(T13)實做類似GridView、SearchBar(搜尋引擎)、Pagging(分頁)、Sorting(排序)、DeleteMultipleRows(執行多筆移除)、SelectAll,UnSelectAll(全選,全不選)  
CourseGUID: 8503b39c-5887-4634-8291-facfb3117924  
=======================================================================  
(T13)實做類似GridView、SearchBar(搜尋引擎)、Pagging(分頁)、Sorting(排序)、DeleteMultipleRows(執行多筆移除)、SelectAll,UnSelectAll(全選,全不選)

(T13-1)實做類似GridView、SearchBar(搜尋引擎) (1. to 4.)

(T13-2)實做類似GridView、Pagging(分頁) (5.)

(T13-3)實做類似GridView、Sorting(排序) (6.)

(T13-4)實做類似GridView、DeleteMultipleRows(執行多筆移除) (7.)

(T13-5)實做類似GridView、SelectAll,UnSelectAll(全選,全不選) (8.)  
=======================================================================  
0. What to learn

-----------

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

2.1.3. Web.config

-----------

3. OnlineGame.Web

3.1. ADO.Net Entity Data Model - Entity Framework

3.2. Controllers/GamersController.cs

-----------

4. OnlineGame.Web - Search Bar

4.1. Views/Gamer/Index.cshtml

4.2. Controllers/GamerController.cs

4.3. Run Search Bar

-----------

5. OnlineGame.Web - PagedList, PagedList.Mvc

5.1. Install NuGet Package

5.2. Controllers/GamerController.cs

5.3. Views/Gamer/Index.cshtml

5.4. Run Search Bar with paging

-----------

6. OnlineGame.Web - Sorting

6.1. Controllers/GamerController.cs

6.2. Views/Gamer/Index.cshtml

6.3. Run Search Bar, paging, and Sorting

-----------

7. OnlineGame.Web - Check box delete All

7.1. Controllers/GamerController.cs

7.2. Views/Gamer/Index.cshtml

7.3. Run Search Bar, paging, and Sorting

-----------

8. OnlineGame.Web - Check box delete All

8.1. Views/Gamer/Index.cshtml  
=======================================================================

0. What to learn

\* 哪尼?手寫搜尋引擎Search Bar搭配MVC實現有分頁的GridView，還可Sorting，可全選取On/Off並執行多筆移除。

\* 完全手寫一個Search Bar搜尋引擎。

\* 使用MVC實現像是Web Form的無敵GridView。

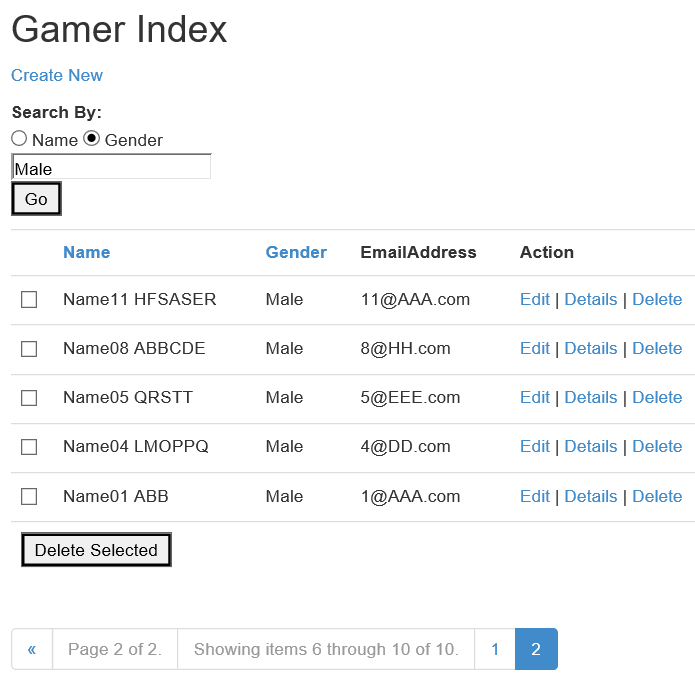
\* 實現GridView分業，排序，可全選On/Off，並執行多筆移除。

其他參考資料

MVC這裡我教了分頁

<http://kevintsengtw.blogspot.com/2014/11/pagedlistmvc-pager.html>

請把這個當作補充教材，這邊是把pagelist中文化



1. OnlineGame DB

1.1. TSQL

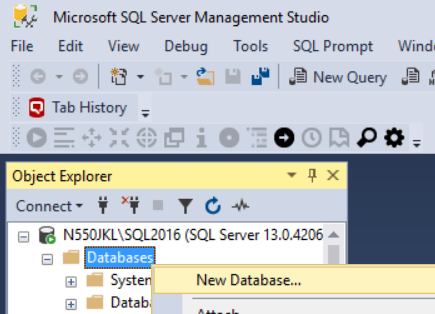
In SQL server Management Studio (SSMS)

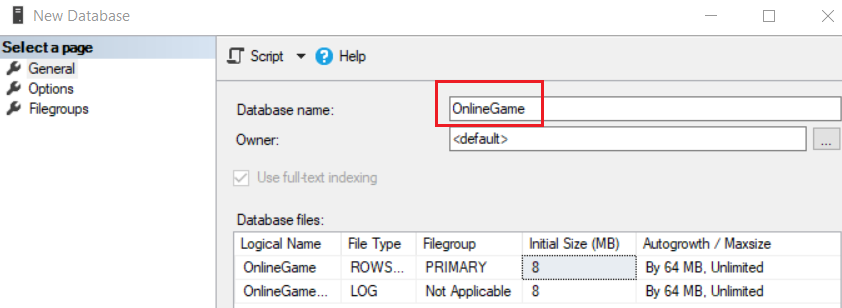
Database --> Right Click --> New Database -->

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

--2. Create Table

CREATE TABLE Gamer

(

   Id INT PRIMARY KEY

             IDENTITY(1, 1)

             NOT NULL ,

   [Name] NVARCHAR(100) NOT NULL ,

   Gender NVARCHAR(10) NOT NULL,

   EmailAddress nvarchar(100) NOT NULL,

)

--3. Insert Data

INSERT  Gamer

VALUES  ( N'Name01 ABB', N'Male', '[1@AAA.com](mailto:1@AAA.com)');

INSERT  Gamer

VALUES  ( N'Name02 CDDE', N'Female', '[2@BBB.com](mailto:2@BBB.com)');

INSERT  Gamer

VALUES  ( N'Name03 FIJK', N'Female', '[3@CCCC.com](mailto:3@CCCC.com)');

INSERT  Gamer

VALUES  ( N'Name04 LMOPPQ', N'Male', '[4@DD.com](mailto:4@DD.com)');

INSERT  Gamer

VALUES  ( N'Name05 QRSTT', N'Male', '[5@EEE.com](mailto:5@EEE.com)');

INSERT  Gamer

VALUES  ( N'Name06 TUVVX', N'Female', '[6@FF.com](mailto:6@FF.com)');

INSERT  Gamer

VALUES  ( N'Name07 XYZZXX', N'Female', '[7@GGGG.com](mailto:7@GGGG.com)');

INSERT  Gamer

VALUES  ( N'Name08 ABBCDE', N'Male', '[8@HH.com](mailto:8@HH.com)');

INSERT  Gamer

VALUES  ( N'Name09 QRSTTUVXX', N'Male', '[9@IIII.com](mailto:9@IIII.com)');

INSERT  Gamer

VALUES  ( N'Name10 GGAAEE', N'Male', '[10@XXWFFS.com](mailto:10@XXWFFS.com)');

INSERT  Gamer

VALUES  ( N'Name11 HFSASER', N'Male', '[11@AAA.com](mailto:11@AAA.com)');

INSERT  Gamer

VALUES  ( N'Name12 ESVSADC', N'Female', '[12@BBB.com](mailto:12@BBB.com)');

INSERT  Gamer

VALUES  ( N'Name13 REDSVF', N'Female', '[13@CCCC.com](mailto:13@CCCC.com)');

INSERT  Gamer

VALUES  ( N'Name14 BBGVDD', N'Male', '[14@DD.com](mailto:14@DD.com)');

INSERT  Gamer

VALUES  ( N'Name15 WWVFSSQ', N'Male', '[15@EEE.com](mailto:15@EEE.com)');

INSERT  Gamer

VALUES  ( N'Name16 TTVSS', N'Female', '[16@FF.com](mailto:16@FF.com)');

INSERT  Gamer

VALUES  ( N'Name17 AAQERR', N'Female', '[17@GGGG.com](mailto:17@GGGG.com)');

INSERT  Gamer

VALUES  ( N'Name18 BBFSAQ', N'Male', '[18@HH.com](mailto:18@HH.com)');

INSERT  Gamer

VALUES  ( N'Name19 QRSTTUVXX', N'Male', '[19@IIII.com](mailto:19@IIII.com)');

INSERT  Gamer

VALUES  ( N'Name20 HHFWSWQ', N'Male', '[20@XXWFFS.com](mailto:20@XXWFFS.com)');

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.









