https://www.halvorsen.blog

ASP.NET Core

appSettings.json

Hans-Petter Halvorsen

Introduction

- **appSettings.json** is a configuration file used in ASP.NET Core Web Applications
- It is typically used to store the Connection String to the Database
- But it can be used to store lots of other settings that you need to use in your application

ASP.NET Core

If you have never used ASP.NET Core, I suggest the following Videos:

- ASP.NET Core - Hello World
  https://youtu.be/lcQsWYgQXK4
- ASP.NET Core – Introduction
  https://youtu.be/zkOtiBcwo8s

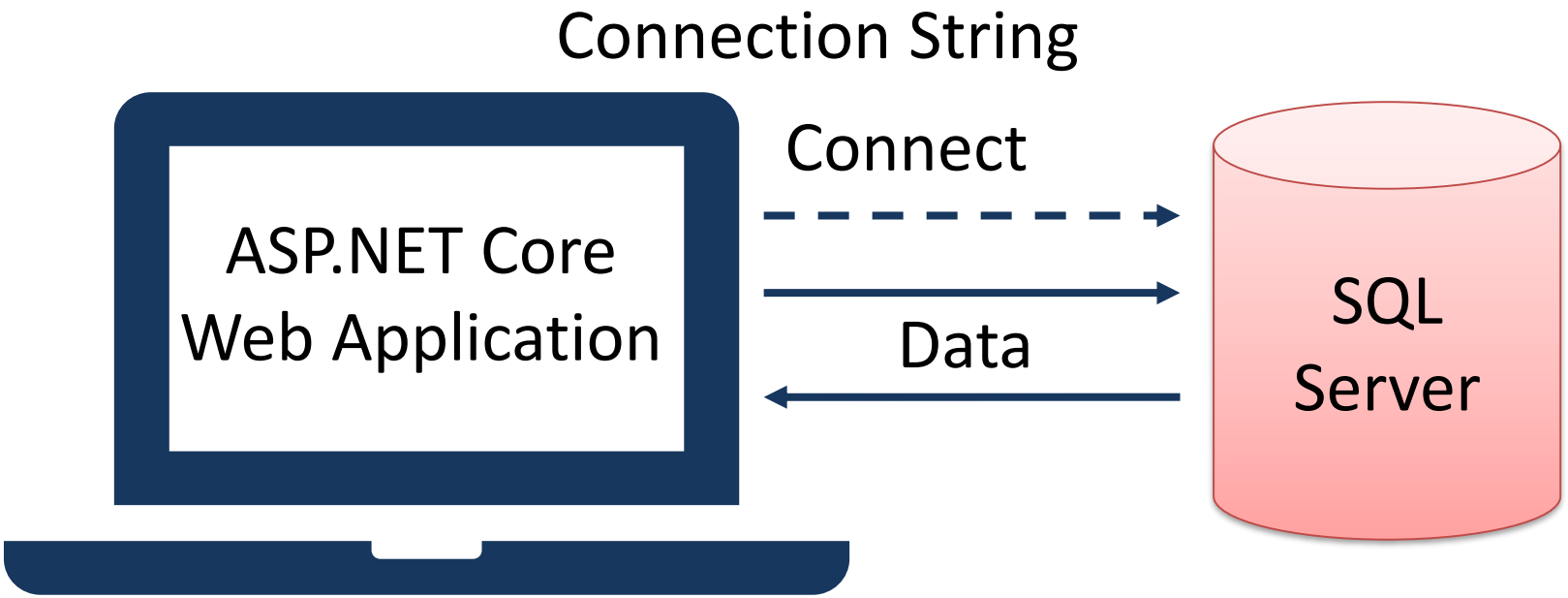
ASP.NET Core Resources:
https://halvorsen.blog/documents/programming/web/aspnet

https://www.halvorsen.blog

Connection String

Hans-Petter Halvorsen

Connection String




```
ConnectionString":"DATA SOURCE=xxx;UID=xxx;PWD=xxx;DATABASE=xxx
```

appSettings.json

```
{
  "Logging": {
    "LogLevel": {
      "Default": "Information",
      "Microsoft": "Warning",
      "Microsoft.Hosting.Lifetime": "Information"
    }
  },
  "AllowedHosts": "*",

  "ConnectionStrings": {
    "ConnectionString": "DATA SOURCE=xxx;UID=xxx;PWD=xxx;DATABASE=xxx"
  }

}
```

Startup.cs

We need to add something to the “**Startup.cs**” file:

```
public void ConfigureServices(IServiceCollection services)
{
    services.AddRazorPages();

    services.AddSingleton<IConfiguration>(Configuration);
}
```

