(T7)討論EfDbFirst的CRUD(Create,Read,Update,Delete)  
CourseGUID: 8503b39c-5887-4634-8291-facfb3117924  
=======================================================================  
(T7)討論EfDbFirst的CRUD(Create,Read,Update,Delete)  
=======================================================================  
0. Summary

0.1. Summary

0.2. MVC conventions

-----------

1. OnlineGame DB

1.1. TSQL

1.2. Security login

-----------

2. New Project - OnlineGame

2.1. New Project - OnlineGame.Web

2.1.1. Global.asax.cs

2.1.2. App\_Start/RouteConfig.cs

2.2. ADO.Net Entity Data Model - Entity Framework

2.3. Controllers/GamersController.cs

-----------

3. OnlineGame.Web

3.1. Models/Teams/Team.cs

3.2. Models/Teams/TeamMetaData.cs

3.3. Views/Gamers/Create.cshtml

3.4. Models/Gamers/Gamer.cs

3.5. Models/Gamers/GamerMetaData.cs

3.6. Views/Gamers/Edit.cshtml

3.7. Models/Teams/TeamTotals.cs

3.8. Controllers/GamersController.cs

3.9. Views/Gamers/GamersByTeam.cshtml  
=======================================================================

0. Summary

0.1. Summary

In this tutorial, we will discuss

\* MvcConventions

\* AdoDotNetEntityDataModel

\* EntityFramework

\* AutoGenerate Delete, Update, Insert, Read

Tutorial 7: 資料庫連結3 - 自動生成程式碼秒殺Entity Framework以及新增/更新/移除資料

本課程一開始先帶你手把手完全手寫Entity Framework和ADO.NET。等你熟悉後，教你如何使用自動生成的程式碼秒殺Entity Framework新增/更新/移除。因此本MVC的課程提供學生多種連接資料庫的選擇。之後的教學還討論當你的資料庫結構改變後，要如何update現有的Entity Framework程式碼!?如此完整的課程值得你的投資。

0.2. MVC conventions

In MVC conventions,

1. Controllers must have the word "Controller" as the suffix and must extend "IController" interface.

2. A view must remain under "Views" folder.

3. If the view is for GamerController, then the view must remain under "Views/Gamer" folder.

4. In the "HomeController", when "Index" action "return View()", it will search the following files in order.

    4.1. ~/Views/Home/Index.aspx

    4.2. ~/Views/Home/Index.ascx

    4.3. ~/Views/Shared/Index.aspx

    4.4. ~/Views/Shared/Index.ascx

    4.5. ~/Views/Home/Index.cshtml

    4.6. ~/Views/Home/Index.vbhtml

    4.7. ~/Views/Shared/Index.cshtml

    4.8. ~/Views/Shared/Index.vbhtml

5. By MVC convention, MVC will look for the view in the following locations

    5.1. Views/ControllerName

    5.2. Views/Shared

6. The extension name of view can be cshtml, vbhtml, aspx, or ascx.

7. Models can be anywhere, even can be in another project.  However, it is better to put it in "Models" folder.

8. You may put Models in another project as business layer.

9. Shared folder stores shared views.

    E.g. Master for aspx and Layout pages for cshtml

1. OnlineGame DB

1.1. TSQL

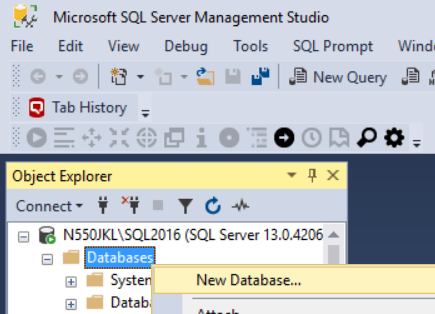
In SQL server Management Studio (SSMS)

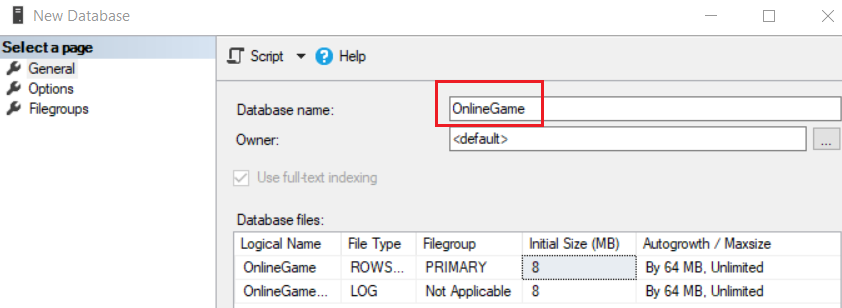
Database --> Right Click --> New Database -->

In General Tab -->

Name: **OnlineGame**

In options Tab --> Recovery model : **Simple**





Graphical user interface, text, application

Description automatically generated

--1. Drop if it exists

--Drop Table if it exists.

IF ( EXISTS ( SELECT    \*

              FROM      INFORMATION\_SCHEMA.TABLES

              WHERE     TABLE\_NAME = 'Gamer' ) )

    BEGIN

        TRUNCATE TABLE Gamer;

        DROP TABLE Gamer;

    END;

GO -- Run the previous command and begins new batch

IF ( EXISTS ( SELECT    \*

              FROM      INFORMATION\_SCHEMA.TABLES

              WHERE     TABLE\_NAME = 'Team' ) )

    BEGIN

        TRUNCATE TABLE Team;

        DROP TABLE Team;

    END;

GO -- Run the previous command and begins new batch

--Drop Stored Procedure if it exists.

--IF OBJECT\_ID('spSearchGamer') IS NOT NULL

IF ( EXISTS ( SELECT    \*

              FROM      INFORMATION\_SCHEMA.ROUTINES

              WHERE     ROUTINE\_TYPE = 'PROCEDURE'

                        AND LEFT(ROUTINE\_NAME, 3) NOT IN ( 'sp\_', 'xp\_', 'ms\_' )

                        AND SPECIFIC\_NAME = 'spGetGamers' ) )

    BEGIN

        DROP PROCEDURE spGetGamers;

    END;

GO -- Run the previous command and begins new batch

IF ( EXISTS ( SELECT    \*

              FROM      INFORMATION\_SCHEMA.ROUTINES

              WHERE     ROUTINE\_TYPE = 'PROCEDURE'

                        AND LEFT(ROUTINE\_NAME, 3) NOT IN ( 'sp\_', 'xp\_', 'ms\_' )

                        AND SPECIFIC\_NAME = 'spAddGamer' ) )

    BEGIN

        DROP PROCEDURE spAddGamer;

    END;

GO -- Run the previous command and begins new batch

IF ( EXISTS ( SELECT    \*

              FROM      INFORMATION\_SCHEMA.ROUTINES

              WHERE     ROUTINE\_TYPE = 'PROCEDURE'

                        AND LEFT(ROUTINE\_NAME, 3) NOT IN ( 'sp\_', 'xp\_', 'ms\_' )

                        AND SPECIFIC\_NAME = 'spSaveGamer' ) )

    BEGIN

        DROP PROCEDURE spSaveGamer;

    END;

GO -- Run the previous command and begins new batch

IF ( EXISTS ( SELECT    \*

              FROM      INFORMATION\_SCHEMA.ROUTINES

              WHERE     ROUTINE\_TYPE = 'PROCEDURE'

                        AND LEFT(ROUTINE\_NAME, 3) NOT IN ( 'sp\_', 'xp\_', 'ms\_' )

                        AND SPECIFIC\_NAME = 'spDeleteGamer' ) )

    BEGIN

        DROP PROCEDURE spDeleteGamer;

    END;

GO -- Run the previous command and begins new batch

--2. Create Table

CREATE TABLE Team

    (

      Id INT PRIMARY KEY

             IDENTITY(1, 1)

             NOT NULL ,

      [Name] NVARCHAR(100) NULL

    );

GO -- Run the previous command and begins new batch

CREATE TABLE Gamer

    (

      Id INT PRIMARY KEY

             IDENTITY(1, 1)

             NOT NULL ,

      [Name] NVARCHAR(100) NULL ,

      Gender NVARCHAR(10) NULL ,

      City NVARCHAR(50) NULL ,

      DateOfBirth DATETIME NULL ,

      TeamId INT FOREIGN KEY REFERENCES Team ( Id )

