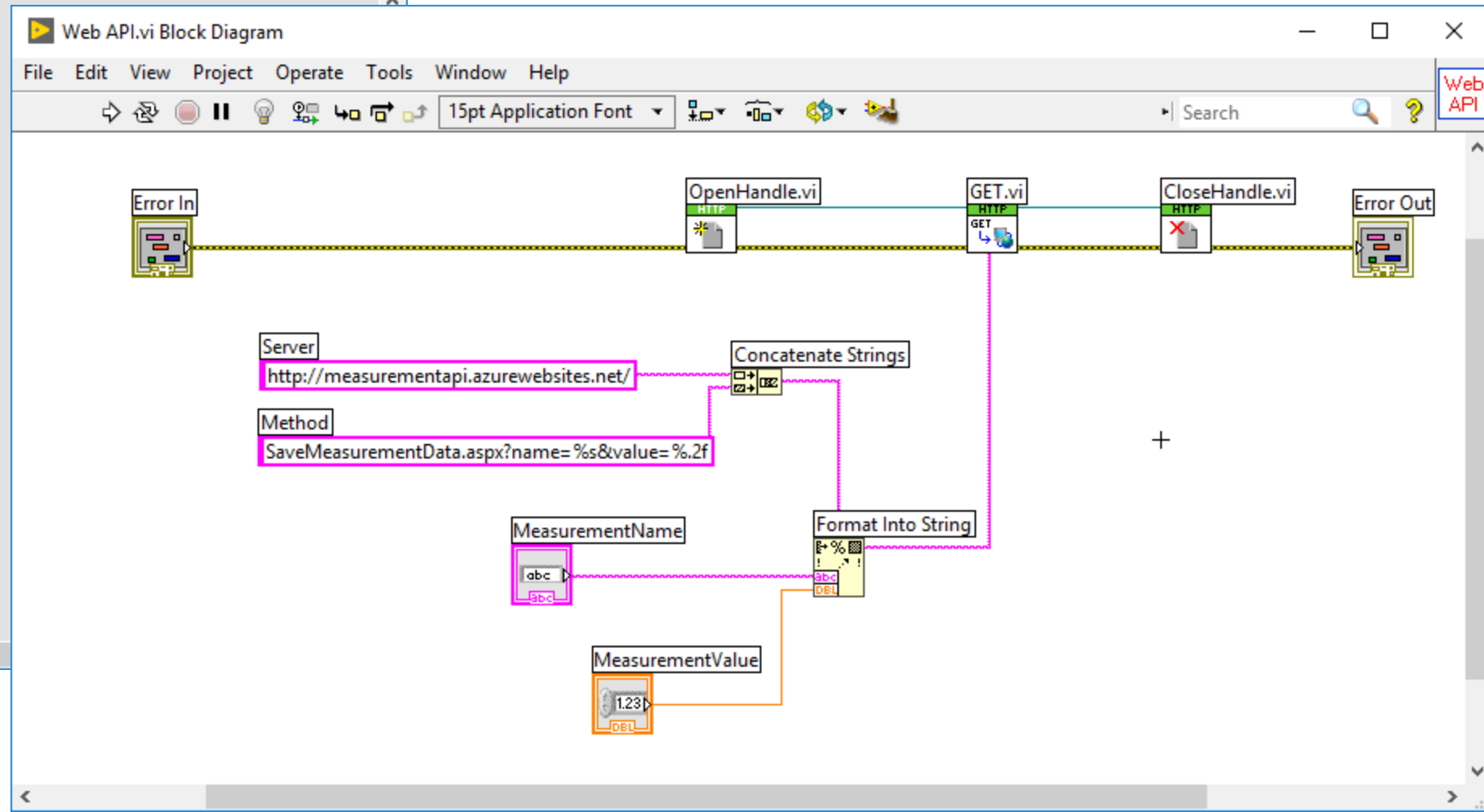
15pt Application Font

MeasurementName

MeasurementValue

Error In

status	code
	0
source	





Data Logging

Visual Studio/C# Example

WinForm Example

Visual Studio/C# Data Logging App with Web API

TC-01 Logging using Web API

Sensor Name:

Data Rate: seconds

Measurement Value:

Time Stamp:

Visual Studio/C# Code

```
void SaveMeasurementData()
{
    string sensorName;
    double measurementValue;

    sensorName = txtSensorName.Text;
    measurementValue = Convert.ToDouble(txtMeasurementValue.Text);

    string server = "http://measurementapi.azurewebsites.net/";
    string webMethod;
    string uri;

    var webclient = new WebClient();

    webMethod = "SaveMeasurementData.aspx?name=" + sensorName + "&value=" + measurementValue;

    uri = server + webMethod;

    webclient.UploadString(uri, "POST", "");
}
```

The Code is almost identical as previous Visual Studio/C# example. The only thing that is changed is the SaveMeasurementData() Method

Hans-Petter Halvorsen

University of Southeast Norway

www.usn.no

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

Web: <http://www.halvorsen.blog>