We have added:
services.AddSingleton<IConfiguration>(Configuration);

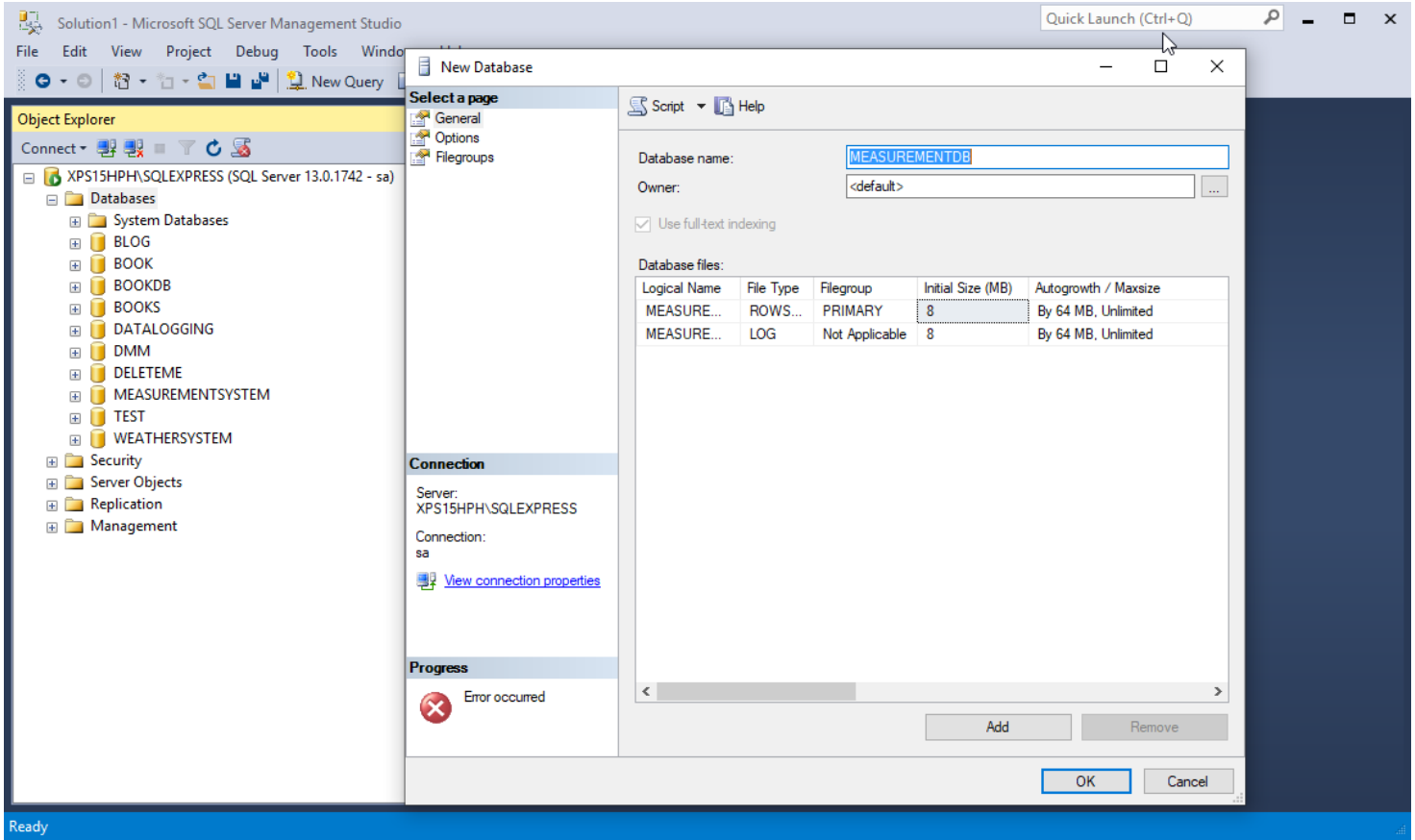
SQL Server

SQL Server

- We will use SQL Server in this example as our database.
- You should have SQL Server locally installed on your computer
- SQL Server Express is recommended.