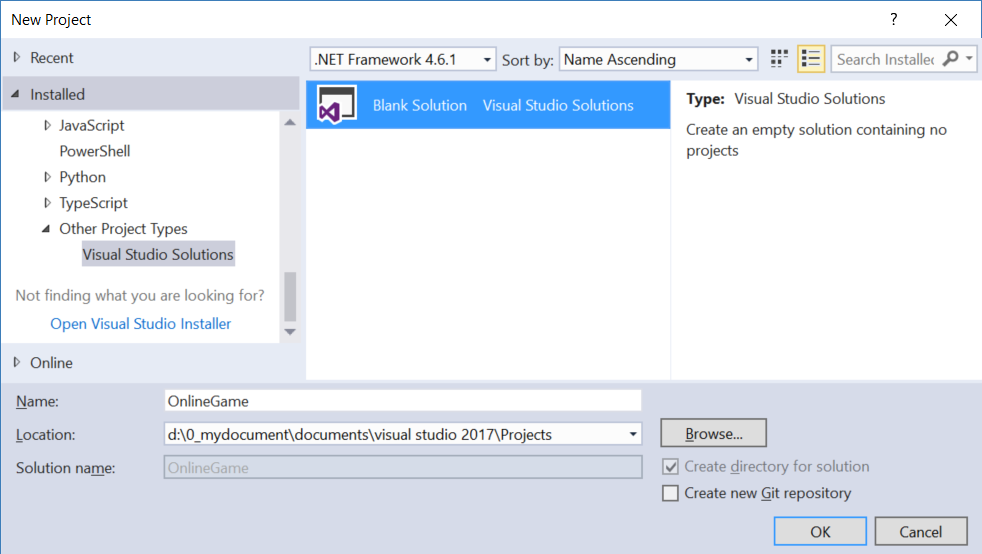
2. New Project - OnlineGame

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

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

-->

Name: **OnlineGame**



2.1. New Project - OnlineGame.Web

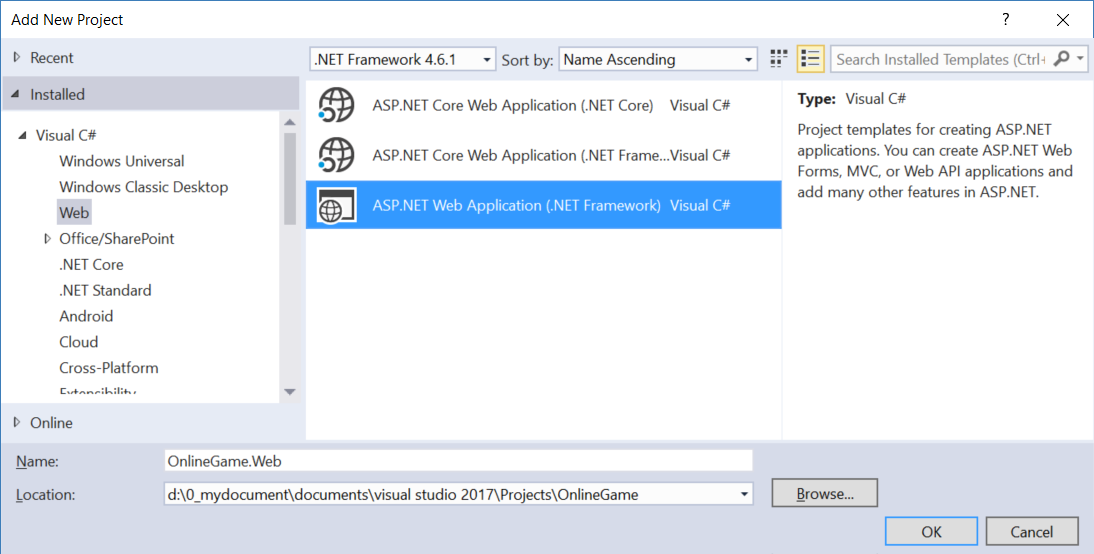
Solutions Name --> Add --> New Project -->

Visual C# --> Web --> [ASP.NET](http://asp.net/)Web Application (.Net Framework)

-->

Name: **OnlineGame.Web**

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



Graphical user interface, text, application

Description automatically generated

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 = "Gamer", 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.

\*/

2.1.3. Web.config

Graphical user interface, text, application

Description automatically generated

  <system.web>

    <globalization culture="en-au"/>

3. OnlineGame.Web

3.1. 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:

**OnlineGameDataModel**

-->

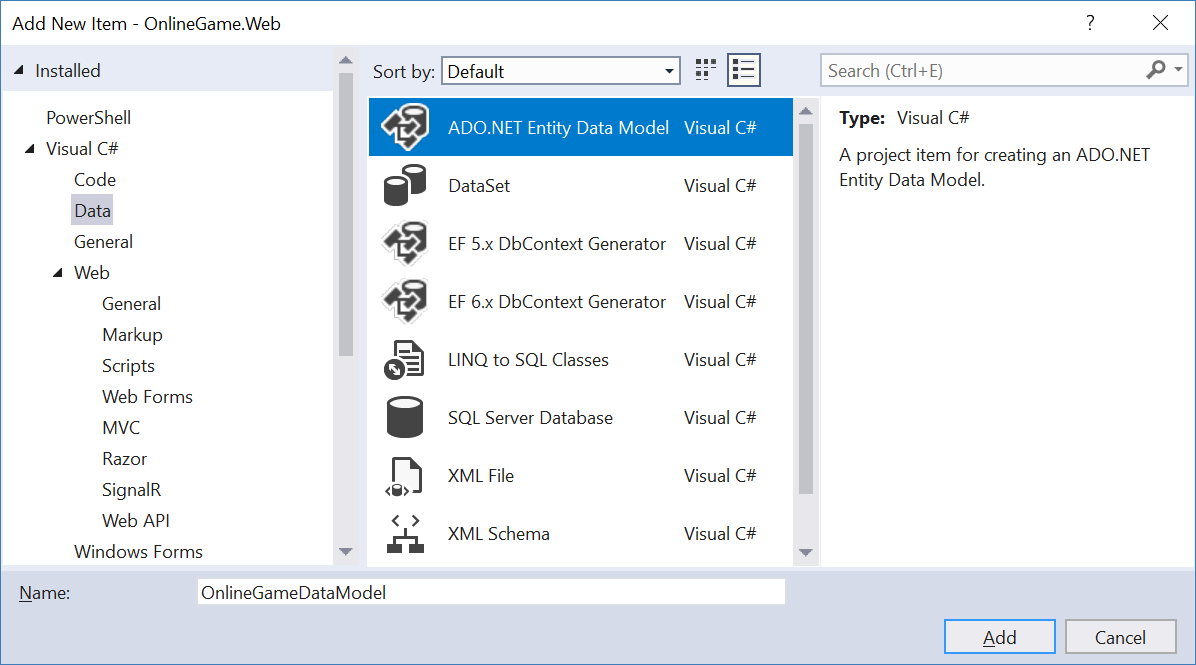
EF Designer from database

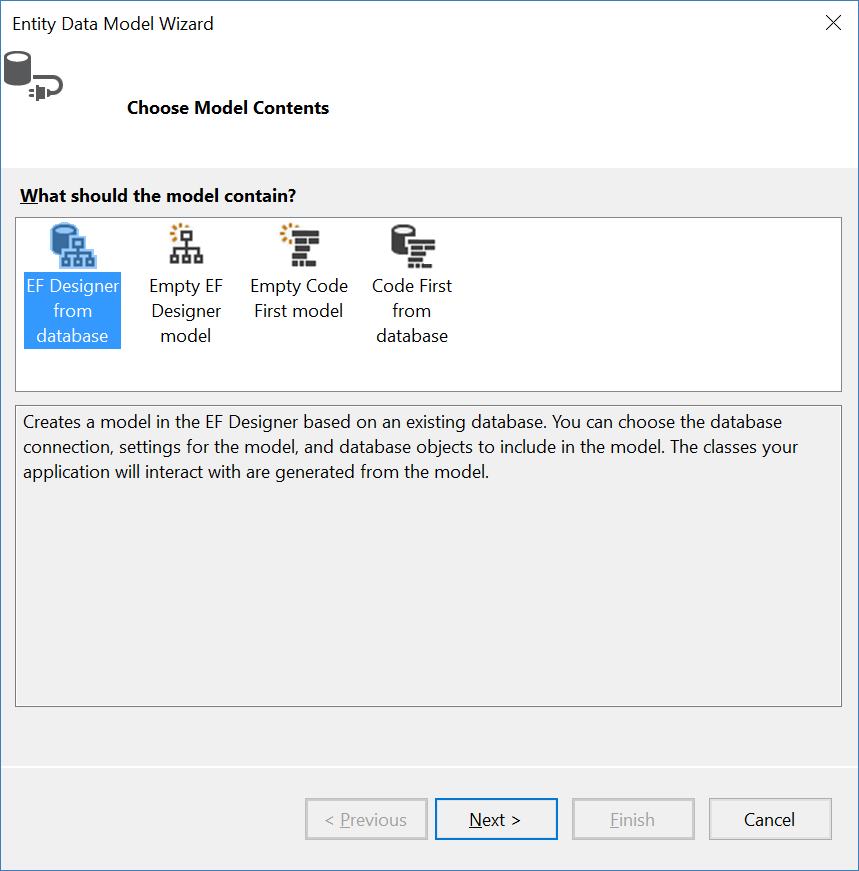
....

