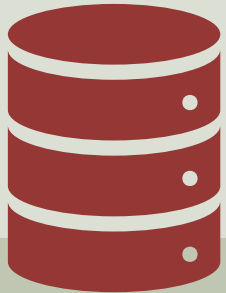


<https://www.halvorsen.blog>

ASP.NET Core

ASP.NET Core with Razor and SQL Server



Hans-Petter Halvorsen



Contents

- Introduction
 - ASP.NET Core and ASP.NET Core with Razor.
- SQL Server
 - Setting up the SQL Server Database.
- Visual Studio
 - Create an ASP.NET Core Razor App that retrieves data from the SQL Server Database.

<https://www.halvorsen.blog>

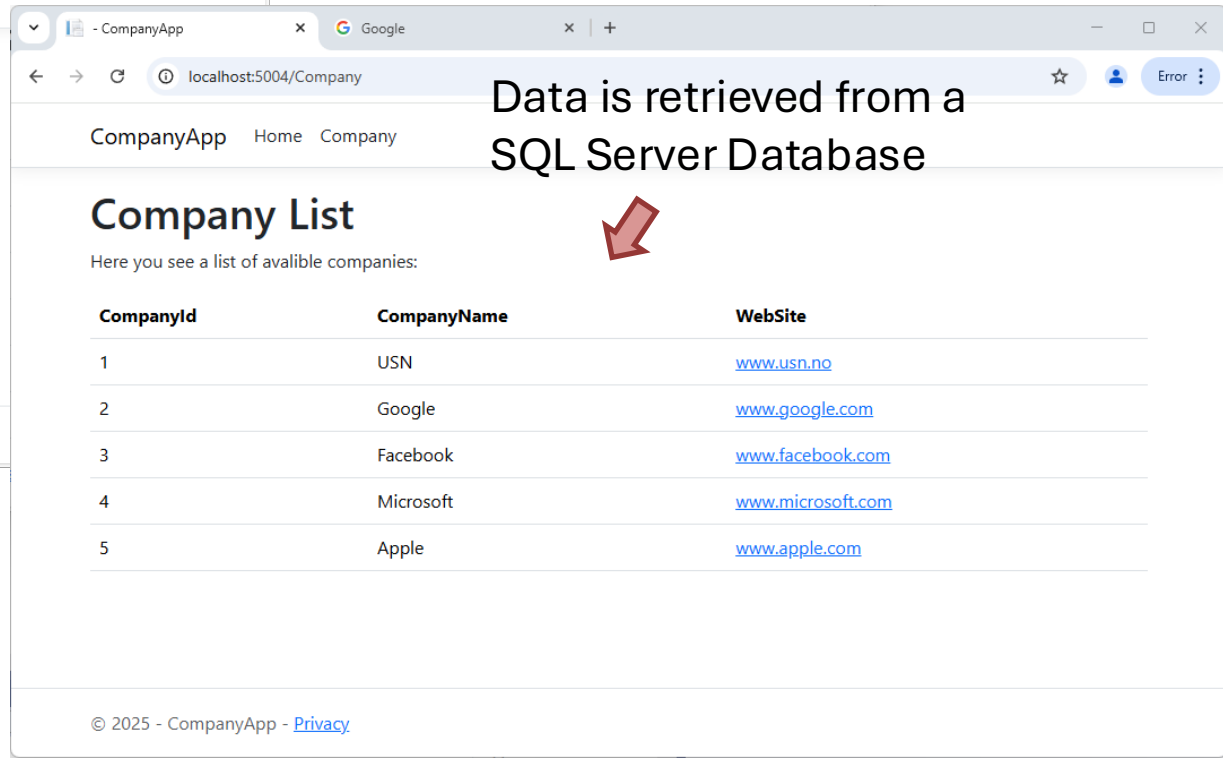
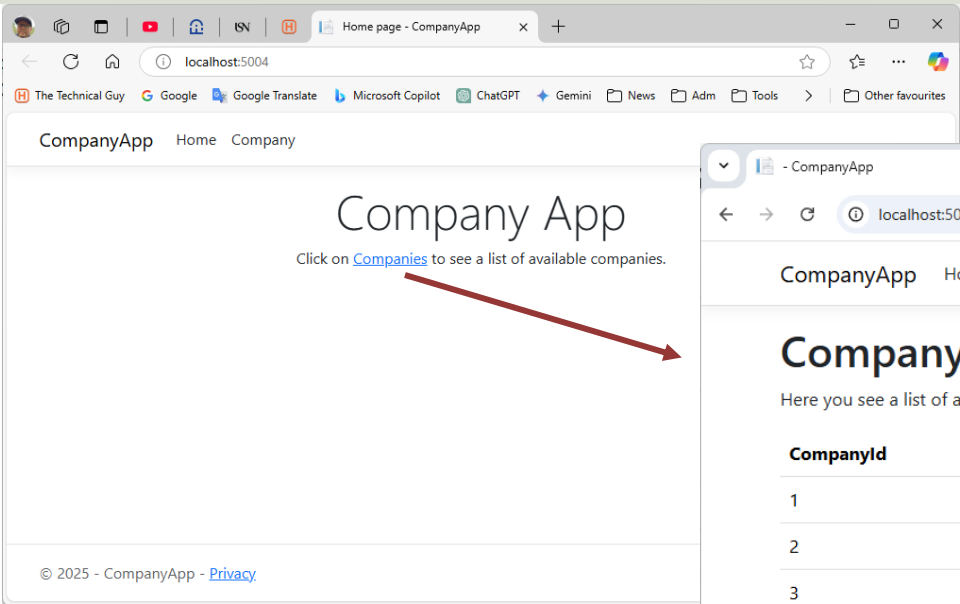
Introduction