Database

SQL Server - Create Database

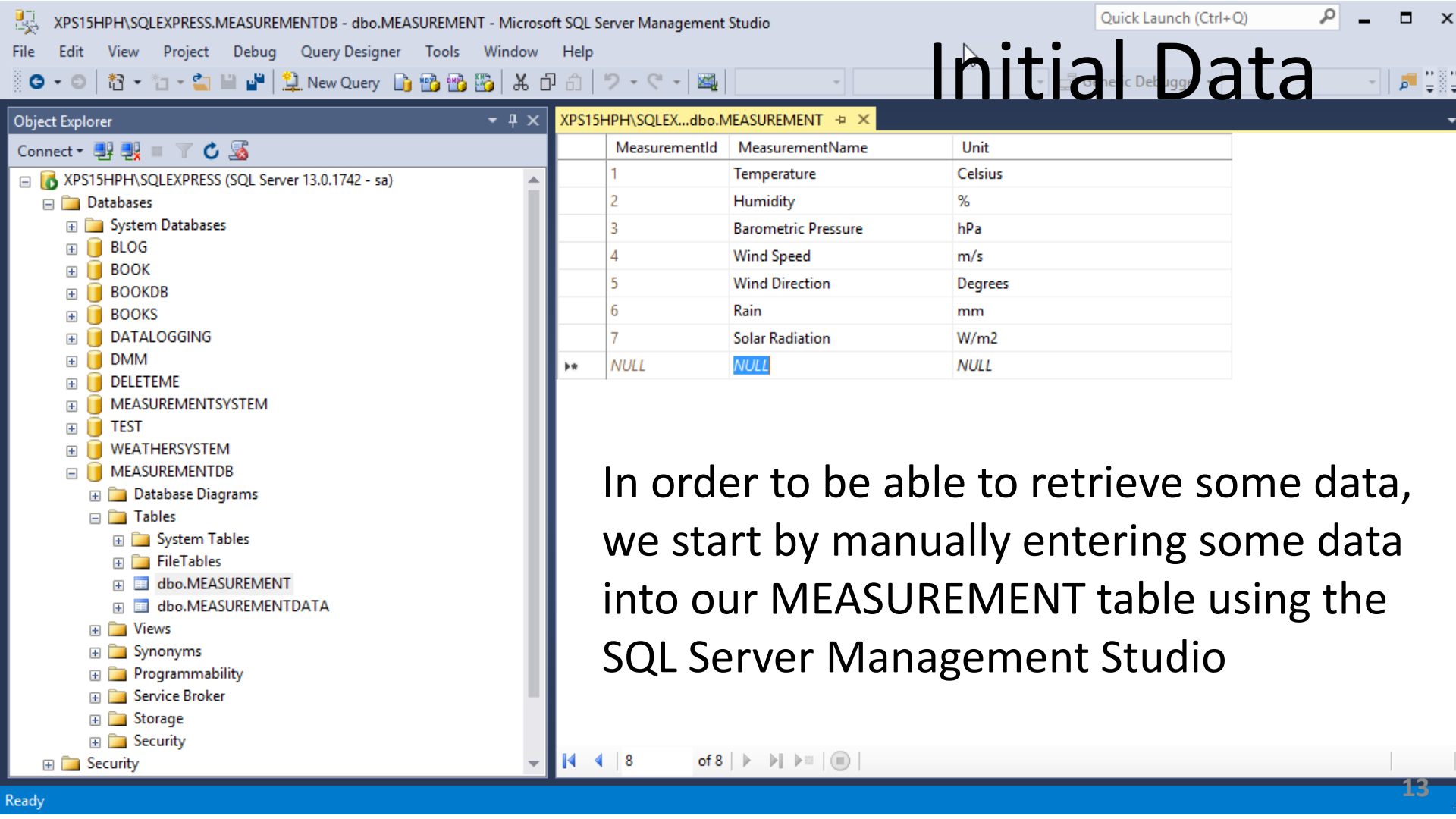


Database Table

```
CREATE TABLE [MEASUREMENT]
(
    [MeasurementId]   int  NOT NULL  IDENTITY ( 1,1 ) Primary Key,
    [MeasurementName] varchar(100) NOT NULL UNIQUE,
    [Unit]            varchar(50)  NULL
)
go
```