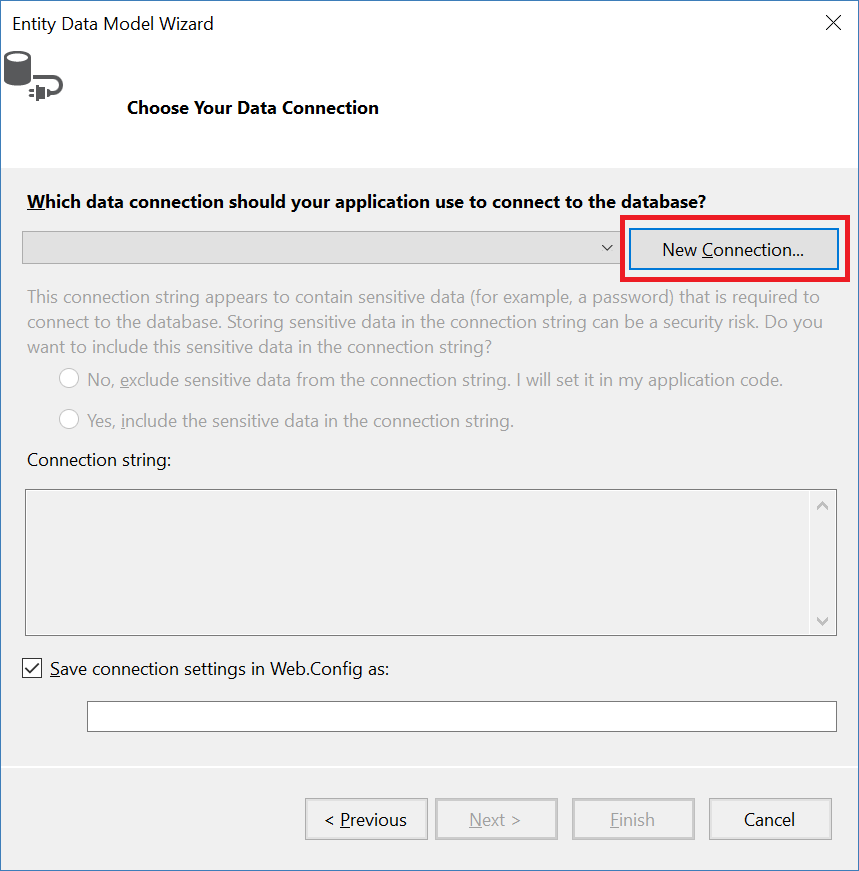
-->

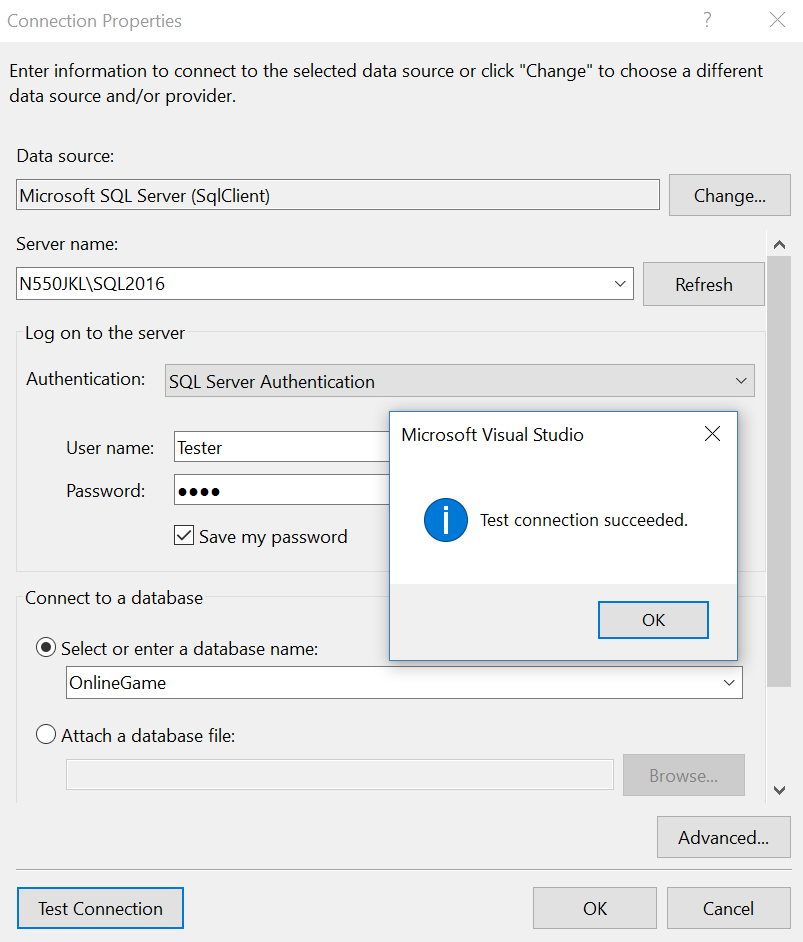
Save Connection settings in Web.Config as:

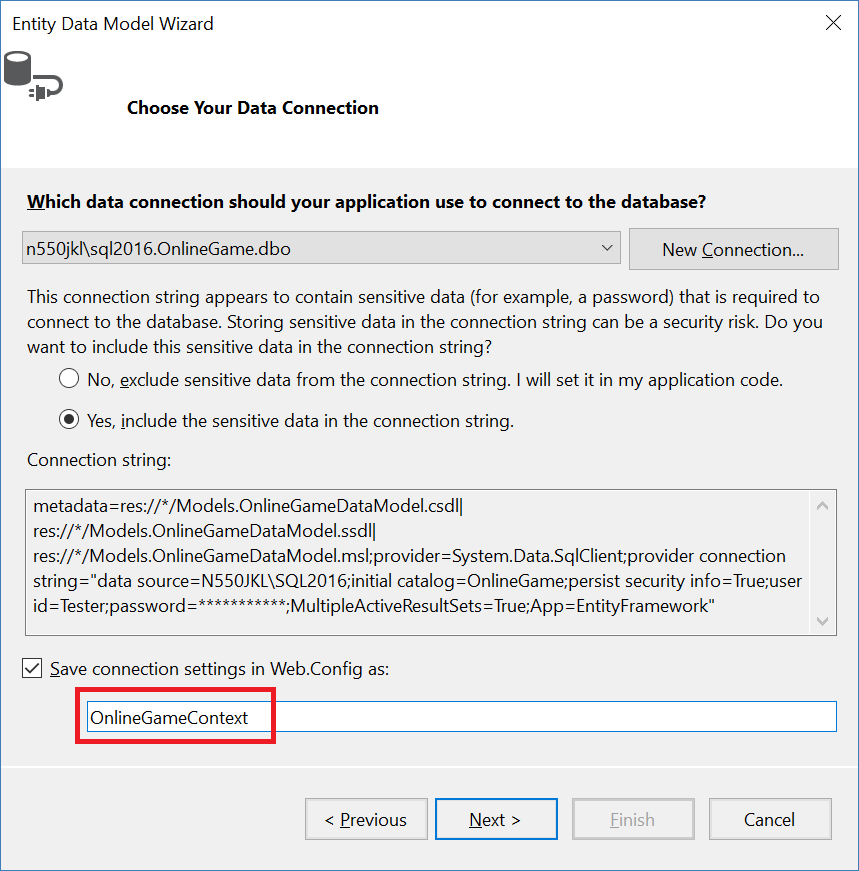
**OnlineGameContext**

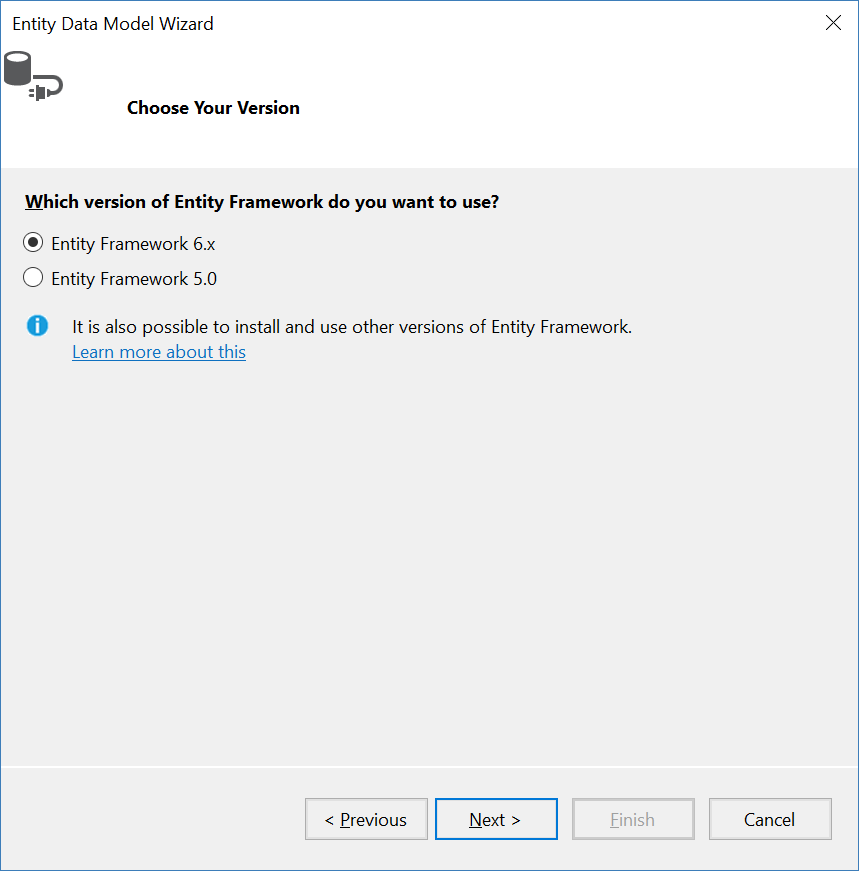


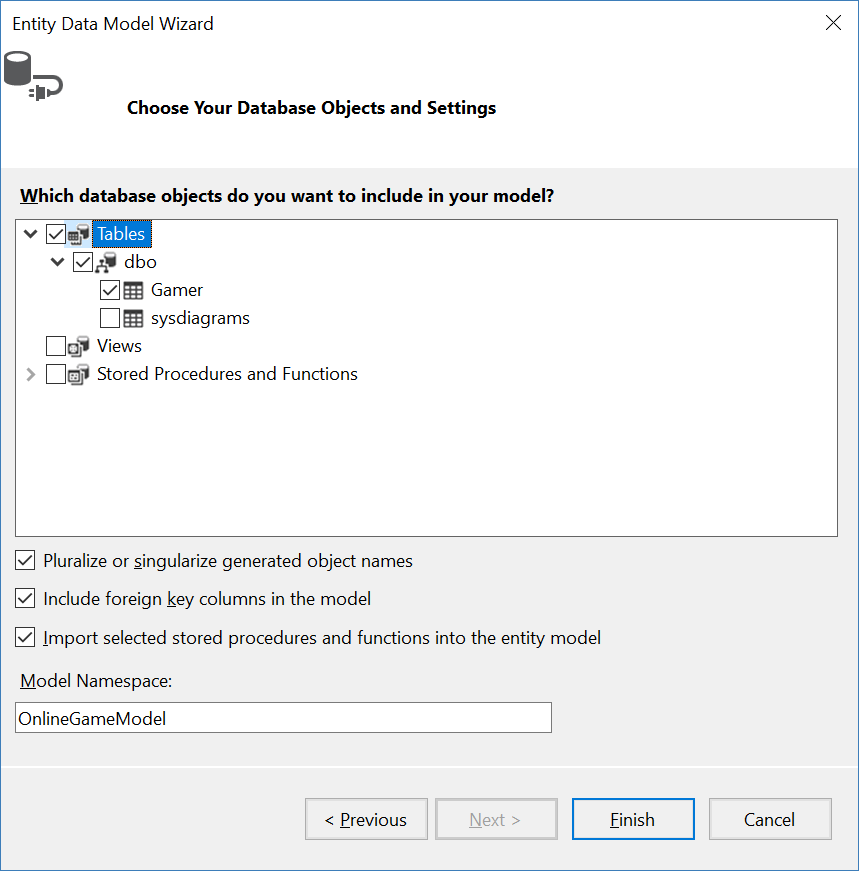


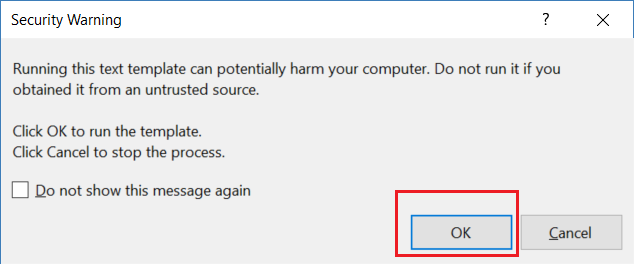


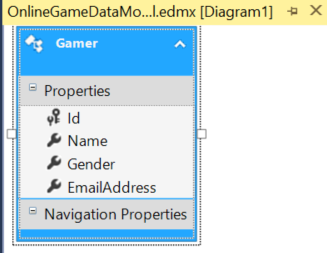










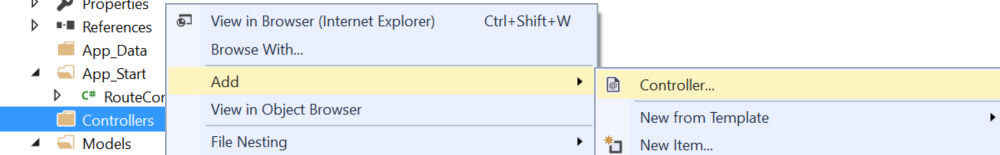


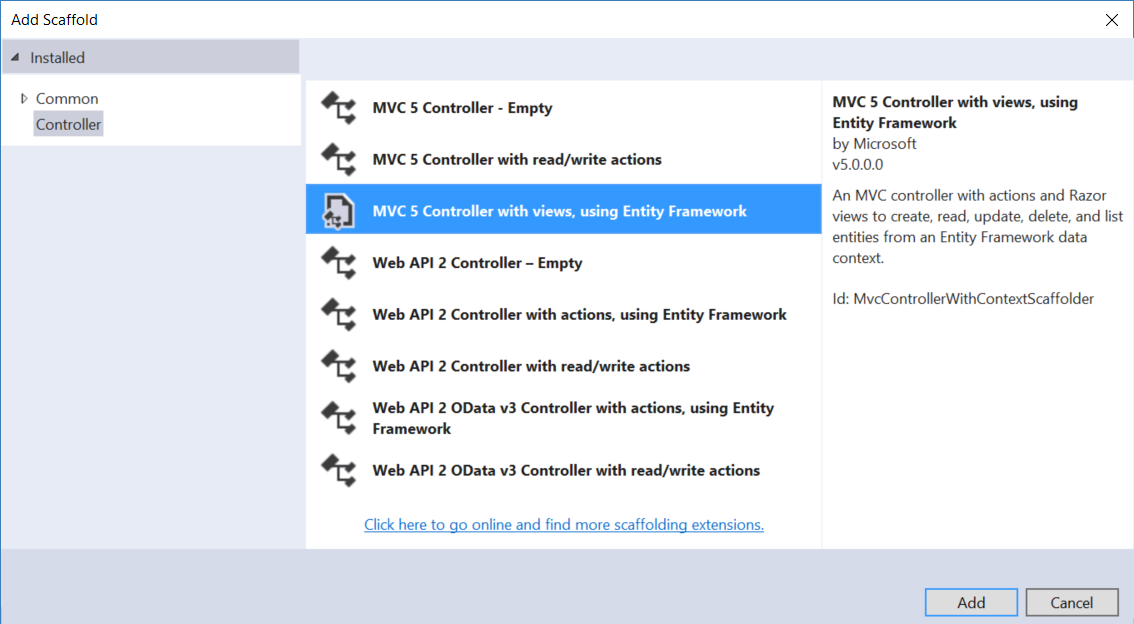
3.2. Controllers/GamersController.cs

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

-->

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





Graphical user interface, text, application, email

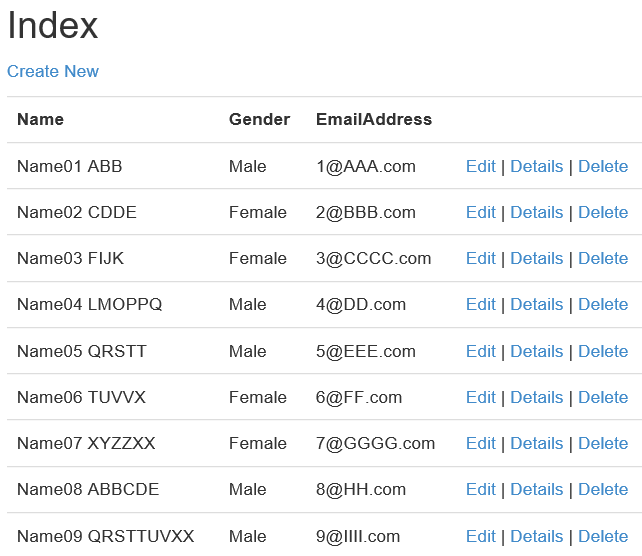
Description automatically generated

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

Graphical user interface, text, application

Description automatically generated with medium confidence

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



4. OnlineGame.Web - Search Bar

4.1. Views/Gamer/Index.cshtml

@model IEnumerable<OnlineGame.Web.Models.Gamer>

@{

    ViewBag.Title = "Gamer Index";

}

<h2>@ViewBag.Title</h2>

<p>

    @Html.ActionLink("Create New", "Create")

</p>

<p>

    @using (Html.BeginForm("Index", "Gamer", FormMethod.Get))

    {

        <b>Search By:</b><br />

        @Html.RadioButton("searchBy", "Name", true) <text>Name</text>

        @Html.RadioButton("searchBy", "Gender") <text>Gender</text><br />

        @Html.TextBox("searchText") <br />

        <input type="submit" value="Go" />

    }

</p>

<table class="table">

    <tr>

        <th>

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

        </th>

        <th>

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

        </th>

        <th>

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

        </th>

        <th>

            Action

        </th>

    </tr>

    @if (!Model.Any())

    {

        <tr>

            <td colspan="4">

                No matched records.

            </td>

        </tr>

    }

    @foreach (var item in Model)

    {

        <tr>

            <td>

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

            </td>

            <td>

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

            </td>

            <td>

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

            </td>

            <td>

                @Html.ActionLink("Edit", "Edit", new { id = item.Id }) |

                @Html.ActionLink("Details", "Details", new { id = item.Id }) |

                @Html.ActionLink("Delete", "Delete", new { id = item.Id })

            </td>

        </tr>

    }

</table>

4.2. Controllers/GamerController.cs

Modify the index action

// GET: Gamer

[HttpGet]

public async Task<ActionResult> Index(string searchBy, string searchText)

{

    List<Gamer> gamers = await db.Gamers.ToListAsync();

    if (searchBy == "Gender")

    {

        gamers = await db.Gamers

            .Where(x => x.Gender == searchText || searchText == null)

            .ToListAsync();

    }

    if (searchBy == "Name")

    {

        gamers = await db.Gamers

            .Where(x => x.Name.Contains(searchText) || searchText == null)

            .ToListAsync();

    }

    return View(gamers);

}

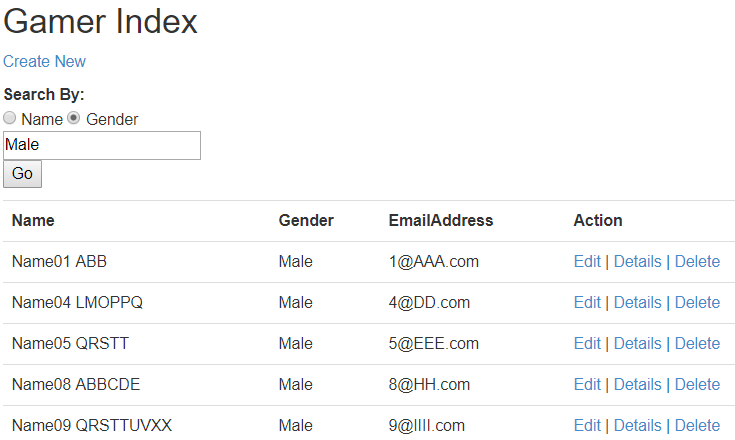
4.3. Run Search Bar

<http://localhost:52319/?searchBy=Name&searchText=01>

Graphical user interface, application, table

Description automatically generated

<http://localhost:52319/?searchBy=Gender&searchText=Male>



5. OnlineGame.Web - PagedList, PagedList.Mvc

5.1. Install NuGet Package

When we install "**PagedList.Mvc**", it will automatically install "**PagedList**".

Graphical user interface, application

Description automatically generated

5.2. Controllers/GamerController.cs

// GET: Gamer

[HttpGet]

public async Task<ActionResult> Index(string searchBy, string searchText, int? pageNumber)

{

    List<Gamer> gamers = await db.Gamers.ToListAsync();

    if (searchBy == "Gender")

    {

        gamers = await db.Gamers

            .Where(x => x.Gender == searchText || searchText == null)