[Table of Contents](#)

Hans-Petter Halvorsen

ASP.NET Core Razor Web App



ASP.NET Core

- ASP.NET Core is a framework for web development.
- ASP.NET Core is based on .NET and C#.
- What is the difference between ASP.NET Core and .NET?
 - ASP.NET Core is specifically designed for web development, while the .NET covers a broader range of application types, including Windows desktop, mobile, and web applications.
- In ASP.NET Core Razor code and layout are separated into 2 files; The layout file has the extension “. cshtml”, and the code-behind file has the extension “. cshtml.cs” (where “cs” is short for C#).
- The layout files “. cshtml” use something called **Razor** syntax and are mixed with HTML.
- ASP, ASP.NET and ASP.NET Core is made by Microsoft.
- Homepage: <https://dotnet.microsoft.com/en-us/apps/aspnet>

ASP.NET Core

ASP.NET Core is a framework for building Web Applications and Services with .NET and C#. ASP.NET Core supports different types, here are some examples:

- **ASP.NET Core with Razor Pages** The focus in this Tutorial!
- ASP.NET Core MVC
- ASP.NET Core Blazor Web Apps
- ASP.NET Core Web API with controllers
- ASP.NET Core Minimal Web APIs

<https://www.halvorsen.blog>

SQL Server

Setting up the SQL Server Database



[Table of Contents](#)

Hans-Petter Halvorsen

SQL Server

- SQL Server Express.
 - Free version of SQL Server that has all we need for the examples in this Tutorial.
- SQL Server Express consist of 2 parts (separate installation packages):
 - SQL Server Express.
 - SQL Server Management Studio (SSMS) – This software can be used to create Databases, create Tables, Insert/Retrieve or Modify Data, etc.
- Download:
<https://www.microsoft.com/sql-server/sql-server-downloads>
- SQL Server Express Installation YouTube:
<https://youtu.be/hhhggAlUYo8>

Database Table

Let's create a basic Table like this:

```
CREATE TABLE COMPANY
(
  CompanyId      int    PRIMARY KEY IDENTITY (1,1),
  CompanyName    varchar(50) NOT NULL,
  WebSite        varchar(100) NOT NULL,
)
GO
```

This is just a basic example, here you can add more columns like Address, Phone, etc.

Create Database and Table

We use **SQL Server Management Studio** to create the Database and the Table

Microsoft SQL Server Management Studio

Object Explorer

Connect - HANS-PETTER\SQLEXPRESS (SQL Server 16.0.1135 - sa)

Databases

- System Databases
- Database Snapshots
- BOOKS
- ORDERS
- STUDENTS
- StudentSystem
- Security
- Server Objects
- Replication
- Management
- XEvent Profiler

New Database

Select a page

- General
- Options
- Filegroups

Database name: WORK

Owner: <default>

Use full-text indexing

Database files:

Logical Name	File Type	Filegroup	Initial Size (MB)	Autogrow
WORK	ROWS...	PRIMARY	8	By 64 MB
WORK_log	LOG	Not Applicable	8	By 64 MB

Connection

Server: Hans-Petter\SQLEXPRESS

Connection: sa

[View connection properties](#)

Progress

Ready

Add

Microsoft SQL Server Management Studio

SQLQuery1.sql - HANS-PETTER\SQLEXPRESS.WORK (sa (64)) - Microsoft SQL Server Management Studio

Quick Launch (Ctrl+Q)

File Edit View Project Tools Window Help

SQLQuery1.sql - HANS-PETTER\SQLEXPRESS (SQL Server 16.0.1135 - sa)

Object Explorer

Connect - HANS-PETTER\SQLEXPRESS (SQL Server 16.0.1135 - sa)

Databases

- System Databases
- Database Snapshots
- BOOKS
- ORDERS
- STUDENTS
- StudentSystem
- WORK
- Database Diagrams
- Tables
- System Tables
- FileTables
- External Tables
- Graph Tables
- dbo.COMPANY
- Views
- External Resources
- Synonyms
- Programmability
- Query Store
- Service Broker
- Storage
- Security
- Security
- Server Objects
- Replication
- Management
- XEvent Profiler

```
CREATE TABLE COMPANY
(
    CompanyId int PRIMARY KEY IDENTITY (1,1),
    CompanyName varchar(50) NOT NULL,
    WebSite varchar(100) NOT NULL,
);
GO
```

150 %

Messages

Commands completed successfully.

Completion time: 2025-02-18T10:27:30.4856309+01:00

150 %

Query executed successfully.

HANS-PETTER\SQLEXPRESS (16... sa (64) WORK 00:00:00 0 rows

Create Data

We also use **SQL Server Management Studio** to insert some dummy data into the **COMPANY** table:

```
insert into COMPANY (CompanyName, WebSite) values ('USN', 'www.usn.no')  
GO
```

```
insert into COMPANY (CompanyName, WebSite) values ('Google', 'www.google.com')  
GO
```

```
insert into COMPANY (CompanyName, WebSite) values ('Facebook', 'www.facebook.com')  
GO
```

```
insert into COMPANY (CompanyName, WebSite) values ('Microsoft', 'www.microsoft.com')  
GO
```

```
insert into COMPANY (CompanyName, WebSite) values ('Apple', 'www.apple.com')  
GO
```

<https://www.halvorsen.blog>

Visual Studio

Create the ASP.NET Core Razor App



Hans-Petter Halvorsen

[Table of Contents](#)

ASP.NET Core Web App with Razor

Create a new project

Recent project templates

- Console App C#
- Setup Project C#
- Windows Forms App C#
- Windows Forms App Visual Basic
- MSTest Test Project C#
- Windows Forms App (.NET Framework) C#

ASP.NET Core x Clear all

All languages All platforms All project types

- ASP.NET Core Web App (Razor Pages)**
A project template for creating an ASP.NET Core application with example ASP.NET Core Razor Pages content
C# Linux macOS Windows Cloud Service Web
- ASP.NET Core Web API
A project template for creating a RESTful Web API using ASP.NET Core controllers or minimal APIs, with optional support for OpenAPI and authentication.
C# Linux macOS Windows API Cloud Service Web Web API
- ASP.NET Core Web API (native AOT)
A project template for creating a RESTful Web API using ASP.NET Core minimal APIs with native AOT.
C# Linux macOS Windows API Cloud Service Web Web API
- gRPC ASP.NET Core gRPC Service
A project template for creating a gRPC service using ASP.NET Core, with optional support for native AOT.
C# Linux macOS Windows Cloud Service Web
- ASP.NET Core Empty
An empty project template for creating an ASP.NET Core application. This template does not have any content in it.
C# Linux macOS Windows Cloud Service Web
- ASP.NET Core Web App (Model-View-Controller)
A project template for creating an ASP.NET Core application with example ASP.NET Core MVC Views and Controllers. This template can also be used for RESTful HTTP services.
C# Linux macOS Windows Cloud Service Web
- ASP.NET Core Empty
An empty project template for creating an ASP.NET Core application. This template does not have any content in it.
F# Linux macOS Windows Cloud Service Web
- ASP.NET Core Web App (Model-View-Controller)
A project template for creating an ASP.NET Core application with example ASP.NET Core MVC Views and Controllers. This template can also be used for RESTful HTTP services.
C# Linux macOS Windows Cloud Service Web

This is the recommended Template for ASP.NET Core Web App with Razor

ASP.NET Core has many different applications and has templates for different application types, services and purposes.