You can use SQL Server Management Studio in order to run this SQL Script

Initial Data

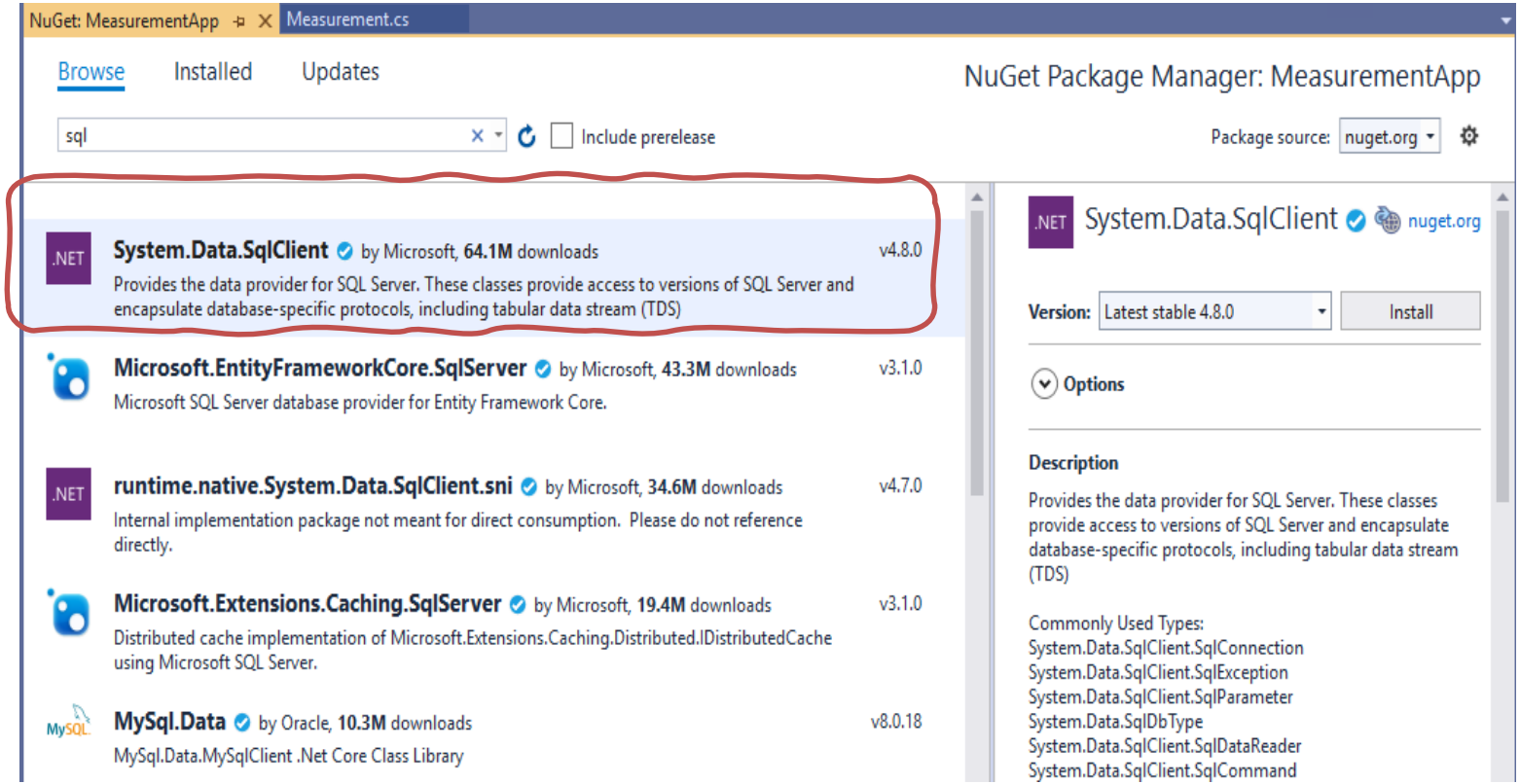


In order to be able to retrieve some data, we start by manually entering some data into our MEASUREMENT table using the SQL Server Management Studio

Visual Studio

NuGet

Make sure to install the necessary NuGet package(s). We will use the **System.Data.SqlClient**



appSettings.json

```
{
  "Logging": {
    "LogLevel": {
      "Default": "Information",
      "Microsoft": "Warning",
      "Microsoft.Hosting.Lifetime": "Information"
    }
  },
  "AllowedHosts": "*",

  "ConnectionStrings": {
    "ConnectionString": "DATA SOURCE=xxx\\SQLEXPRESS;UID=sa;PWD=xxx;DATABASE=xxx"
  }

}
```

C# Code

```
…
using Microsoft.Extensions.Configuration;
public class xxxModel : PageModel
{
    readonly IConfiguration _configuration;

    private string connectionString;

    public xxxModel(IConfiguration configuration)
    {
        _configuration = configuration;
    }
    …
    connectionString =
    _configuration.GetConnectionString("ConnectionString");

}
```