            .ToListAsync();

    }

    if (searchBy == "Name")

    {

        gamers = await db.Gamers

            .Where(x => x.Name.Contains(searchText) || searchText == null)

            .ToListAsync();

    }

    //return View(gamers);

    //1.

    //The first parameter is pagenumber

    //pageNumber ?? 1 means if the pageNumber==null, then pageNumber==1

    //2.

    //The 2nd parameter is page size.

    //We set page size is 5.

    IPagedList<Gamer> gamerPagedList = gamers.ToPagedList(pageNumber ?? 1, 5);

    return View(gamerPagedList);

}

5.3. Views/Gamer/Index.cshtml

@using OnlineGame.Web.Models

@using PagedList

@using PagedList.Mvc

@\*@model IEnumerable<Gamer>\*@

@model IPagedList<Gamer>

@{

    ViewBag.Title = "Gamer Index";

}

<h2>@ViewBag.Title</h2>

<p>

    @Html.ActionLink("Create New", "Create")

</p>

<p>

    @using (Html.BeginForm("Index", "Gamer", FormMethod.Get))

    {

        <b>Search By:</b><br />

        @Html.RadioButton("searchBy", "Name", true) <text>Name</text>

        @Html.RadioButton("searchBy", "Gender") <text>Gender</text><br />

        @Html.TextBox("searchText") <br />

        <input type="submit" value="Go" />

    }

</p>

<table class="table">

    <tr>

        <th>

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

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

        </th>

        <th>

            @\*@Html.DisplayNameFor(model => model.Gender)\*@

            @Html.DisplayNameFor(model => model.First().Gender)

        </th>

        <th>

            @\*@Html.DisplayNameFor(model => model.EmailAddress)\*@

            @Html.DisplayNameFor(model => model.First().EmailAddress)

        </th>

        <th>

            Action

        </th>

    </tr>

    @if (!Model.Any())

    {

        <tr>

            <td colspan="4">

                No matched records.

            </td>

        </tr>

    }

    @foreach (var item in Model)

    {

        <tr>

            <td>

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

            </td>

            <td>

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

            </td>

            <td>

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

            </td>

            <td>

                @Html.ActionLink("Edit", "Edit", new { id = item.Id }) |

                @Html.ActionLink("Details", "Details", new { id = item.Id }) |

                @Html.ActionLink("Delete", "Delete", new { id = item.Id })

            </td>

        </tr>

    }

</table>

@\*@Html.PagedListPager(Model, pageNumber => Url.Action("Index",

    new

    {

        //pageNumber = pageNumber,

        pageNumber,

        searchBy = Request.QueryString["searchBy"],

        searchText = Request.QueryString["searchText"]

    }))\*@

@\*@Html.PagedListPager(Model, pageNumber => Url.Action("Index",

    new

    {

        //pageNumber = pageNumber,

        pageNumber,

        searchBy = Request.QueryString["searchBy"],

        searchText = Request.QueryString["searchText"]

    }),

    new PagedListRenderOptions{ Display = PagedListDisplayMode.IfNeeded })\*@

@\*@Html.PagedListPager(Model, pageNumber => Url.Action("Index",

    new

    {

        //pageNumber = pageNumber,

        pageNumber,

        searchBy = Request.QueryString["searchBy"],

        searchText = Request.QueryString["searchText"]

    }),

    new PagedListRenderOptions

    {

        Display = PagedListDisplayMode.IfNeeded,

        DisplayPageCountAndCurrentLocation = true

    })\*@

@Html.PagedListPager(Model, pageNumber => Url.Action("Index",

    new

    {

//pageNumber = pageNumber,

pageNumber,

        searchBy = Request.QueryString["searchBy"],

        searchText = Request.QueryString["searchText"]

    }),

    new PagedListRenderOptions

    {

        Display = PagedListDisplayMode.IfNeeded,

        DisplayPageCountAndCurrentLocation = true,

        DisplayItemSliceAndTotal = true

    })

@\*

1.