    );

GO -- Run the previous command and begins new batch

--3. Insert Data

INSERT  Team

VALUES  ( N'Team1' );

INSERT  Team

VALUES  ( N'Team2' );

INSERT  Team

VALUES  ( N'Team3' );

INSERT  Gamer

VALUES  ( N'Name01 ABB', N'Male', N'City01', '1979/4/28', 1 );

INSERT  Gamer

VALUES  ( N'Name02 CDDE', N'Female', N'City03', '1981/7/24', 2 );

INSERT  Gamer

VALUES  ( N'Name03 FIJK', N'Female', N'City01', '1984/12/5', 3 );

INSERT  Gamer

VALUES  ( N'Name04 LMOPPQ', N'Male', N'City02', '1983/5/29', 1 );

INSERT  Gamer

VALUES  ( N'Name05 QRSTT', N'Male', N'City01', '1979/6/20', 3 );

INSERT  Gamer

VALUES  ( N'Name06 TUVVX', N'Female', N'City03', '1984/5/15', 3 );

INSERT  Gamer

VALUES  ( N'Name07 XYZZXX', N'Female', N'City01', '1986/4/29', 2 );

INSERT  Gamer

VALUES  ( N'Name08 ABBCDE', N'Male', N'City02', '1985/7/28', 1 );

INSERT  Gamer

VALUES  ( N'Name09 QRSTTUVXX', N'Male', N'City02', '1983/4/16', 1 );

GO -- Run the previous command and begins new batch

--4. SP

CREATE PROCEDURE spGetGamers

AS

    BEGIN

        SELECT  \*

        FROM    Gamer;

    END;

GO -- Run the previous command and begins new batch

CREATE PROCEDURE spAddGamer

    (

      @Name NVARCHAR(50) ,

      @Gender NVARCHAR(10) ,

      @City NVARCHAR(50) ,

      @DateOfBirth DateTime ,

      @TeamId INT

    )

AS

    BEGIN

        INSERT  INTO Gamer

        VALUES  ( @Name, @Gender, @City, @DateOfBirth, @TeamId );

    END;

GO -- Run the previous command and begins new batch

CREATE PROCEDURE spSaveGamer

    (

      @Id INT ,

      @Name NVARCHAR(50) ,

      @Gender NVARCHAR(10) ,

      @City NVARCHAR(50) ,

      @DateOfBirth DateTime ,

      @TeamId INT

    )

AS

    BEGIN

        UPDATE  dbo.Gamer

        SET     Name = @Name ,

                Gender = @Gender ,

                City = @City ,

                DateOfBirth = @DateOfBirth ,

                TeamId = @TeamId

        WHERE   Id = @Id;

    END;

GO -- Run the previous command and begins new batch

CREATE PROCEDURE spDeleteGamer ( @Id int )

AS

    BEGIN

        DELETE  FROM Gamer

        WHERE   Id = @Id;

    END;

GO -- Run the previous command and begins new batch

--EXEC spGetGamers

--GO -- Run the previous command and begins new batch

1.2. Security login

In SQL server

Object Explorer --> Security --> Logins --> New Logins

-->

General Tab

Login Name :

**Tester**

Password:

**1234**

Default Database:

**OnlineGame**

-->

Server Roles Tab

Select

**sysadmin**

-->

User Mapping Tab

Select **OnlineGame**

Select every single role.



Graphical user interface, application

Description automatically generated

Graphical user interface, text, application

Description automatically generated



2. New Project - OnlineGame

File --> New --> Project... -->

Other Project Types --> Visual Studio Solutions -->  Blank Solution

-->

Name: **OnlineGame**

Graphical user interface, application

Description automatically generated

2.1. New Project - OnlineGame.Web

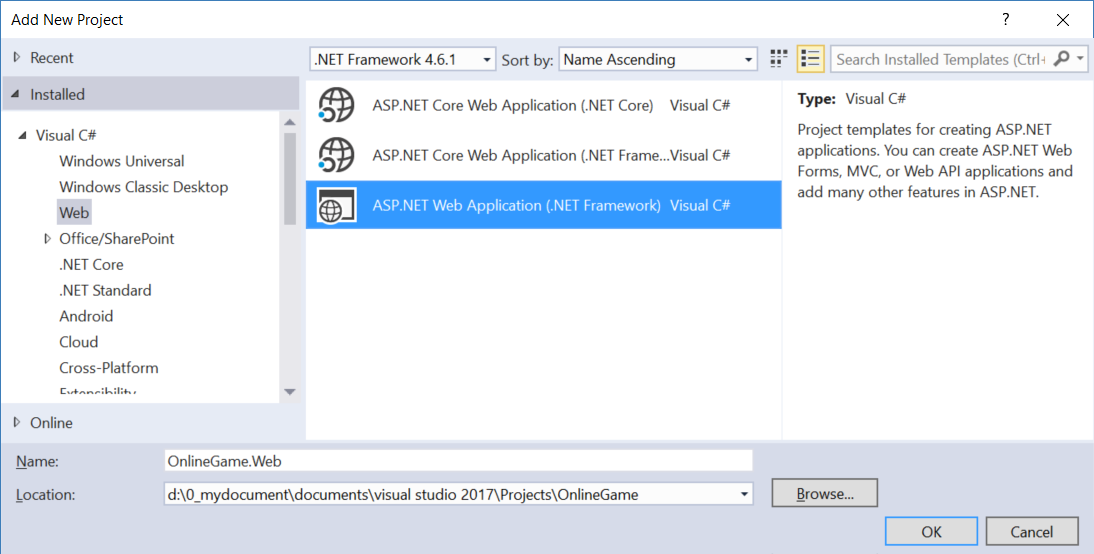
Solutions Name --> Add --> New Project -->

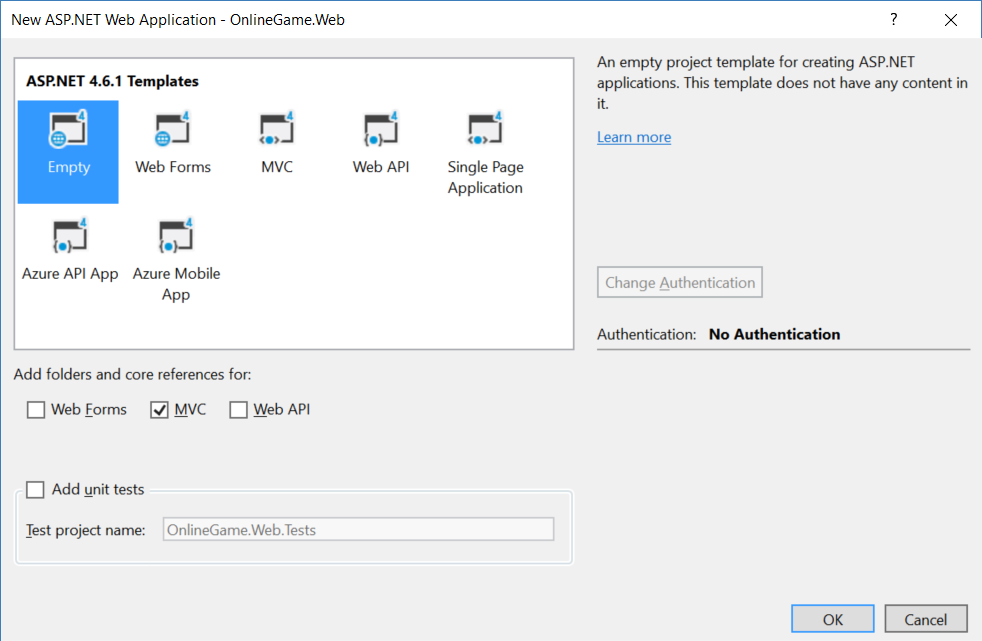
Visual C# --> Web --> ASP.NET Web Application (.Net Framework)

-->

Name: **OnlineGame.Web**

Empty --> Select "MVC" --> OK





2.1.1. Global.asax.cs

using System.Web.Mvc;

using System.Web.Routing;

namespace OnlineGame.Web

{

    public class MvcApplication : System.Web.HttpApplication

    {

        //Application\_Start() is the magic start point of this application

        protected void Application\_Start()

        {

            AreaRegistration.RegisterAllAreas();

            //1.