The Constructor

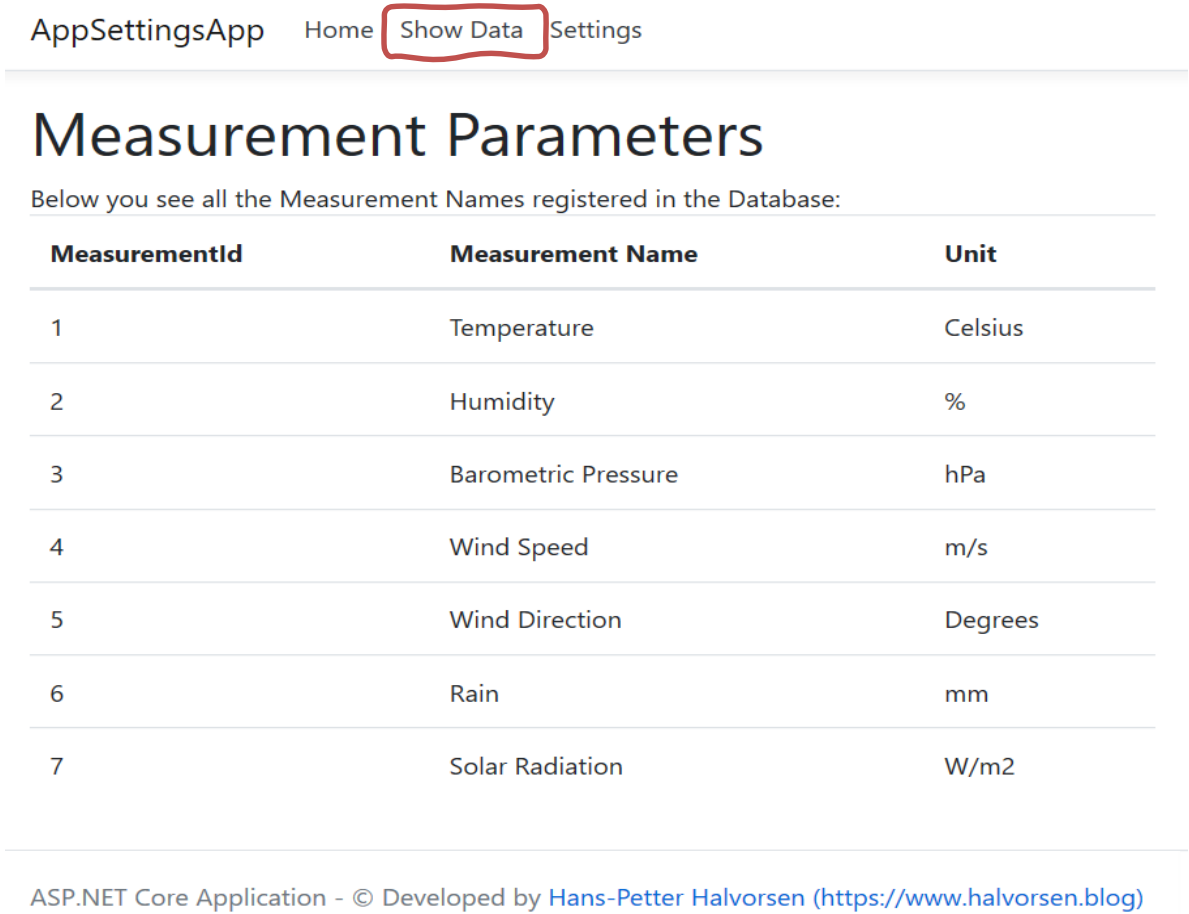
Demo

Connection String in appSettings.json

ASP.NET Core Web Application

The following Application will be demonstrated here:

We will retrieve these data from a SQL Server Database



Create Database Class

Measurement.cs

AppSettingsApp — AppSettingsApp.Models.Measurement

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

namespace AppSettingsApp.Models
{
    9 references
    public class Measurement
    {
        2 references
        public int MeasurementId { get; set; }
        2 references
        public string MeasurementName { get; set; }
        2 references
        public string MeasurementUnit { get; set; }

        1 reference
        public List<Measurement> GetMeasurmentParameters(string connectionString)
        {

            List<Measurement> measurementParameterList = new List<Measurement>();

            SqlConnection con = new SqlConnection(connectionString);

            string sqlQuery = "select MeasurementId, MeasurementName, Unit from MEASUREMENT";

            con.Open();

            SqlCommand cmd = new SqlCommand(sqlQuery, con);

            SqlDataReader dr = cmd.ExecuteReader();

            if (dr != null)
            {
                while (dr.Read())
                {
                    Measurement measurmentParameter = new Measurement();

                    measurmentParameter.MeasurementId = Convert.ToInt32(dr["MeasurementId"]);
                    measurmentParameter.MeasurementName = dr["MeasurementName"].ToString();
                    measurmentParameter.MeasurementUnit = dr["Unit"].ToString();

                    measurementParameterList.Add(measurmentParameter);
                }
            }
```

- We start by creating a **Models** folder in our project using the Solutions Explorer
- Then we create a new Class (“**Measurement.cs**”)
- Then we create C# Code for retrieving data from the Database

“Measurement.cs”