Visual Studio

Configure your new project

ASP.NET Core Web App (Razor Pages) C#

Project name

CompanyApp

Location

C:\Users\hansp\OneDrive\Courses\Webutvikling\Tutorials\ASP

Solution name ⓘ

CompanyApp

Place solution and project in the same directory

Project will be created in "C:\Users\hansp\OneDrive\Courses\Webutvikling\Tutorials\ASP.NET\ASP.NET Core with Razor\Development\CompanyApp\CompanyApp\"

Additional information

ASP.NET Core Web App (Razor Pages) C# Linux macOS Windows Cloud Service Web

Framework ⓘ

.NET 9.0 (Standard Term Support)

Authentication type ⓘ

None

Configure for HTTPS ⓘ

Enable container support ⓘ

Container OS ⓘ

Linux

Container build type ⓘ

Dockerfile

Do not use top-level statements ⓘ

Enlist in .NET Aspire orchestration ⓘ

Back

Create

NuGet – Database Communication

CompanyApp* NuGet: ...panyApp

Browse Installed Updates

Microsoft.Data Include pr...

Microsoft.Data.SqlClient by Microsoft, nugetsqltools, 775M downloads 6.0.1
The current data provider for SQL Server and Azure SQL databases. This has replaced System.Data.SqlClient. These classes provide access to SQL and encapsul...

Microsoft.Data.SqlClient.SNI.runtime by Microsoft, nugetsqltools, 576M 6.0.2
Internal implementation package not meant for direct consumption. Please do not reference directly.

Microsoft.Data.Sqlite.Core by aspnet, dotnetframework, EntityFramework, M 9.0.2
Microsoft.Data.Sqlite is a lightweight ADO.NET provider for SQLite. This package does not include a copy of the native SQLite library.

Microsoft.Data.OData by Microsoft, OData, 175M downloads 5.8.5
Classes to serialize, deserialize and validate OData JSON payloads.
This package version is deprecated.

Microsoft.Data.Edm by Microsoft, OData, 175M downloads 5.8.5
Classes to represent, construct, parse, serialize and validate entity data models. Targets .NET 4.0, Silverlight 4.0, or .NET Portable Lib with support for .NET 4.0, SL...

Microsoft.Data.Services.Client by Microsoft, OData, 115M downloads 5.8.5
LINQ-enabled client API for issuing OData queries and consuming OData payloads. Supports OData v3. Targets .NET 4.0, Silverlight 4.0 or .NET Portable Lib with supp...

Microsoft.Data.Sqlite by aspnet, dotnetframework, EntityFramework, Microsof 9.0.2
Microsoft.Data.Sqlite is a lightweight ADO.NET provider for SQLite.

Microsoft.Extensions.Configuration.Binder by aspnet, dotnetframew 9.0.2
Provides the functionality to bind an object to data in configuration providers for Microsoft.Extensions.Configuration. This package enables you to represent the conf...

Microsoft.EntitvFrameworkCore by aspnet, dotnetframework, EntityFrame 9.0.2

Each package is licensed to you by its owner. NuGet is not responsible for, nor does it grant any licenses to, third-party packages.

Don't show this again

NuGet Package Manager: CompanyApp

Package source: nuget.org

Microsoft.Data.SqlClient nuget.org

Installed: 6.0.1 Uninstall

Version: 6.0.1 Update

Package source mapping is off. Configure

Options

README Package Details

license MIT Nuget.org Downloads 775M

Azure Pipelines failed

Microsoft SqlClient Data Provider for SQL Server

Microsoft.Data.SqlClient is a .NET data provider for Microsoft SQL Server and the Azure SQL family of databases. It grew from a union of the two System.Data.SqlClient components which live independently in .NET Framework and .NET Core. Going forward, support for new SQL Server and Azure SQL features will only be implemented in Microsoft.Data.SqlClient.