            //Register Route Configure in RouteConfig.cs

            //If you want to see route configuration,

            //you may find it in RouteConfig.cs

            //2.

            //System.Web.Routing.RouteCollection Routes { get; }

            //Gets a collection of objects that derive from the System.Web.Routing.RouteBase class.

            RouteConfig.RegisterRoutes(RouteTable.Routes);

        }

    }

}

2.1.2. App\_Start/RouteConfig.cs

using System.Web.Mvc;

using System.Web.Routing;

namespace OnlineGame.Web

{

    public class RouteConfig

    {

        public static void RegisterRoutes(RouteCollection routes)

        {

            //Handle the Route of the axd request file.

            //E.g. [ASP.Net](http://asp.net/) Tracing

            routes.IgnoreRoute("{resource}.axd/{\*pathInfo}");

            //Handle the Route called "Default".

            //The mapping URL is "{controller}/{action}/{id}"

            //Set the default value of Controller, action, and id.

            routes.MapRoute(

                name: "Default",

                url: "{controller}/{action}/{id}",

                defaults: new { controller = "Gamers", action = "Index", id = UrlParameter.Optional }

            );

        }

    }

}

/\*

1.

//routes.MapRoute(

//    name: "Default",

//    url: "{controller}/{action}/{id}",

//    defaults: new { controller = "Home", action = "Index", id = UrlParameter.Optional }

//);

1.1.

When a request comes in,

it's trying to do a pattern match based on

all the templates it sees in these mapped routes.

A route is some instructions for

how to take a URI coming into a request

and map it to some code,

normally a controller.

In this case,

look at defaults parameter,

when user request <http://localhost:PortNumber/>

IIS Express will run

HomeController Index action.

It will map to Controllers/HomeController.cs

and   map to Index Method

1.2.

By convention in MVC.

All controllers will have Controller suffix.

This suffix is not required in the URL.

So, if you want to invoke Home controller,

you specify /Home and not /HomeController.

-----------------------------------

2.

//routes.IgnoreRoute("{resource}.axd/{\*pathInfo}");

2.1.

Reference:

<https://stackoverflow.com/questions/9016650/what-is-routes-ignorerouteresource-axd-pathinfo>

This line can handle the axd file request route,

E.g. trace.axd

.axd files don't exist physically.

[ASP.NET](http://asp.net/) uses URLs with .axd extensions

(ScriptResource.axd and WebResource.axd) internally,

and they are handled by an HttpHandler.

Therefore, you should keep this rule,

to prevent [ASP.NET](http://asp.net/) MVC from trying to handle the request

instead of letting the dedicated HttpHandler do it.

2.2.

trace.axd

Reference:

<https://msdn.microsoft.com/en-us/library/wwh16c6c.aspx>

trace.axd trace details for a specific request.

If you want to enable trace.axd,

then you have to go to Web.config

Add <trace enabled="true" pageOutput="false"/> under <system.web>

Then run the project, type the following URL

<http://localhost/OnlineGame.Web/trace.axd>

This will return [ASP.NET](http://asp.net/) trace, trace.axd.

If you do not have

// routes.IgnoreRoute("{resource}.axd/{\*pathInfo}");

then you can not enable the trace.axd.

\*/

2.2. ADO.Net Entity Data Model - Entity Framework

In Visual Studio 2017

**Models** folder --> Right Click --> Add --> New Item

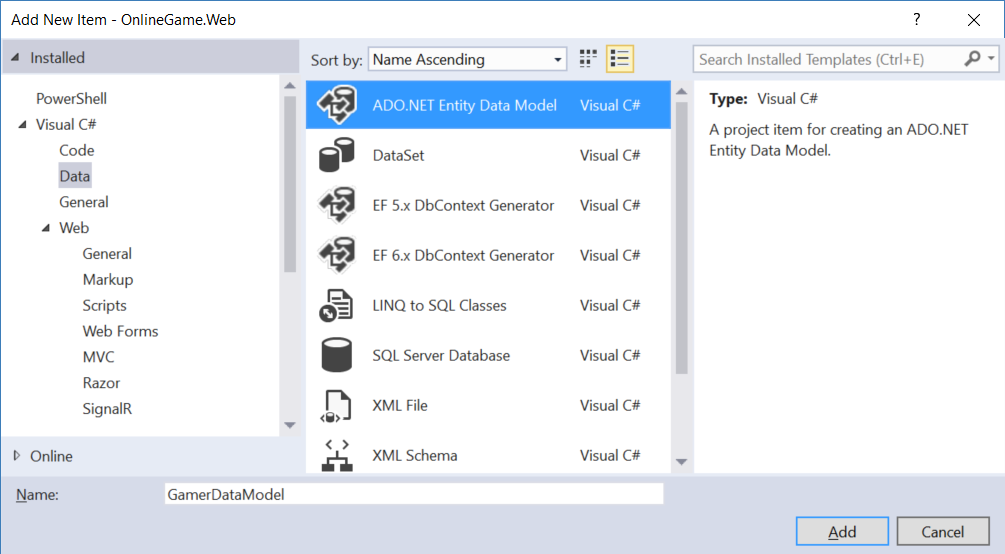
--> Visual C# --> Data  -->  ADO.Net Entity Data Model

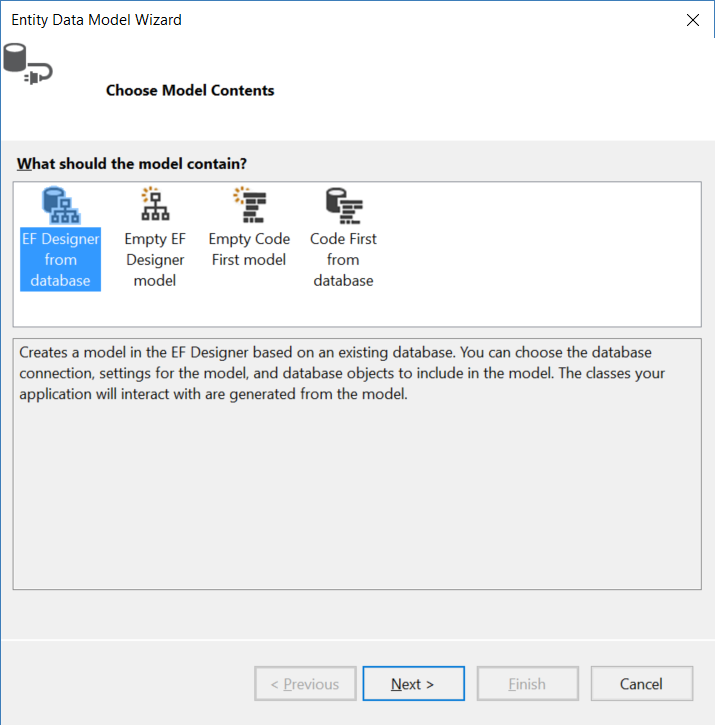
Name:

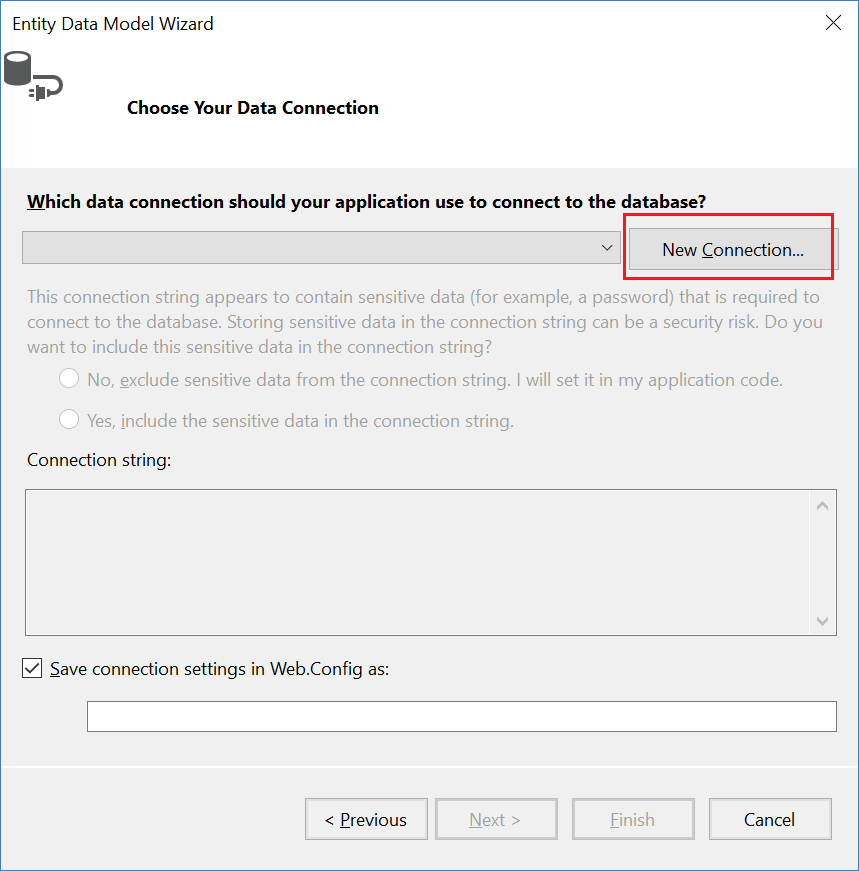
**GamerDataModel**

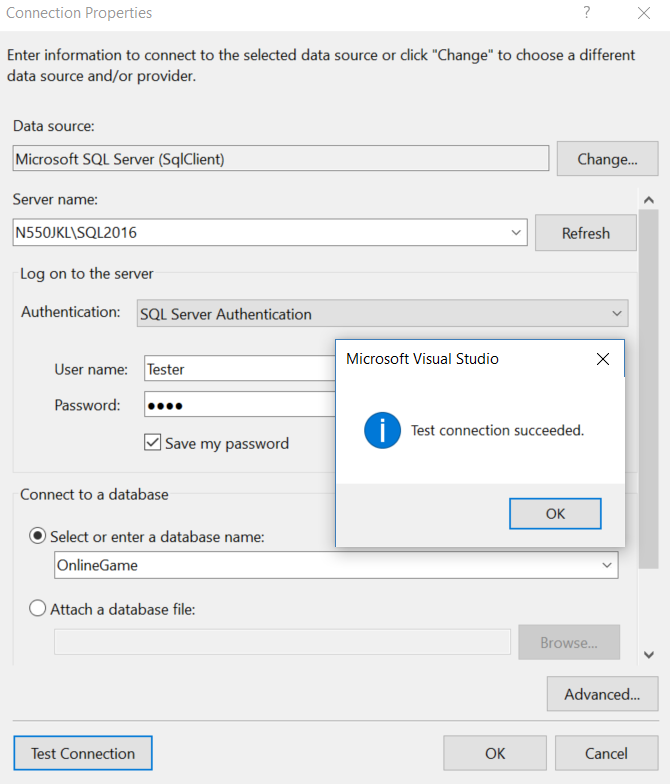
-->

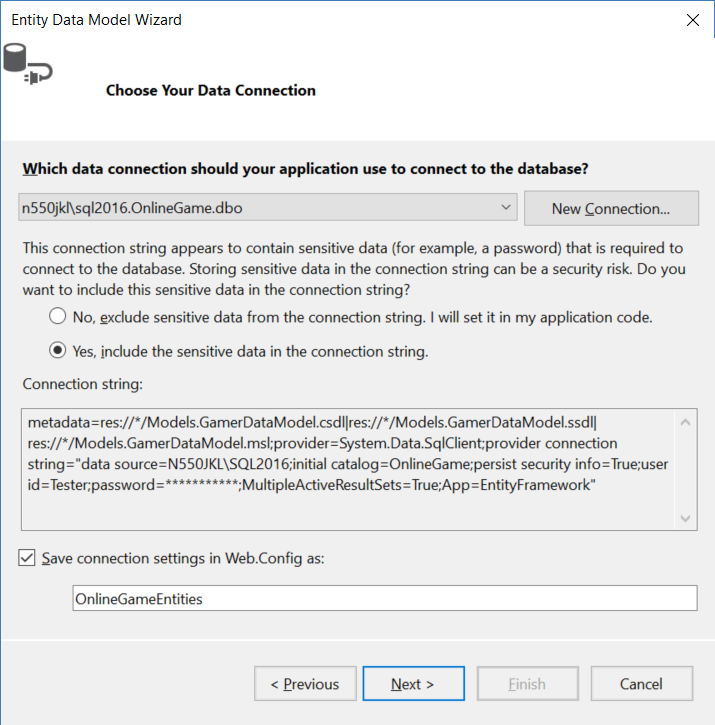
EF Designer from database

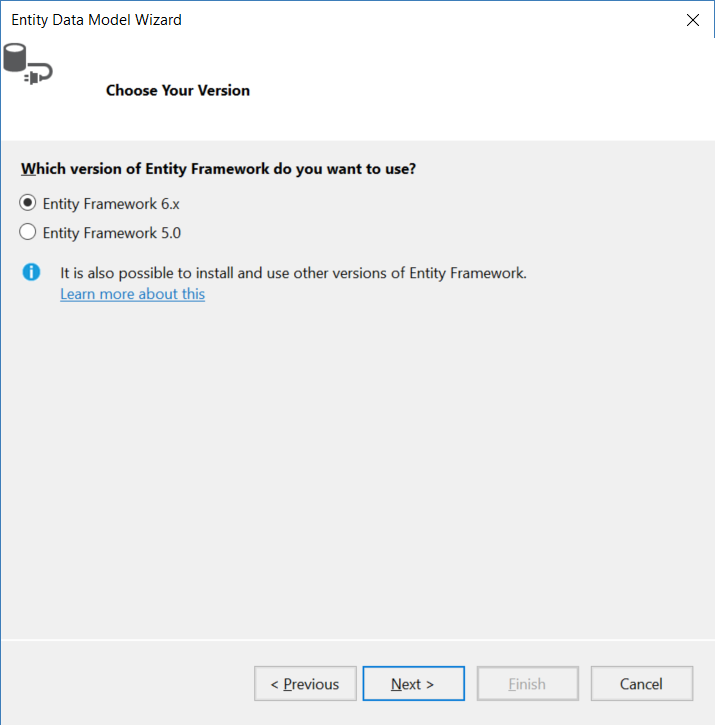


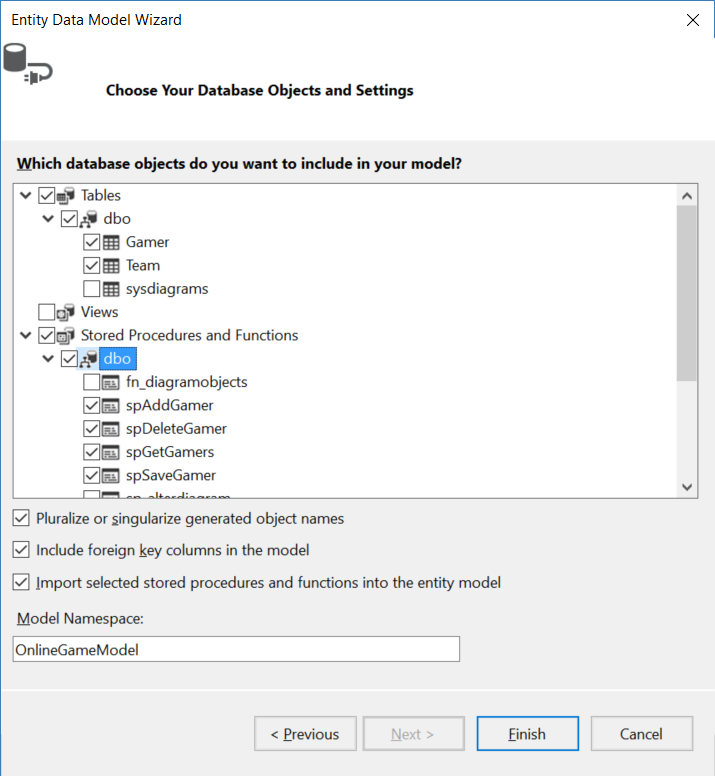