//@Html.PagedListPager(Model, pageNumber => Url.Action("Index",

//    new {

//        //pageNumber = pageNumber,

//        pageNumber,

//        searchBy = Request.QueryString["searchBy"],

//        searchText = Request.QueryString["searchText"] }))

1.1.

The PagedListPager() 1st parameter is IPagedList

which is the collection of gamers of current page.

1.2.

The PagedListPager() 2nd parameter is Func<int,string> generatePageURL

which we use Url.Action() to generate the Func<int,string>.

In this case, Func<int,string> means that

the input parameter is int, and the output is a string.

1.2.1.

The input parameter is the pageNumber which comes from the query string.

1.2.2.

The output is a string which is the generatePageURL.

In this case, it is Url.Action().

1.2.2.1.

The Url.Action() 1st parameter is action name which is "Index" action.

1.2.2.2.

The Url.Action() 2nd parameter is the route value.

1.2.2.2.1.

searchBy parameter should come from the query string, Request.QueryString["searchBy"].

1.2.2.2.2.

searchText parameter should come from the query string, Request.QueryString["searchText"].

1.3.

It will display the page number even there is only one page.

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

2.

//@Html.PagedListPager(Model, pageNumber => Url.Action("Index",

//    new

//    {

//        //pageNumber = pageNumber,

//        pageNumber,

//        searchBy = Request.QueryString["searchBy"],

//        searchText = Request.QueryString["searchText"]

//    }),

//    new PagedListRenderOptions{ Display = PagedListDisplayMode.IfNeeded })

2.1.

//Display = PagedListDisplayMode.IfNeeded,

It will hide the page number when there is only one page,

because only one page means no need paging.

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

3.

//@Html.PagedListPager(Model, pageNumber => Url.Action("Index",

//    new

//    {

//        //pageNumber = pageNumber,

//        pageNumber,

//        searchBy = Request.QueryString["searchBy"],

//        searchText = Request.QueryString["searchText"]

//    }),

//    new PagedListRenderOptions

//    {

//        Display = PagedListDisplayMode.IfNeeded,

//        DisplayPageCountAndCurrentLocation = true

//    })

3.1.

//Display = PagedListDisplayMode.IfNeeded,

It will hide the page number when there is only one page,

because only one page means no need paging.

3.2.

//DisplayPageCountAndCurrentLocation = true

It will display "Page 1 of 3"

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

4.

//@Html.PagedListPager(Model, pageNumber => Url.Action("Index",

//    new

//    {

//        //pageNumber = pageNumber,

//        pageNumber,

//        searchBy = Request.QueryString["searchBy"],

//        searchText = Request.QueryString["searchText"]

//    }),

//    new PagedListRenderOptions

//    {

//        Display = PagedListDisplayMode.IfNeeded,

//        DisplayPageCountAndCurrentLocation = true,

//        DisplayItemSliceAndTotal = true

//    })

4.1.

//Display = PagedListDisplayMode.IfNeeded,

It will hide the page number when there is only one page,

because only one page means no need paging.

4.2.

//DisplayPageCountAndCurrentLocation = true

It will display "Page 1 of 3"

4.3.

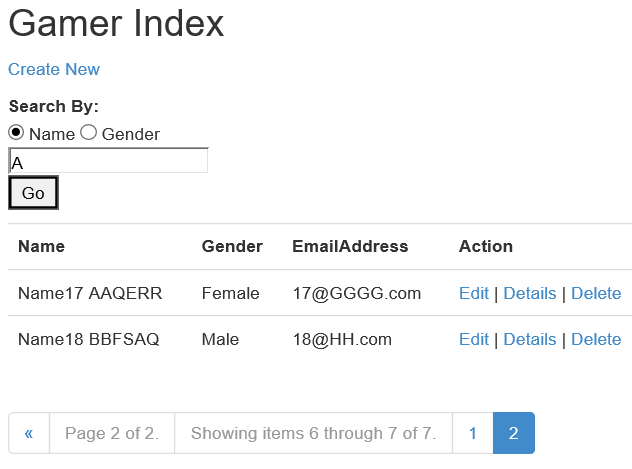
//DisplayItemSliceAndTotal = true

It will display "Showing items 6 through 7 of 7"

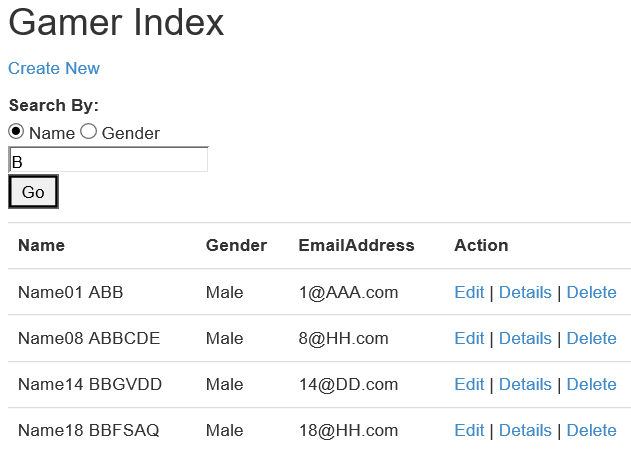
\*@

5.4. Run Search Bar with paging

<http://localhost:52319/?pageNumber=2&searchBy=Name&searchText=A>



<http://localhost:52319/?searchBy=Name&searchText=B>



6. OnlineGame.Web - Sorting

6.1. Controllers/GamerController.cs

// GET: Gamer

[HttpGet]

public async Task<ActionResult> Index(string searchBy, string searchText, int? pageNumber, string sortBy)

{

    ViewBag.NameSort = String.IsNullOrEmpty(sortBy) ? "Name desc" : "";

    ViewBag.GenderSort = sortBy == "Gender" ? "Gender desc" : "Gender";

    List<Gamer> gamers = await db.Gamers.ToListAsync();

    if (searchBy == "Gender")

    {

        gamers = await db.Gamers

            .Where(x => x.Gender == searchText || searchText == null)

            .ToListAsync();

    }

    if (searchBy == "Name")

    {

        gamers = await db.Gamers

            .Where(x => x.Name.Contains(searchText) || searchText == null)

            .ToListAsync();

    }

    IOrderedEnumerable<Gamer> gamersOrderedEnumerable;

    switch (sortBy)

    {

        case "Name desc":

            gamersOrderedEnumerable = gamers.OrderByDescending(x => x.Name);

            break;

        case "Gender desc":

            gamersOrderedEnumerable = gamers.OrderByDescending(x => x.Gender);

            break;

        case "Gender":

            gamersOrderedEnumerable = gamers.OrderBy(x => x.Gender);

            break;

        default:

            gamersOrderedEnumerable = gamers.OrderBy(x => x.Name);

            break;

    }

    //1.

    //The first parameter is pagenumber

    //pageNumber ?? 1 means if the pageNumber==null, then pageNumber==1

    //2.

    //The 2nd parameter is page size.

    //We set page size is 5.

    //IPagedList<Gamer> gamerPagedList = gamers.ToPagedList(pageNumber ?? 1, 5);

    IPagedList<Gamer> gamerPagedList = gamersOrderedEnumerable.ToPagedList(pageNumber ?? 1, 5);

    return View(gamerPagedList);

}

6.2. Views/Gamer/Index.cshtml

@using OnlineGame.Web.Models

@using PagedList

@using PagedList.Mvc

@\*@model IEnumerable<Gamer>\*@

@model IPagedList<Gamer>

@{

    ViewBag.Title = "Gamer Index";

}

<h2>@ViewBag.Title</h2>

<p>

    @Html.ActionLink("Create New", "Create")

</p>

<p>

    @using (Html.BeginForm("Index", "Gamer", FormMethod.Get))

    {

        <b>Search By:</b><br />

        @Html.RadioButton("searchBy", "Name", true) <text>Name</text>

        @Html.RadioButton("searchBy", "Gender") <text>Gender</text><br />

        @Html.TextBox("searchText") <br />

        <input type="submit" value="Go" />

    }

</p>

<table class="table">

    <tr>

        <th>

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

            @\*@Html.DisplayNameFor(model => model.First().Name)\*@

            @Html.ActionLink("Name", "Index", new

            {

                sortBy = ViewBag.NameSort,

                searchBy = Request.QueryString["searchBy"],

                searchText = Request.QueryString["searchText"],

            })

            @\*<a href="/?sortBy=Name%20desc">Name</a>\*@

        </th>

        <th>

            @\*@Html.DisplayNameFor(model => model.Gender)\*@

            @\*@Html.DisplayNameFor(model => model.First().Gender)\*@

            @Html.ActionLink("Gender", "Index", new

            {

                sortBy = ViewBag.GenderSort,

                searchBy = Request.QueryString["searchBy"],

                searchText = Request.QueryString["searchText"],

            })

            @\*<a href="/?sortBy=Gender">Gender</a>\*@

        </th>

        <th>

            @\*@Html.DisplayNameFor(model => model.EmailAddress)\*@

            @Html.DisplayNameFor(model => model.First().EmailAddress)

        </th>

        <th>

            Action

        </th>

    </tr>

    @if (!Model.Any())

    {

        <tr>

            <td colspan="4">

                No matched records.