Supportability

The Microsoft Data SqlClient package supports the

Solution Explorer

Search Solution Explorer (Ctrl+)

Solution 'CompanyApp' (1 of 1 project)

- CompanyApp
 - Connected Services
 - Dependencies
 - Analyzers
 - Frameworks
 - Packages
 - Microsoft.Data.SqlClient (6.0.1)
 - Properties
 - wwwroot
 - Models
 - Company.cs
 - Pages
 - appsettings.json
 - Program.cs

GitHub Copilot Chat Solution Explorer

Properties

CompanyApp General

- General
- Misc

UserSecretsId	
File Name	CompanyApp.csproj
Full Path	C:\Users\hansp\OneDr
Project Folder	C:\Users\hansp\OneDr

Error List

Entire Solution 0 Errors 0 Warnings 0 Messages Build + IntelliSe

Search Error List

Create Class

The screenshot shows the Visual Studio IDE with the 'Add New Item - CompanyApp' dialog box open. The dialog box is divided into several sections:

- Left Panel:** A tree view showing the project structure. Under 'Installed', 'C#' is expanded, and 'ASP.NET Core' is further expanded. The 'Code' folder is selected.
- Center Panel:** A list of item types to create. 'Class' is selected, showing a preview: 'Type: C#' and 'An empty class declaration'. Other options include 'Interface' and 'Code File'.
- Right Panel:** A search bar and a preview area for the selected item type.
- Bottom:** A 'Name:' field containing 'Company.cs', which is circled in red. Below it are 'Add' and 'Cancel' buttons.

In the background, the Solution Explorer on the right shows the project structure for 'CompanyApp', with the 'Models' folder highlighted by a red rectangle. The Properties window at the bottom right shows the 'Misc' properties for the 'Models' folder, including its name and full path.

Company Class

The screenshot displays the Visual Studio IDE with the following components:

- Code Editor:** Shows the `Company.cs` file with the following code:

```
1 using Microsoft.Data.SqlClient;
2
3 namespace CompanyApp.Models
4 {
5     9 references
6     public class Company
7     {
8         2 references
9         public int companyId { get; set; }
10        2 references
11        public string? companyName { get; set; }
12        2 references
13        public string? website { get; set; }
14
15        1 reference
16        public List<Company> GetCompanies()
17        {
18            string connectionString = "Data Source=HANS-PETTER\\SQLEXPRESS;Initial Catalog=WORK;Integrated Security=True;TrustServerCertificate=True";
19            SqlConnection con = new SqlConnection(connectionString);
20            con.Open();
21
22            string sqlQuery = "select CompanyId, CompanyName, WebSite from COMPANY";
23
24            SqlCommand cmd = new SqlCommand(sqlQuery, con);
25
26            SqlDataReader dr = cmd.ExecuteReader();
27
28            List<Company> companyList = new List<Company>();
29
30            while (dr.Read())
31            {
32                Company company = new Company();
33
34                company.companyId = Convert.ToInt32(dr["CompanyId"]);
35                company.companyName = dr["CompanyName"].ToString();
36                company.website = dr["WebSite"].ToString();
37
38                companyList.Add(company);
39            }
40            con.Close();
41            return companyList;
42        }
43    }
44 }
```
- Solution Explorer:** Shows the project structure for `CompanyApp`. The `Models` folder is expanded, and `Company.cs` is highlighted.
- Properties Window:** Shows the `Company.cs` File Properties. The **Advanced** tab is selected, showing the **Build Action** as `C# compiler` and the **Full Path** as `C:\Users\hansp\OneDrive\...`.

Company.cs

```
using Microsoft.Data.SqlClient;
```

```
namespace CompanyApp.Models
```

```
{  
    public class Company  
    {  
        public int companyId { get; set; }  
        public string? companyName { get; set; }  
        public string? webSite { get; set; }  
  
        public List<Company> GetCompanies()  
        {  
            string connectionString = "Data Source=SERVERNAME\\SQLEXPRESS;Initial Catalog=WORK;Integrated  
            Security=True;TrustServerCertificate=True";  
            SqlConnection con = new SqlConnection(connectionString);  
            con.Open();  
  
            string sqlQuery = "select CompanyId, CompanyName, WebSite from COMPANY";  
  
            SqlCommand cmd = new SqlCommand(sqlQuery, con);  
  
            SqlDataReader dr = cmd.ExecuteReader();  
  
            List<Company> companyList = new List<Company>();  
  
            while (dr.Read())  
            {  
                Company company = new Company();  
  
                company.companyId = Convert.ToInt32(dr["CompanyId"]);  
                company.companyName = dr["CompanyName"].ToString();  
                company.webSite = dr["WebSite"].ToString();  
  
                companyList.Add(company);  
            }  
            con.Close();  
            return companyList;  
        }  
    }  
}
```