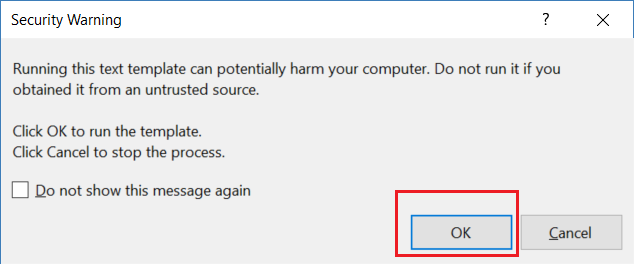


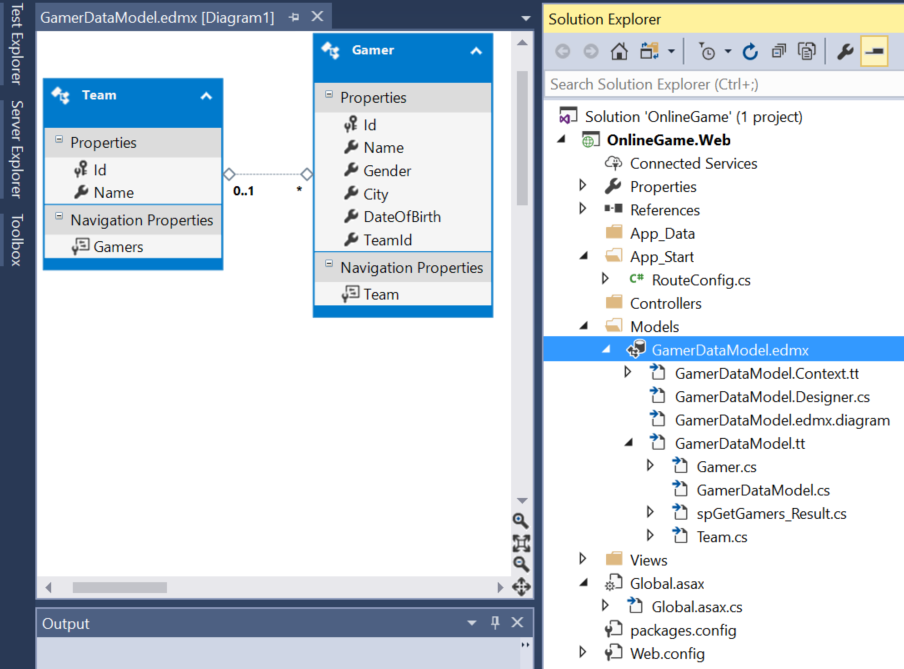












2.3. Controllers/GamersController.cs

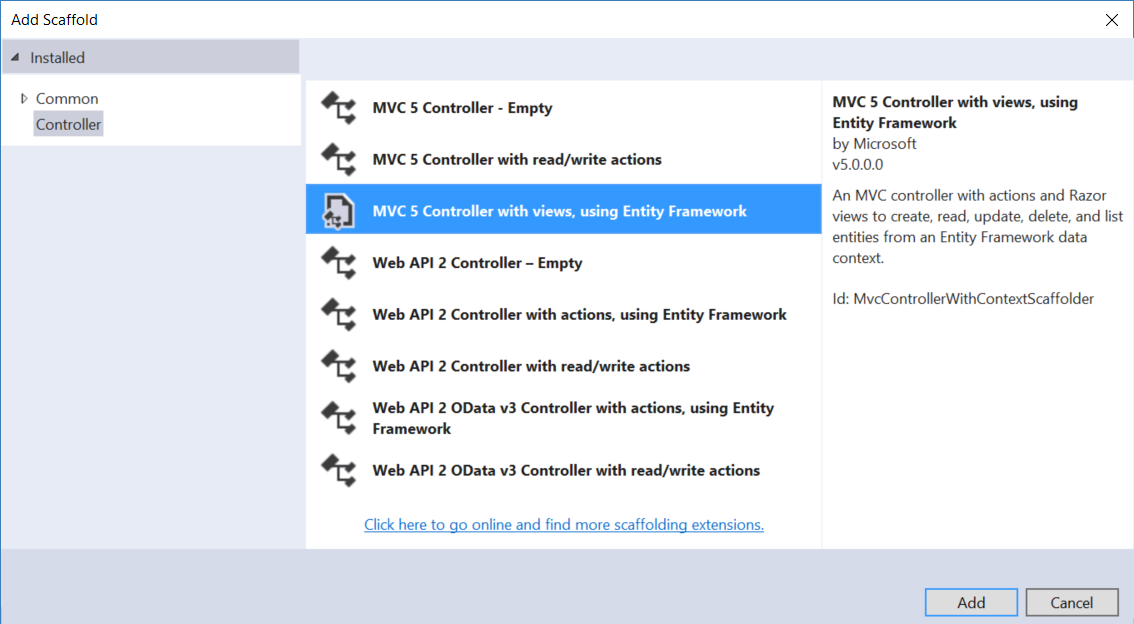
Controllers --> Right click --> Add --> Controller