```
using System.Data.SqlClient;

namespace MeasurementApp.Model
{
    public class Measurement
    {
        public int MeasurementId { get; set; }
        public string MeasurementName { get; set; }
        public string MeasurementUnit { get; set; }

        public List<Measurement> GetMeasurmentParameters(string connectionString)
        {
            List<Measurement> measurementParameterList = new List<Measurement>();

            SqlConnection con = new SqlConnection(connectionString);

            string sqlQuery = "select MeasurementId, MeasurementName, Unit from MEASUREMENT";

            con.Open();

            SqlCommand cmd = new SqlCommand(sqlQuery, con);

            SqlDataReader dr = cmd.ExecuteReader();

            if (dr != null)
            {
                while (dr.Read())
                {
                    Measurement measurmentParameter = new Measurement();

                    measurmentParameter.MeasurementId = Convert.ToInt32(dr["MeasurementId"]);
                    measurmentParameter.MeasurementName = dr["MeasurementName"].ToString();
                    measurmentParameter.MeasurementUnit = dr["Unit"].ToString();

                    measurementParameterList.Add(measurmentParameter);
                }
            }
            return measurementParameterList;
        }
    }
}
```

An ASP.NET Core Web Page consist of the following:

- “Database.**cshtml”** - HTML/Razor code
- “Database.**cshtml.cs”** - Page Model (Code behind C# File)

“Database.cshtml.cs”

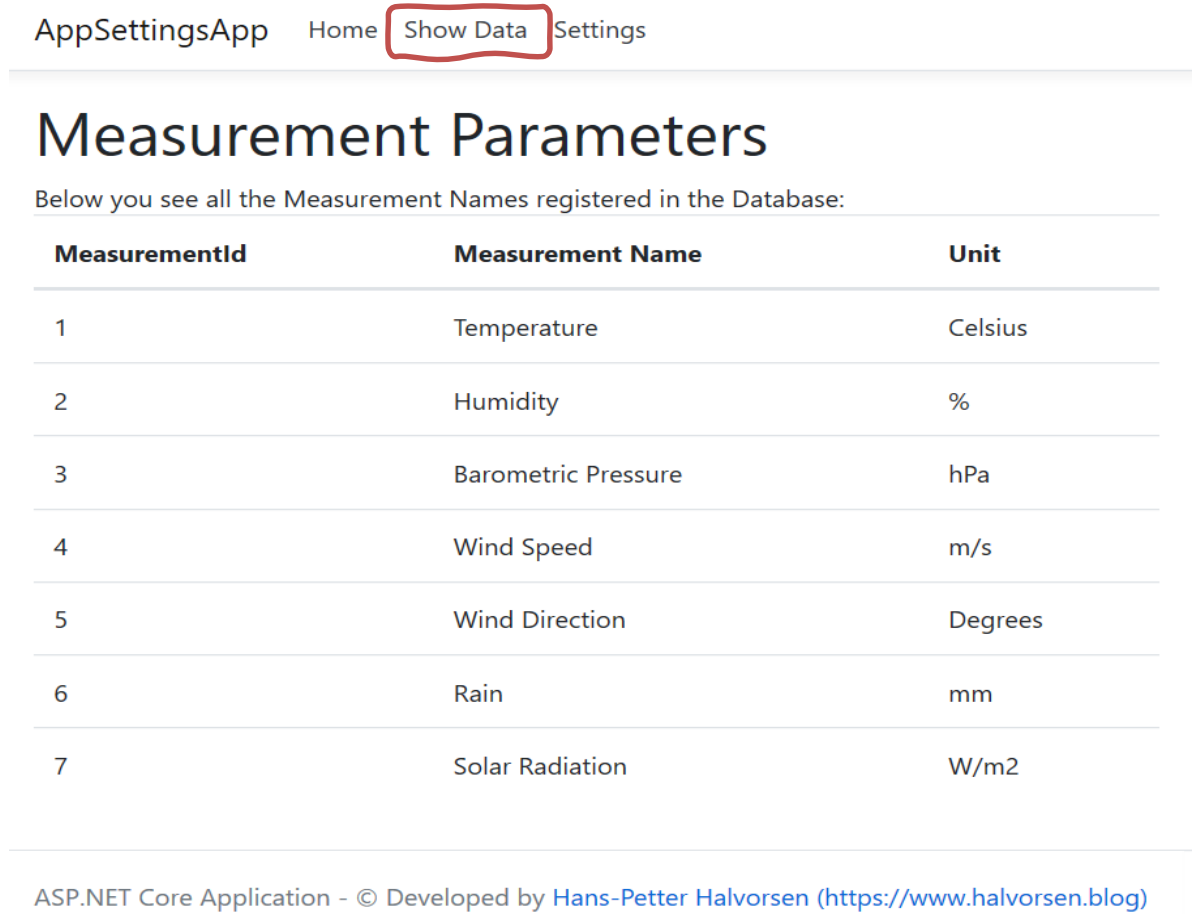
```
…
using Microsoft.Extensions.Configuration;
using AppSettingsApp.Models;