New Razor Page – “Company.cshtml”

The image shows a screenshot of Visual Studio with the following elements:

- Solution Explorer:** Shows the project structure for 'CompanyApp'. The 'Pages' folder is expanded, and the 'Add' button is highlighted.
- Context Menu:** A right-click context menu is open over the 'Pages' folder. The 'Razor Page...' option is highlighted.
- Add New Item Dialog:** A dialog box titled 'Add New Item - CompanyApp' is open. It shows a list of item types under the 'C#' category. The 'Razor Page - Empty' item is selected and highlighted.
- Name Field:** The 'Name' field at the bottom of the dialog is filled with 'Company.cshtml'.
- Type Description:** On the right side of the dialog, the type is identified as 'C#' and described as 'A Razor page with a page model'.

Red boxes highlight the 'Razor Page...' menu item, the 'Add' button, the 'Razor Page - Empty' item in the list, and the 'Company.cshtml' name field.

“Company.cshtml.cs”

```
Company.cshtml | Company...html.cs | CompanyApp | CompanyApp.Page
1  using Microsoft.AspNetCore.Mvc.RazorPages;
2  using CompanyApp.Models;
3
4  namespace CompanyApp.Pages
5  {
6      4 references
7      public class CompanyModel : PageModel
8      {
9          public List<Company> companyList = new List<Company>();
10
11         0 references
12         public void OnGet()
13         {
14             Company company = new Company();
15             companyList = company.GetCompanies();
16         }
17     }
18 }
```

When a user request a Webpage, the code inside the **OnGet()** method are executed on the Server before the code is sent to the Client

public List<Company> companyList = new List<Company>();

0 references

public void OnGet()

{

Company company = new Company();

companyList = company.GetCompanies();

}

“Company.cshtml.cs”

```
using Microsoft.AspNetCore.Mvc.RazorPages;
using CompanyApp.Models;

namespace CompanyApp.Pages
{
    public class CompanyModel : PageModel
    {
        public List<Company> companyList = new List<Company>();

        public void OnGet()
        {
            Company company = new Company();
            companyList = company.GetCompanies();
        }
    }
}
```

Company.cshtml

```
1 @page
2 @model CompanyApp.Pages.CompanyModel
3 @{
4 }
5
6 <div>
7     <h1>Company List</h1>
8     <p>Here you see a list of available companies:</p>
9
10    <table class="table">
11        <thead>
12            <tr>
13                <th>CompanyId</th>
14                <th>CompanyName</th>
15                <th>WebSite</th>
16            </tr>
17        </thead>
18        <tbody>
19            @foreach (var company in Model.companyList)
20            {
21                <tr>
22                    <td> @company.companyId</td>
23                    <td> @company.companyName</td>
24                    <td> @company.webSite</td>
25                </tr>
26            }
27        </tbody>
28    </table>
29 </div>
```

```
@page
@model CompanyApp.Pages.CompanyModel
@{
}

<div>
    <h1>Company List</h1>
    <p>Here you see a list of available companies:</p>

    <table class="table">
        <thead>
            <tr>
                <th>CompanyId</th>
                <th>CompanyName</th>
                <th>WebSite</th>
            </tr>
        </thead>
        <tbody>
            @foreach (var company in Model.companyList)
            {
                <tr>
                    <td> @company.companyId</td>
                    <td> @company.companyName</td>
                    <td> @company.webSite</td>
                </tr>
            }
        </tbody>
    </table>
</div>
```

Add Web Page to Menu

The screenshot shows the Visual Studio Code editor with the following components:

- Toolbox:** Located on the left, it shows the 'HTML' category expanded. A red box highlights the 'Div' item, which is being dragged into the code editor.
- Code Editor:** The central pane shows the `_Layout.cshtml` file. The code defines a navigation bar with a collapse button and a list of navigation items. A red box highlights the new menu item being added: `<li class="nav-item">Company`.
- Solution Explorer:** On the right, it shows the project structure for 'CompanyApp'. The 'Pages' folder is expanded, showing the location of the new page to be added.
- Properties Window:** At the bottom right, it shows the 'Properties' window for the selected element in the code editor.