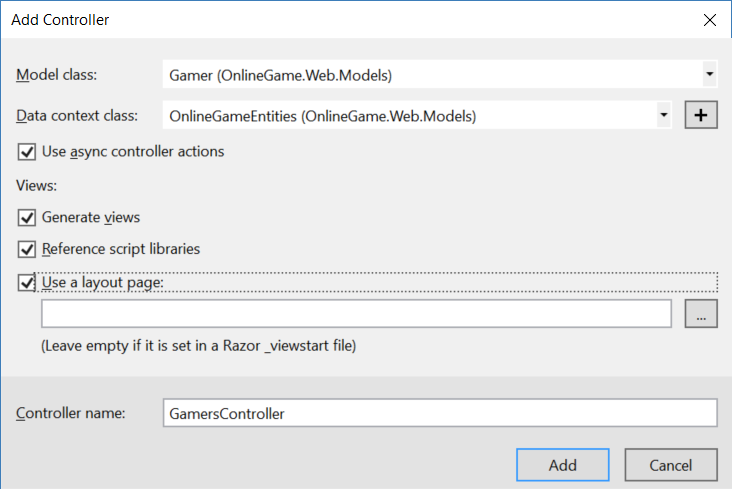
-->

**MVC 5 Controller with views, using Entity Framework**

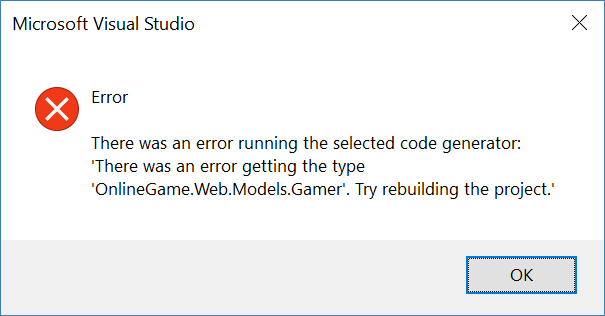
Graphical user interface, application

Description automatically generated

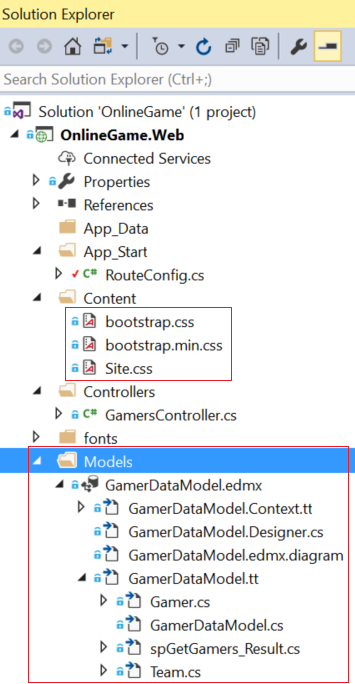


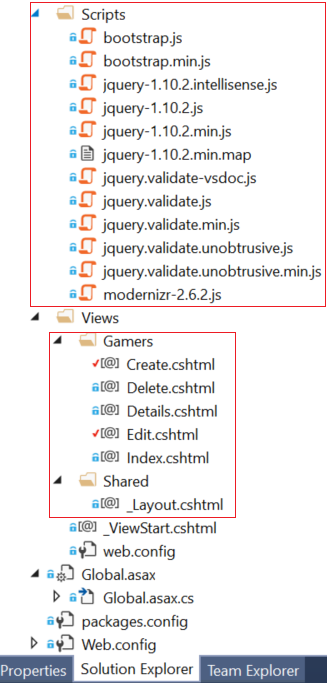


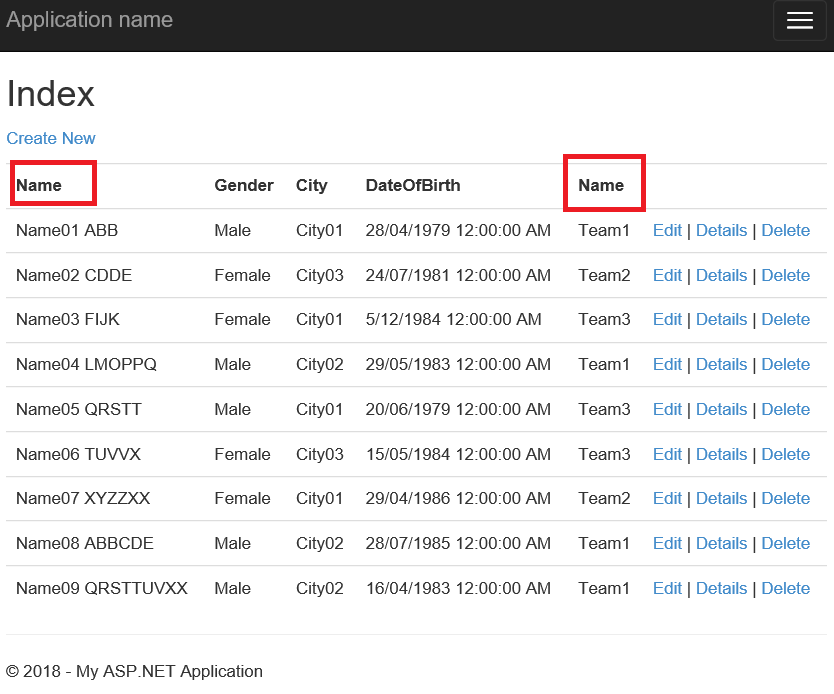
If you see the following error message, then you have to re-build solution before you create the controller.



It will automatically generate the controller, views, and several javascript and css files.







3. OnlineGame.Web

3.1. Models/Teams/Team.cs

using System.ComponentModel.DataAnnotations;

namespace OnlineGame.Web.Models

{

    [MetadataType(typeof(TeamMetaData))]

    public partial class Team

    {

        //[Display(Name = "Team Name")]

        //public string Name { get; set; }

        //// Error!!

        //// Memeber with the same name is areadly declared in other auto-generated partial class.

        //// Thus, you need MetadataType to add extra code for the Property.

        //// E.g. tadataType(typeof(TeamMetaData))]

        //// In this case, you may add some extra code for the Property in MetadataType class

    }

}

3.2. Models/Teams/TeamMetaData.cs

using System.ComponentModel.DataAnnotations;

namespace OnlineGame.Web.Models

{

    public class TeamMetaData

    {

        [Display(Name = "Team Name")]

        public string Name { get; set; }

        // Here is the place you may add some extra code for the property

        // which is already in the auto-generate partail class.

    }

}

Table

Description automatically generated

3.3. Views/Gamers/Create.cshtml

@model OnlineGame.Web.Models.Gamer

@{

    ViewBag.Title = "Create";

}

<h2>Create</h2>

@\*@using (Html.BeginForm("Create", "Gamer"))\*@

@using (Html.BeginForm())

{

    @Html.AntiForgeryToken()

    <div class="form-horizontal">

        <h4>Gamer</h4>

        <hr />

        @Html.ValidationSummary(true, "", new { @class = "text-danger" })

        <div class="form-group">

            @Html.LabelFor(model => model.Name, new { @class = "control-label col-md-2" })

            <div class="col-md-10">

                @Html.EditorFor(model => model.Name, new { htmlAttributes = new { @class = "form-control" } })

                @Html.ValidationMessageFor(model => model.Name, "", new { @class = "text-danger" })

            </div>

        </div>

        <div class="form-group">

            @Html.LabelFor(model => model.Gender, new { @class = "control-label col-md-2" })

            <div class="col-md-10">

                @\*@Html.EditorFor(model => model.Gender, new { htmlAttributes = new { @class = "form-control" } })\*@

                @\*@Html.DropDownList("Gender",new List<SelectListItem>

                    {

                        new SelectListItem{Text = "Male", Value = "Male"},

                        new SelectListItem{Text = "Female", Value = "Female"}

                    })\*@

                @Html.DropDownList("Gender", new List<SelectListItem>

                    {

                        new SelectListItem{Text = "Male", Value = "Male"},

                        new SelectListItem{Text = "Female", Value = "Female"}

                    }, "Select Gender")

                @Html.ValidationMessageFor(model => model.Gender, "", new { @class = "text-danger" })

            </div>

        </div>

        <div class="form-group">

            @Html.LabelFor(model => model.City, new { @class = "control-label col-md-2" })

            <div class="col-md-10">

                @Html.EditorFor(model => model.City, new { htmlAttributes = new { @class = "form-control" } })

                @Html.ValidationMessageFor(model => model.City, "", new { @class = "text-danger" })

            </div>

        </div>

        <div class="form-group">

            @Html.LabelFor(model => model.DateOfBirth, new { @class = "control-label col-md-2" })

            <div class="col-md-10">

                @Html.EditorFor(model => model.DateOfBirth, new { htmlAttributes = new { @class = "form-control" } })

                @Html.ValidationMessageFor(model => model.DateOfBirth, "", new { @class = "text-danger" })

            </div>

        </div>

        <div class="form-group">

            @Html.LabelFor(model => model.TeamId, "TeamId", new { @class = "control-label col-md-2" })

            <div class="col-md-10">

                @Html.DropDownList("TeamId", null, "Select Team", new { @class = "form-control" })

                @Html.ValidationMessageFor(model => model.TeamId, "", new { @class = "text-danger" })

            </div>

        </div>

        <div class="form-group">

            <div class="col-md-offset-2 col-md-10">

                <input type="submit" value="Create" class="btn btn-default" />

            </div>

        </div>

    </div>

}

<div>

    @Html.ActionLink("Back to List", "Index")

</div>

<script src="~/Scripts/jquery-1.10.2.min.js"></script>

<script src="~/Scripts/jquery.validate.min.js"></script>

<script src="~/Scripts/jquery.validate.unobtrusive.min.js"></script>

@\*

1.

1.1.

Select <form>....</form>

We are using Web Essentials extension

Select part of HTML text, then press Shift + Alt + W

it will surround those pieces of text with <div>

1.2.

// <div class="row">

class="row" in bootstrap means this is the container for the bootstrap 12 columns grid system

1.3.

// <div class="col-md-6">

1.3.1.

.col-lg-XX means the screen size is >= 1200px, lg means large

.col-md-XX means the screen size is >= 992px, means 992px to 1199px, md means medium

.col-sm-XX means the screen size is >= 768px, means 768px to 991px, sm means small

.col-xs-XX means the screen size is < 768px, xs means extra small

1.3.2.

// <div class="col-md-6">

The md in   class="col-md-6"  means the screen size is 992px to 1199px

12 columns grid system divide this screen to 12 columns,

The 6 in   class="col-md-6" means this div occupy 6 columns out of 12 columns grid system.

Therefore,

// <div class="col-md-6">

//     ...

// </div>

// <div class="col-md-6">

//     <h2>The Map</h2>

// </div>

It will become left half and right half when it is full screen.

However, when reducing  screen size less than 11 columns,

then it will become top and bottom.

1.3.3.

If we do this

// <div class="col-md-6 col-xs-8">

//     ...

// </div>

// <div class="col-md-6 col-xs-4">

//     <h2>The Map</h2>

// </div>

The  .col-xs-XX   will override  .col-md-XX

However, we actually only need big one, .col-md-XX

because for small screen device like phone,

we want it automatically align vertically.

so let's only use   .col-md-XX   and delete   .col-xs-XX

2.

// <div class="form-group">

//    <label>Date</label>

//    <input class="form-control"/>

// </div>

// ...

// <input type="submit" value="Add" class="btn btn-success"/>

2.1.

.form-group in bootstrap normally used for group label and input.

2.2.

.form-control in bootstrap will occupy the 100% width

2.3.

.btn-success in bootstrap will become green btn

2.4.

btn-danger in bootstrap will become red btn

2.5.

btn-warning in bootstrap will become orange btn

2.6.

btn-default in bootstrap will become gray btn

3.