namespace AppSettingsApp.Pages
{
   public class DatabaseModel : PageModel
   {
      readonly IConfiguration _configuration;

      public List<Measurement> measurementParameterList = new List<Measurement>();

      public string connectionString;

      public DatabaseModel(IConfiguration configuration)
      {
         _configuration = configuration;
      }
      public void OnGet()
      {
         GetData();
      }

      void GetData()
      {
            Measurement measurement = new Measurement();

            connectionString = _configuration.GetConnectionString("ConnectionString");

            measurementParameterList = measurement.GetMeasurmentParameters(connectionString);
      }
   }
}
```

“Database.cshtml”

```
...

<div>

   <h1>Measurement Parameters</h1>

   Below you see all the Measurement Names registered in the Database:

   <table class="table">
      <thead>
         <tr>
            <th>MeasurementId</th>
            <th>Measurement Name</th>
            <th>Unit</th>
         </tr>
      </thead>
      <tbody>
         @foreach (var measurement in Model.measurementParameterList)
         {
            <tr>
               <td> @measurement.MeasurementId</td>
               <td> @measurement.MeasurementName</td>
               <td> @measurement.MeasurementUnit</td>
            </tr>
         }
      </tbody>
   </table>

</div>
```

Run the Application

Now we can run the Application



https://www.halvorsen.blog

Get Configuration Data from appSettings.json in your C# Code

Hans-Petter Halvorsen

appSettings.json

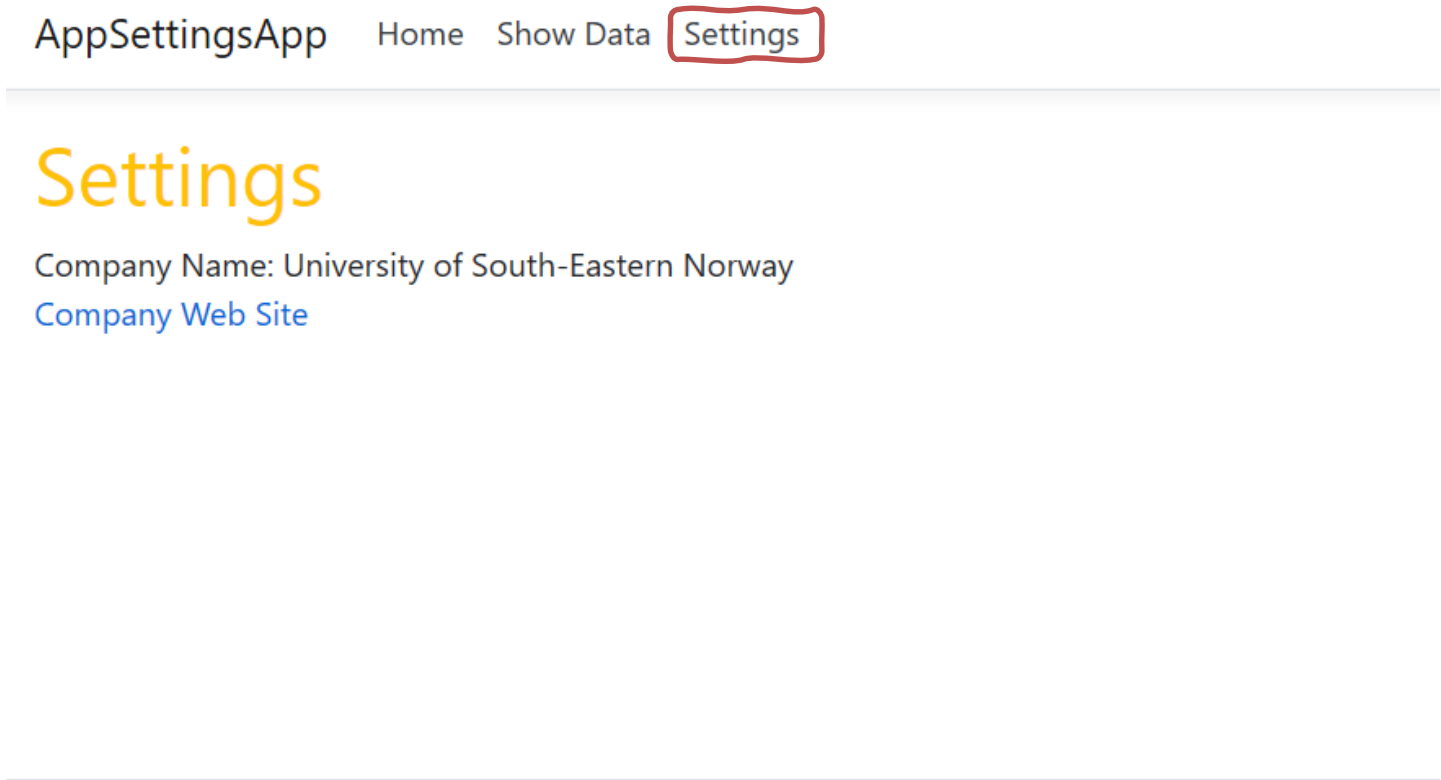
```
{
  …