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="utf-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1.0" />
  <title>@ViewData["Title"] - CompanyApp</title>
  <script type="importmap"></script>
  <link rel="stylesheet" href="/lib/bootstrap/dist/css/bootstrap.min.css" />
  <link rel="stylesheet" href="/css/site.css" asp-append-version="true" />
  <link rel="stylesheet" href="/CompanyApp.styles.css" asp-append-version="true" />
</head>
<body>
  <header>
    <nav class="navbar navbar-expand-sm navbar-toggleable-sm navbar-light bg-white border-bottom box-shadow mb-3">
      <div class="container">
        <a class="navbar-brand asp-area="" asp-page="/Index">CompanyApp</a>
        <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target=".navbar-collapse" aria-cont
          aria-expanded="false" aria-label="Toggle navigation">
          <span class="navbar-toggler-icon"></span>
        </button>
        <div class="navbar-collapse collapse d-sm-inline-flex justify-content-between">
          <ul class="navbar-nav flex-grow-1">
            <li class="nav-item">
              <a class="nav-link text-dark asp-area="" asp-page="/Index">Home</a>
            </li>
            <li class="nav-item">
              <a class="nav-link text-dark asp-area="" asp-page="/Company">Company</a>
            </li>
          </ul>
        </div>
      </div>
    </nav>
  </header>
  <div class="container">
    <main role="main" class="pb-3">
      @RenderBody()
    </main>
  </div>
  <footer class="border-top footer text-muted">
    <div class="container">
      &copy; 2025 - CompanyApp - <a asp-area="" asp-page="/Privacy">Privacy</a>
    </div>
  </footer>
  <script src="/lib/jquery/dist/jquery.min.js"></script>
  <script src="/lib/bootstrap/dist/js/bootstrap.bundle.min.js"></script>
  <script src="/js/site.js" asp-append-version="true"></script>
  @await RenderSectionAsync("Scripts", required: false)
</body>
</html>
```

Testing the Web Application

Home page - CompanyApp

localhost:5004

CompanyApp Home **Company**

Welcome

Learn about [building Web apps with](#)

© 2025 - CompanyApp - [Privacy](#)

- CompanyApp

localhost:5004/Company

CompanyApp Home Company

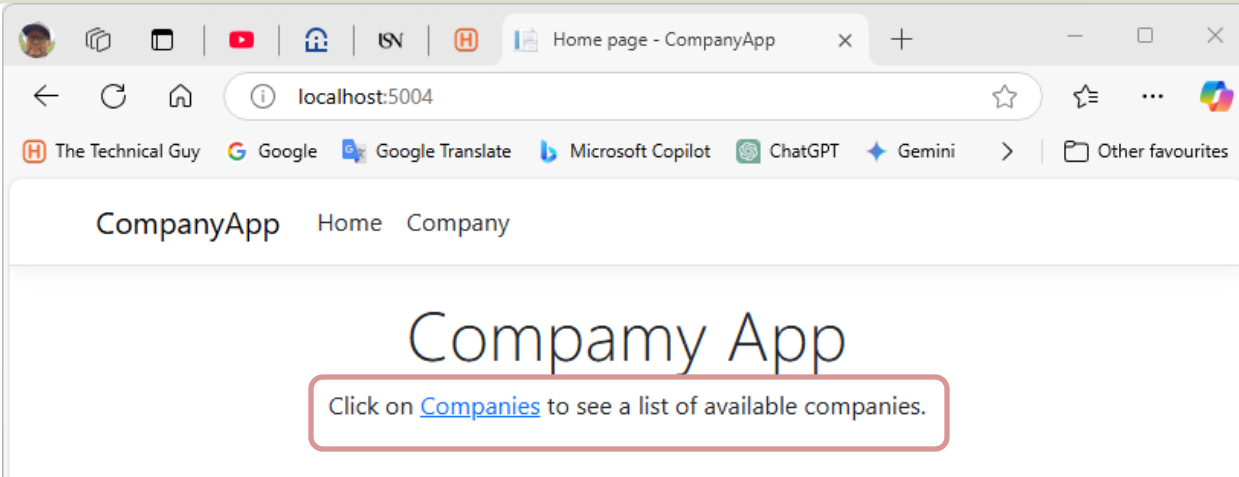
Company List

Here you see a list of available companies:

CompanyId	CompanyName	WebSite
1	USN	www.usn.no
2	Google	www.google.com
3	Facebook	www.facebook.com
4	Microsoft	www.microsoft.com
5	Apple	www.apple.com

© 2025 - CompanyApp - [Privacy](#)

Create Hyperlink to “Company.cshtml”

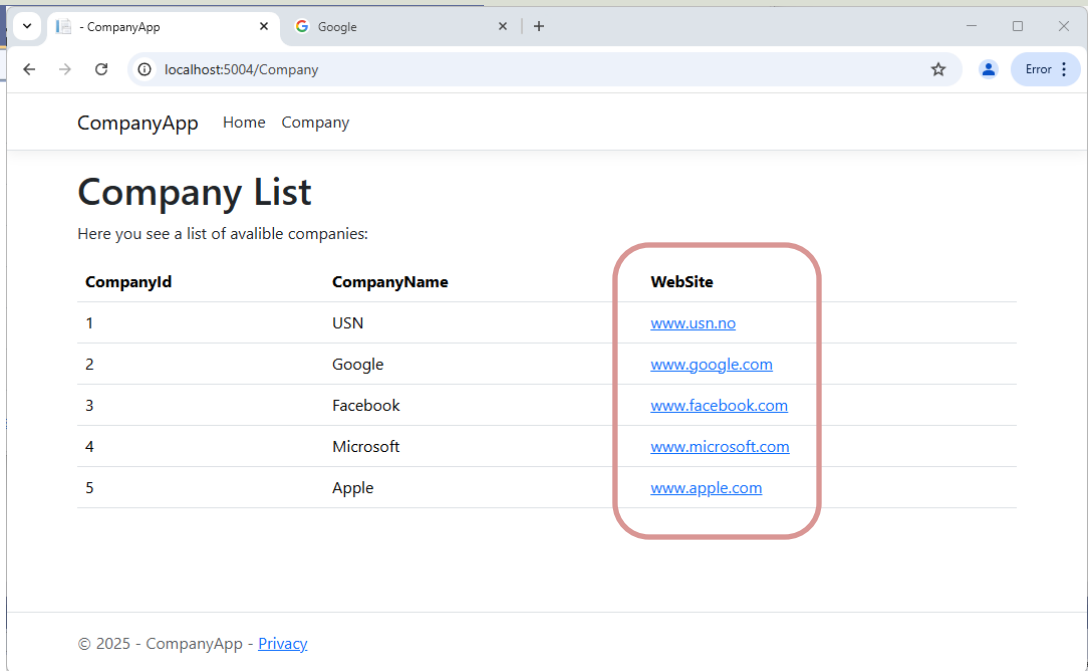