3.1.

//<input type="submit" value="Create" class="btn btn-default" />

It is the submit button of the form.

3.2.

//@using (Html.BeginForm())

//@using (Html.BeginForm("Create", "Gamer"))

These two using will create

//<form action="/Gamer/Create3" method="post" novalidate="novalidate">

It will run the "Gamer" controller and "Create" HttpPost action.

//@using (Html.BeginForm())

It does not specify any controller or any action.

Thus, it will use the default controller and default HttpPost action.

It is Create.cshtml in Views/Gamer folder.

Thus, it will run "Gamer" controller and "Create" HttpPost action

3.3.

//@using (Html.BeginForm("Create2", "Gamer"))

It will create

//<form action="/Gamer/Create2" method="post" novalidate="novalidate">

It will run "Gamer" controller and "Create2" HttpPost action.

4.

4.1.

//@Html.LabelFor(model => model.Name, new { @class = "control-label col-md-2" })

It will create

//<label class="control-label col-md-2" for="Name">Name</label>

4.2.

//@Html.EditorFor(model => model.Name, new { htmlAttributes = new { @class = "form-control" } })

It will create

//<input class="form-control text-box single-line" id="Name" name="Name" type="text" value="">

4.3.

//@Html.ValidationMessageFor(model => model.Name, "", new { @class = "text-danger" })

It will create

//<span class="field-validation-valid text-danger" data-valmsg-for="Name" data-valmsg-replace="true"></span>

The cshtml also use the following JS

//<script src="~/Scripts/jquery-1.10.2.min.js"></script>

//<script src="~/Scripts/jquery.validate.min.js"></script>

//<script src="~/Scripts/jquery.validate.unobtrusive.min.js"></script>

Thus, the span will display the error message of Name input.

4.4.

//@Html.DropDownList("Gender",new List<SelectListItem>

//{

//    new SelectListItem{Text = "Male", Value = "Male"},

//    new SelectListItem{Text = "Female", Value = "Female"}

//})

It will create

//<select id="Gender" name="Gender">

//    <option value="Male">Male</option>

//    <option value="Female">Female</option>

//</select>

4.5.

//@Html.DropDownList("Gender", new List<SelectListItem>

//{

//    new SelectListItem{Text = "Male", Value = "Male"},

//    new SelectListItem{Text = "Female", Value = "Female"}

//}, "Select Gender")

It will create

//<select id="Gender" name="Gender">

//    <option value="">Select Gender</option>

//    <option value="Male">Male</option>

//    <option value="Female">Female</option>

//</select>

4.6.

//@Html.ActionLink("Back to List", "Index2")

It will create

//<a href="/Gamer/Index2">Back to List</a>

\*@

Graphical user interface, application

Description automatically generated

3.4. Models/Gamers/Gamer.cs

using System.ComponentModel.DataAnnotations;

namespace OnlineGame.Web.Models

{

    [MetadataType(typeof(GamerMetaData))]

    public partial class Gamer

    {

        //[Required]

        //public string Name { get; set; }

        //[Required]

        //public string Gender { get; set; }

        //[Required]

        //public string City { get; set; }

        //[Required]

        //[Display(Name = "Team")]

        //public int TeamId { get; set; }

        //// Error!!

        //// Memeber with the same name is areadly declared in other auto-generated partial class.

        //// Thus, you need MetadataType to add extra code for the Property.

        //// E.g. [MetadataType(typeof(GamerMetaData))

        //// In this case, you may add some extra code for the Property in MetadataType class

    }

}

3.5. Models/Gamers/GamerMetaData.cs

using System;

using System.ComponentModel.DataAnnotations;

namespace OnlineGame.Web.Models

{

    public class GamerMetaData

    {

        // Here is the place you may add some extra code for the property

        // which is already in the auto-generate partail class.

        //[Required]

        public string Name { get; set; }

        [Required]

        public string Gender { get; set; }

        [Required]

        public string City { get; set; }

        [Required]

        [Display(Name = "Team")]

        public Nullable<int> TeamId { get; set; }

        //In the database, TeamId is Nullable,

        //so the [Required] attibure here will not affect any thing.

        //If the TeamId in database is not Nullable,

        //then without [Display(Name = "Team")] attibute,

        //the validation message will display "TeamId is required".

        //if it is with [Display(Name = "Team")] attibute,

        //then validation message will display "Team is required".

    }

}

3.6. Views/Gamers/Edit.cshtml

@model OnlineGame.Web.Models.Gamer

@{

    ViewBag.Title = "Edit";

}

<h2>Edit</h2>

@using (Html.BeginForm())

{

    @Html.AntiForgeryToken()

    <div class="form-horizontal">

        <h4>Gamer</h4>

        <hr />

        @Html.ValidationSummary(true, "", new { @class = "text-danger" })

        @Html.HiddenFor(model => model.Id)

        <div class="form-group">

            @Html.LabelFor(model => model.Name, new { @class = "control-label col-md-2" })

            <div class="col-md-10">

                @\*@Html.EditorFor(model => model.Name, new { htmlAttributes = new { @class = "form-control" } })\*@

                @Html.HiddenFor(model => model.Name, new { htmlAttributes = new { @class = "form-control" } })

                @Html.DisplayFor(model => model.Name, new { htmlAttributes = new { @class = "form-control" } })

                @Html.ValidationMessageFor(model => model.Name, "", new { @class = "text-danger" })

            </div>

        </div>

        <div class="form-group">

            @Html.LabelFor(model => model.Gender, new { @class = "control-label col-md-2" })

            <div class="col-md-10">

                @\*@Html.EditorFor(model => model.Gender, new { htmlAttributes = new { @class = "form-control" } })\*@

                @\*@Html.DropDownList("Gender",new List<SelectListItem>

                    {

                        new SelectListItem{Text = "Male", Value = "Male"},

                        new SelectListItem{Text = "Female", Value = "Female"}

                    })\*@

                @Html.DropDownList("Gender", new List<SelectListItem>

                    {

                        new SelectListItem{Text = "Male", Value = "Male"},

                        new SelectListItem{Text = "Female", Value = "Female"}

                    }, "Select Gender")

                @Html.ValidationMessageFor(model => model.Gender, "", new { @class = "text-danger" })

            </div>

        </div>

        <div class="form-group">

            @Html.LabelFor(model => model.City, new { @class = "control-label col-md-2" })

            <div class="col-md-10">

                @Html.EditorFor(model => model.City, new { htmlAttributes = new { @class = "form-control" } })

                @Html.ValidationMessageFor(model => model.City, "", new { @class = "text-danger" })

            </div>

        </div>

        <div class="form-group">

            @Html.LabelFor(model => model.DateOfBirth, new { @class = "control-label col-md-2" })

            <div class="col-md-10">

                @Html.EditorFor(model => model.DateOfBirth, new { htmlAttributes = new { @class = "form-control" } })

                @Html.ValidationMessageFor(model => model.DateOfBirth, "", new { @class = "text-danger" })

            </div>

        </div>

        <div class="form-group">

            @Html.LabelFor(model => model.TeamId, "TeamId", new { @class = "control-label col-md-2" })

            <div class="col-md-10">

                @Html.DropDownList("TeamId", null, "Select Team", new { @class = "form-control" })

                @Html.ValidationMessageFor(model => model.TeamId, "", new {@class = "text-danger"})

            </div>

        </div>

        <div class="form-group">

            <div class="col-md-offset-2 col-md-10">

                <input type="submit" value="Save" class="btn btn-default" />

            </div>

        </div>

    </div>

}

<div>

    @Html.ActionLink("Back to List", "Index")

</div>

<script src="~/Scripts/jquery-1.10.2.min.js"></script>

<script src="~/Scripts/jquery.validate.min.js"></script>

<script src="~/Scripts/jquery.validate.unobtrusive.min.js"></script>

@\*

1.

1.1.

//@Html.HiddenFor(model => model.Name, new { htmlAttributes = new { @class = "form-control" } })

It will create the following.

//<input data-val="true" data-val-required="The Name field is required." htmlattributes="{ class = form-control }" id="Name" name="Name" type="hidden" value="Name01 ABB">

1.2.

//@Html.DisplayFor(model => model.Name, new { htmlAttributes = new { @class = "form-control" } })

It will create the following.