            </td>

        </tr>

    }

    @foreach (var item in Model)

    {

        <tr>

            <td>

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

            </td>

            <td>

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

            </td>

            <td>

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

            </td>

            <td>

                @Html.ActionLink("Edit", "Edit", new { id = item.Id }) |

                @Html.ActionLink("Details", "Details", new { id = item.Id }) |

                @Html.ActionLink("Delete", "Delete", new { id = item.Id })

            </td>

        </tr>

    }

</table>

@\*@Html.PagedListPager(Model, pageNumber => Url.Action("Index",

    new

    {

        //pageNumber = pageNumber,

        pageNumber,

        searchBy = Request.QueryString["searchBy"],

        searchText = Request.QueryString["searchText"]

    }))\*@

@\*@Html.PagedListPager(Model, pageNumber => Url.Action("Index",

    new

    {

        //pageNumber = pageNumber,

        pageNumber,

        searchBy = Request.QueryString["searchBy"],

        searchText = Request.QueryString["searchText"]

    }),

    new PagedListRenderOptions{ Display = PagedListDisplayMode.IfNeeded })\*@

@\*@Html.PagedListPager(Model, pageNumber => Url.Action("Index",

    new

    {

        //pageNumber = pageNumber,

        pageNumber,

        searchBy = Request.QueryString["searchBy"],

        searchText = Request.QueryString["searchText"]

    }),

    new PagedListRenderOptions

    {

        Display = PagedListDisplayMode.IfNeeded,

        DisplayPageCountAndCurrentLocation = true

    })\*@

@Html.PagedListPager(Model, pageNumber => Url.Action("Index",

    new

    {

        //pageNumber = pageNumber,

        pageNumber,

        searchBy = Request.QueryString["searchBy"],

        searchText = Request.QueryString["searchText"],

        sortBy = Request["sortBy"]  //Request from ViewBag

    }),

    new PagedListRenderOptions

    {

        Display = PagedListDisplayMode.IfNeeded,

        DisplayPageCountAndCurrentLocation = true,

        DisplayItemSliceAndTotal = true

    })

@\*

1.

//@Html.PagedListPager(Model, pageNumber => Url.Action("Index",

//    new {

//        //pageNumber = pageNumber,

//        pageNumber,

//        searchBy = Request.QueryString["searchBy"],

//        searchText = Request.QueryString["searchText"] }))

1.1.

The PagedListPager() 1st parameter is IPagedList

which is the collection of gamers of current page.

1.2.

The PagedListPager() 2nd parameter is Func<int,string> generatePageURL

which we use Url.Action() to generate the Func<int,string>.

In this case, Func<int,string> means that

the input parameter is int, and the output is a string.

1.2.1.

The input parameter is the pageNumber which comes from the query string.

1.2.2.

The output is a string which is the generatePageURL.

In this case, it is Url.Action().

1.2.2.1.

The Url.Action() 1st parameter is action name which is "Index" action.

1.2.2.2.

The Url.Action() 2nd parameter is the route value.

1.2.2.2.1.

searchBy parameter should come from the query string, Request.QueryString["searchBy"].

1.2.2.2.2.

searchText parameter should come from the query string, Request.QueryString["searchText"].

1.3.

It will display the page number even there is only one page.

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

2.

//@Html.PagedListPager(Model, pageNumber => Url.Action("Index",

//    new

//    {

//        //pageNumber = pageNumber,

//        pageNumber,

//        searchBy = Request.QueryString["searchBy"],

//        searchText = Request.QueryString["searchText"]

//    }),

//    new PagedListRenderOptions{ Display = PagedListDisplayMode.IfNeeded })

2.1.

//Display = PagedListDisplayMode.IfNeeded,

It will hide the page number when there is only one page,

because only one page means no need paging.

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

3.

//@Html.PagedListPager(Model, pageNumber => Url.Action("Index",

//    new

//    {

//        //pageNumber = pageNumber,

//        pageNumber,

//        searchBy = Request.QueryString["searchBy"],

//        searchText = Request.QueryString["searchText"]

//    }),

//    new PagedListRenderOptions

//    {

//        Display = PagedListDisplayMode.IfNeeded,

//        DisplayPageCountAndCurrentLocation = true

//    })

3.1.

//Display = PagedListDisplayMode.IfNeeded,

It will hide the page number when there is only one page,

because only one page means no need paging.

3.2.

//DisplayPageCountAndCurrentLocation = true

It will display "Page 1 of 3"

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

4.

//@Html.PagedListPager(Model, pageNumber => Url.Action("Index",

//    new

//    {

//        //pageNumber = pageNumber,

//        pageNumber,

//        searchBy = Request.QueryString["searchBy"],

//        searchText = Request.QueryString["searchText"]

//    }),

//    new PagedListRenderOptions

//    {

//        Display = PagedListDisplayMode.IfNeeded,

//        DisplayPageCountAndCurrentLocation = true,

//        DisplayItemSliceAndTotal = true

//    })

4.1.

//Display = PagedListDisplayMode.IfNeeded,

It will hide the page number when there is only one page,

because only one page means no need paging.

4.2.

//DisplayPageCountAndCurrentLocation = true

It will display "Page 1 of 3"

4.3.

//DisplayItemSliceAndTotal = true

It will display "Showing items 6 through 7 of 7"

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

5.

//@Html.ActionLink("Name", "Index", new

//    {

//        sortBy = ViewBag.NameSort,

//        searchBy = Request.QueryString["searchBy"],

//        searchText = Request.QueryString["searchText"],

//    })

...

//@Html.ActionLink("Gender", "Index", new

//    {

//        sortBy = ViewBag.GenderSort,

//        searchBy = Request.QueryString["searchBy"],

//        searchText = Request.QueryString["searchText"],

//    })

5.1.

When <http://localhost:52319/Gamer/Index>

//<a href="/?sortBy=Name%20desc">Name</a>

//<a href="/?sortBy=Gender">Gender</a>

5.2.

<http://localhost:52319/?searchBy=Gender&searchText=Male>

//<a href="/?sortBy=Name%20desc">Name</a>

//<a href="/?sortBy=Gender">Gender</a>

5.3.

<http://localhost:52319/?sortBy=Name%20desc&searchBy=Gender&searchText=Male>

//<a href="/?searchBy=Gender&amp;searchText=Male">Name</a>

//<a href="/?sortBy=Gender&amp;searchBy=Gender&amp;searchText=Male">Gender</a>

\*@

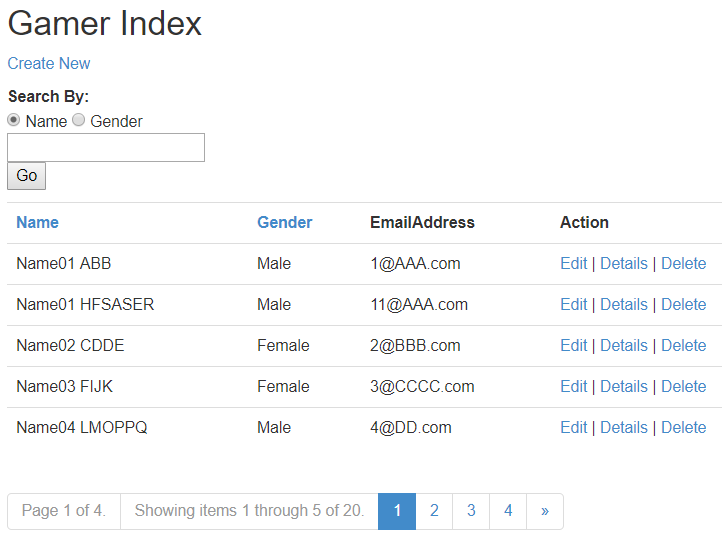
6.3. Run Search Bar, paging, and Sorting