```
Index.cshtml
CompanyApp
1 @page
2 @model IndexModel
3 @{
4     ViewData["Title"] = "Home page";
5 }
6
7 <div class="text-center">
8     <h1 class="display-4">Company App</h1>
9     <p>Click on <a href="/Company">Companies</a> to see a list of available companies.</p>
10 </div>
11
```

Index.cshtml

Create WebSite Hyperlinks

```
Company.cshtml + x
CompanyApp
1 @page
2 @model CompanyApp.Pages.CompanyModel
3 @{
4 }
5
6 <div>
7 <h1>Company List</h1>
8 <p>Here you see a list of available companies:</p>
9
10 <table class="table">
11 <thead>
12 <tr>
13 <th>CompanyId</th>
14 <th>CompanyName</th>
15 <th>WebSite</th>
16 </tr>
17 </thead>
18 <tbody>
19 @foreach (var company in Model.companyList)
20 {
21 <tr>
22 <td>@company.companyId</td>
23 <td>@company.companyName</td>
24 <td><a href="https://@company.webSite" target="_blank">@company.webSite</a></td>
25 </tr>
26 }
27 </tbody>
28 </table>
29 </div>
```



Resources and References

- Tutorial: Get started with Razor Pages in ASP.NET Core: <https://learn.microsoft.com/en-us/aspnet/core/tutorials/razor-pages/razor-pages-start>
- Learn Razor Pages: <https://www.learnrazorpages.com>
- Introduction to ASP.NET Core Razor Pages: <https://www.csharp.com/article/introduction-to-asp-net-core-razor-pages/>

Hans-Petter Halvorsen

University of South-Eastern Norway

www.usn.no

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

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