//Name01 ABB

1.3.

//@Html.EditorFor(model => model.Name, new {htmlAttributes = new {@class = "form-control"}})

It will create the following.

//<input class="form-control text-box single-line valid" id="Name" name="Name" type="text" value="Name01 ABB">

\*@

3.7. Models/Teams/TeamTotals.cs

namespace OnlineGame.Web.Models

{

    public class TeamTotals

    {

        public string Name { get; set; }

        public int Total { get; set; }

    }

}

3.8. Controllers/GamersController.cs

using System.Collections.Generic;

using System.Data.Entity;

using System.Linq;

using System.Threading.Tasks;

using[System.Net](http://system.net/);

using System.Web.Mvc;

using OnlineGame.Web.Models;

namespace OnlineGame.Web.Controllers

{

    public class GamersController : Controller

    {

        private OnlineGameEntities db = new OnlineGameEntities();

        public ActionResult GamersByTeam()

        {

            ////db.Gamers.Include("Team")

            //Retrive the Gamers with their Team data.

            List<TeamTotals> teamTotals =

                db.Gamers.Include("Team")

                .GroupBy(g => g.Team.Name)

                .Select(gamer => new TeamTotals

                {

                    Name = gamer.Key,

                    Total = gamer.Count()

                }).ToList();

            return View(teamTotals);

        }

        // GET: Gamers

        public async Task<ActionResult> Index()

        {

            IQueryable<Gamer> gamers = db.Gamers.Include(g => g.Team);

            //return View(await gamers.ToListAsync());  //~/Views/Gamers/Index.cshtml

            //return View("Index", await gamers.ToListAsync());    //~/Views/Gamers/Index.cshtml

            //return View("Index.cshtml", await gamers.ToListAsync());    //  Error

            return View(await gamers.ToListAsync());

        }

        // GET: Gamers/Details/5

        public async Task<ActionResult> Details(int? id)

        {

            if (id == null)

            {

                //return BadRequest code.

                return new HttpStatusCodeResult(HttpStatusCode.BadRequest);

            }

            Gamer gamer = await db.Gamers.FindAsync(id);

            if (gamer == null)

            {

                //return HttpNotFound code.

                return HttpNotFound();

            }

            return View(gamer);

        }

        // GET: Gamers/Create

        public ActionResult Create()

        {

            //Use the collection of teams as the parameter to create SelectList

            //which value is Team Id and the text is Team Name.

            //ViewBag.TeamId will bind this SelectList to View Model TeamId property.

            ViewBag.TeamId = new SelectList(db.Teams, "Id", "Name");

            return View();

        }

        // POST: Gamers/Create

        // To protect from overposting attacks, please enable the specific properties you want to bind to, for

        // more details see <https://go.microsoft.com/fwlink/?LinkId=317598>.

        [HttpPost]

        [ValidateAntiForgeryToken]

        public async Task<ActionResult> Create([Bind(Include = "Id,Name,Gender,City,DateOfBirth,TeamId")] Gamer gamer)

        {

            // We don't allow Fiddler to compose the Post body to change Name property,

            //so we don't use [Required] attribute on Name property.

            //However, in Create mode, we want to set Name is required property.

            //Thus, we have to dynamically add the ModelState.AddModelError in Create action

            if (string.IsNullOrEmpty(gamer.Name))

            {

                ModelState.AddModelError("Name", "Name is required.");

            }

            if (ModelState.IsValid)

            {

                db.Gamers.Add(gamer);

                await db.SaveChangesAsync();

                return RedirectToAction("Index");

            }

            //Use the collection of teams as the parameter to create SelectList

            //which value is Team Id and the text is Team Name.

            //ViewBag.TeamId will bind this SelectList to View Model TeamId property.

            ViewBag.TeamId = new SelectList(db.Teams, "Id", "Name", gamer.TeamId);

            return View(gamer);

        }

        // GET: Gamers/Edit/5

        public async Task<ActionResult> Edit(int? id)

        {

            if (id == null)

            {

                //return BadRequest code.

                return new HttpStatusCodeResult(HttpStatusCode.BadRequest);

            }

            Gamer gamer = await db.Gamers.FindAsync(id);

            if (gamer == null)

            {

                //return HttpNotFound code.

                return HttpNotFound();

            }

            //Use the collection of teams as the parameter to create SelectList

            //which value is Team Id and the text is Team Name.

            //ViewBag.TeamId will bind this SelectList to View Model TeamId property.

            ViewBag.TeamId = new SelectList(db.Teams, "Id", "Name", gamer.TeamId);

            return View(gamer);

        }

        //1.

        // POST: Gamers/Edit/5

        // To protect from overposting attacks, please enable the specific properties you want to bind to, for

        // more details see <https://go.microsoft.com/fwlink/?LinkId=317598>.

        //2.

        ////public async Task<ActionResult> Edit([Bind(Include = "Id,Gender,City,DateOfBirth,TeamId")] Gamer gamer)

        //Only update properties in the list, and ignore rest of properties.

        //In this case, update will exclude the Name property.

        //Thus, The post body generated by Fiddler can not update Name property.

        [HttpPost]

        [ValidateAntiForgeryToken]

        public async Task<ActionResult> Edit([Bind(Include = "Id,Gender,City,DateOfBirth,TeamId")] Gamer gamer)

        {

            //Get the gamer

            Gamer gamerFromDb = db.Gamers.Single(g => g.Id == gamer.Id);

            //Update the gamerFromDb

            gamerFromDb.Id = gamer.Id;

            gamerFromDb.Gender = gamer.Gender;

            gamerFromDb.City = gamer.City;

            gamerFromDb.TeamId = gamer.TeamId;

            //In the beginning, gamer.Name is null.

            //In order to pass ModelState.IsValid,

            //we need to set value for gamer.Name

            gamer.Name = gamerFromDb.Name;

            if (ModelState.IsValid)

            {

                //Update the entity by gamerFromDb, and set state as EntityState.Modified

                db.Entry(gamerFromDb).State = EntityState.Modified;

                await db.SaveChangesAsync();    //Save changes.

                return RedirectToAction("Index");

            }

            //1.

            //if validation is failed, then stay in the same page.

            //2.

            //Use the collection of teams as the parameter to create SelectList

            //which value is Team Id and the text is Team Name.

            //ViewBag.TeamId will bind this SelectList to View Model TeamId property.

            ViewBag.TeamId = new SelectList(db.Teams, "Id", "Name", gamer.TeamId);

            return View(gamer);

        }

        // GET: Gamers/Delete/5

        public async Task<ActionResult> Delete(int? id)

        {

            if (id == null)

            {

                // bad request.

                return new HttpStatusCodeResult(HttpStatusCode.BadRequest);

            }

            //Get the gamers

            Gamer gamer = await db.Gamers.FindAsync(id);

            if (gamer == null)

            {

                //return HttpNotFound code.

                return HttpNotFound();

            }

            return View(gamer);

        }

        // POST: Gamers/Delete/5

        [HttpPost, ActionName("Delete")]

        [ValidateAntiForgeryToken]

        public async Task<ActionResult> DeleteConfirmed(int id)

        {

            Gamer gamer = await db.Gamers.FindAsync(id);

            if (gamer != null) db.Gamers.Remove(gamer);

            await db.SaveChangesAsync();

            return RedirectToAction("Index");

        }

        protected override void Dispose(bool disposing)

        {

            if (disposing)

            {

                db.Dispose();

            }

            base.Dispose(disposing);

        }

    }

}

3.9. Views/Gamers/GamersByTeam.cshtml

Graphical user interface, text, application, email

Description automatically generated

@using OnlineGame.Web.Models

@model IEnumerable<TeamTotals>

@{

    ViewBag.Title = "GamersByTeam";

}

<h2>GamersByTeam</h2>

<table>

    <tr>

        <th>

            @Html.DisplayNameFor(model => model.Name)

        </th>

        <th>

            @Html.DisplayNameFor(model => model.Total)

        </th>

    </tr>

    @foreach (TeamTotals item in Model)

    {

        <tr>

            <td>

                @Html.DisplayFor(modelItem => item.Name)

            </td>

            <td>

                @Html.DisplayFor(modelItem => item.Total)

            </td>

        </tr>

    }

</table>

<http://localhost:49271/Gamers/GamersByTeam>