  "ConnectionStrings": {
    "ConnectionString": "DATA SOURCE=xxx\\SQLEXPRESS;UID=sa;PWD=xxx;DATABASE=VOTINGSYSTEM"
  },

  "Company": {
    "CompanyName": "University of South-Eastern Norway",
    "WebSite": "https://www.usn.no/english/"
  },

  "Appearance": {
    "BackColor": "warning"
  }
}
```

C# Code

```
string companyName;
…

companyName = _configuration.GetSection(" Company").GetValue<string>(" CompanyName ");

…
```

Demo

ASP.NET Core Web Application



An ASP.NET Core Web Page consist of the following:

- “Settings.**cshtml**” - HTML/Razor code
- “Settings.**cshtml.cs**” - Page Model (Code behind C# File)

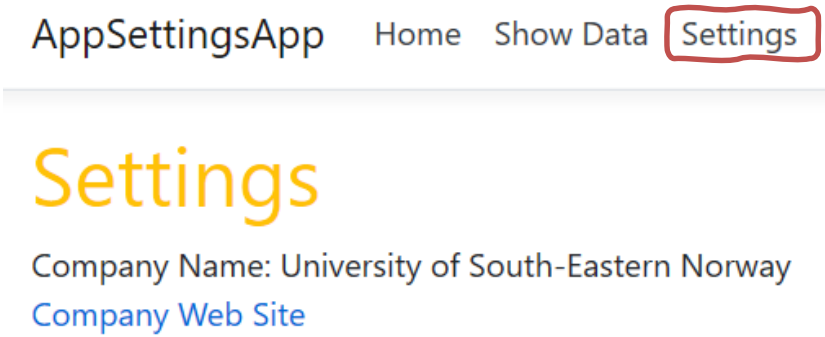
“Settings.cshtml.cs”

```
…
using Microsoft.Extensions.Configuration;

namespace AppSettingsApp.Pages
{
   public class SettingsModel : PageModel
   {
      readonly IConfiguration _configuration;

      public string companyName;
      public string webSite;
      public string backColor;

      public SettingsModel(IConfiguration configuration)
      {
         _configuration = configuration;
      }

      public void OnGet()
      {
         GetAppSettings();
      }

      void GetAppSettings()
      {
         companyName = _configuration.GetSection("Company").GetValue<string>("CompanyName");

         webSite = _configuration.GetSection("Company").GetValue<string>("WebSite");

         backColor = _configuration.GetSection("Appearance").GetValue<string>("BackColor");
      }
   }
}
```

“Settings.cshtml”

```
@page
@model AppSettingsApp.Pages.SettingsModel
@{
   ViewData["Title"] = "Settings";
}
      <div>

         <h1 class="text-@Model.backColor">Settings</h1>

         Company Name: @Model.companyName
         <br />

         <a href="@Model.webSite" target="_blank">Company Web Site</a>
         <br />

      </div>
```

Run the Application



Now we can run the Application

appSetting.json:

"CompanyName": "University of South-Eastern Norway"
"WebSite": "https://www.usn.no/english/"
"BackColor": "warning"

Resources

- https://docs.microsoft.com/en-us/aspnet/core/fundamentals/configuration

Hans-Petter Halvorsen

University of South-Eastern Norway

www.usn.no

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

Web: https://www.halvorsen.blog