When

<http://localhost:52319/Gamer/Index>

//<a href="/?sortBy=Name%20desc">Name</a>

//<a href="/?sortBy=Gender">Gender</a>



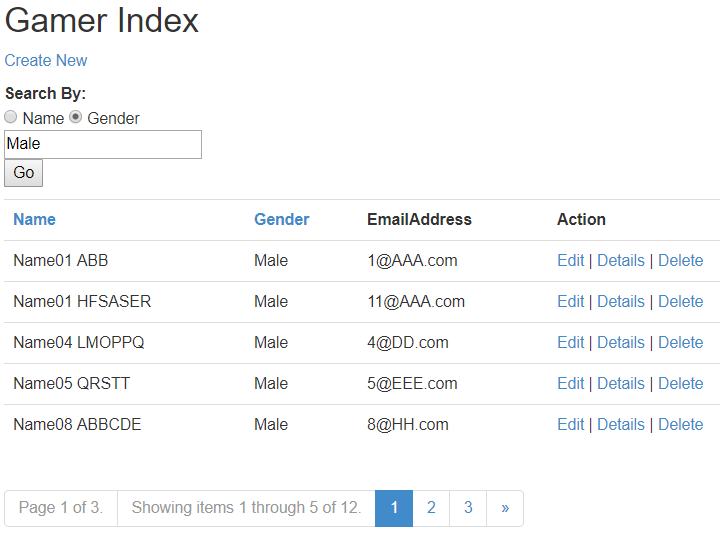
-----------------------------------------------------------------

When

<http://localhost:52319/?searchBy=Gender&searchText=Male>

//<a href="/?sortBy=Name%20desc">Name</a>

//<a href="/?sortBy=Gender">Gender</a>



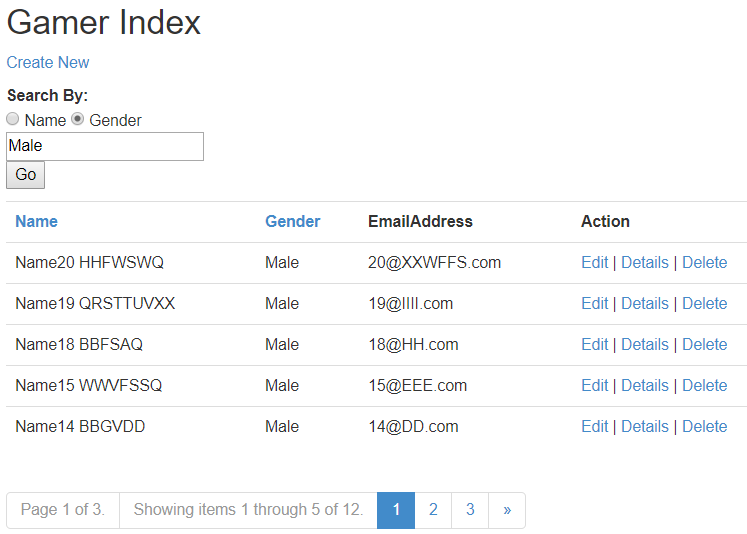
-----------------------------------------------------------------

When

<http://localhost:52319/?sortBy=Name%20desc&searchBy=Gender&searchText=Male>

//<a href="/?searchBy=Gender&amp;searchText=Male">Name</a>

//<a href="/?sortBy=Gender&amp;searchBy=Gender&amp;searchText=Male">Gender</a>



7. OnlineGame.Web - Check box delete All

7.1. Controllers/GamerController.cs

[HttpPost]

public async Task<ActionResult> DeleteMultiple(IEnumerable<int> GamerIdsToDelete, string searchBy, string searchText, int? pageNumber, string sortBy)

{

    //Delete a list of gamers

    List<Gamer> gamers = await db.Gamers.Where(g=> GamerIdsToDelete.Contains(g.Id)).ToListAsync();

    gamers.ForEach(g => db.Gamers.Remove(g));

    await db.SaveChangesAsync();

    return RedirectToAction("Index", new{ searchBy, searchText, pageNumber, sortBy });

}

7.2. Views/Gamer/Index.cshtml

@using OnlineGame.Web.Models

@using PagedList

@using PagedList.Mvc

@\*@model IEnumerable<Gamer>\*@

@model IPagedList<Gamer>

@{

    ViewBag.Title = "Gamer Index";

}

<h2>@ViewBag.Title</h2>

<p>

    @Html.ActionLink("Create New", "Create")

</p>

<p>

    @using (Html.BeginForm("Index", "Gamer", FormMethod.Get))

    {

        <b>Search By:</b><br />

        @Html.RadioButton("searchBy", "Name", true) <text>Name</text>

        @Html.RadioButton("searchBy", "Gender") <text>Gender</text><br />

        @Html.TextBox("searchText") <br />

        <input type="submit" value="Go" />

    }

</p>

@using (Html.BeginForm("DeleteMultiple", "Gamer",new

{

    searchBy = Request.QueryString["searchBy"],

    searchText = Request.QueryString["searchText"],

    pageNumber = Request.QueryString["pageNumber"],

    sortBy = Request["sortBy"]  //Request from ViewBag

}, FormMethod.Post))

{

<table class="table">

    <tr>

        <th></th>

        <th>

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

            @\*@Html.DisplayNameFor(model => model.First().Name)\*@

            @Html.ActionLink("Name", "Index", new

            {

                sortBy = ViewBag.NameSort,

                searchBy = Request.QueryString["searchBy"],

                searchText = Request.QueryString["searchText"],

            })

            @\*<a href="/?sortBy=Name%20desc">Name</a>\*@

        </th>

        <th>

            @\*@Html.DisplayNameFor(model => model.Gender)\*@

            @\*@Html.DisplayNameFor(model => model.First().Gender)\*@

            @Html.ActionLink("Gender", "Index", new

            {

                sortBy = ViewBag.GenderSort,

                searchBy = Request.QueryString["searchBy"],

                searchText = Request.QueryString["searchText"],

            })

            @\*<a href="/?sortBy=Gender">Gender</a>\*@

        </th>

        <th>

            @\*@Html.DisplayNameFor(model => model.EmailAddress)\*@

            @Html.DisplayNameFor(model => model.First().EmailAddress)

        </th>

        <th>

            Action

        </th>

    </tr>

    @if (!Model.Any())

    {

        <tr>

            <td colspan="5">

                No matched records.

            </td>

        </tr>

    }

    @foreach (Gamer item in Model)

    {

        <tr>

            <td>

                <input type="checkbox" name="GamerIdsToDelete" id="GamerIdsToDelete" value="@item.Id" />

            </td>

            <td>

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

            </td>

            <td>

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

            </td>

            <td>

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

            </td>

            <td>

                @Html.ActionLink("Edit", "Edit", new { id = item.Id }) |

                @Html.ActionLink("Details", "Details", new { id = item.Id }) |

                @Html.ActionLink("Delete", "Delete", new { id = item.Id })

            </td>

        </tr>

    }

    <tr>

        <td colspan="5">

            <input type="submit" value="Delete Selected" />

        </td>

    </tr>

</table>

}

@\*@Html.PagedListPager(Model, pageNumber => Url.Action("Index",

    new

    {

        //pageNumber = pageNumber,

        pageNumber,

        searchBy = Request.QueryString["searchBy"],

        searchText = Request.QueryString["searchText"]

    }))\*@

@\*@Html.PagedListPager(Model, pageNumber => Url.Action("Index",

    new

    {

        //pageNumber = pageNumber,

        pageNumber,

        searchBy = Request.QueryString["searchBy"],

        searchText = Request.QueryString["searchText"]

    }),

    new PagedListRenderOptions{ Display = PagedListDisplayMode.IfNeeded })\*@

@\*@Html.PagedListPager(Model, pageNumber => Url.Action("Index",

    new

    {

        //pageNumber = pageNumber,

        pageNumber,

        searchBy = Request.QueryString["searchBy"],

        searchText = Request.QueryString["searchText"]

    }),

    new PagedListRenderOptions

    {

        Display = PagedListDisplayMode.IfNeeded,

        DisplayPageCountAndCurrentLocation = true

    })\*@

@Html.PagedListPager(Model, pageNumber => Url.Action("Index",

    new

    {

//pageNumber = pageNumber,

pageNumber,

        searchBy = Request.QueryString["searchBy"],

        searchText = Request.QueryString["searchText"],

        sortBy = Request["sortBy"]  //Request from ViewBag

}),

    new PagedListRenderOptions

    {

        Display = PagedListDisplayMode.IfNeeded,

        DisplayPageCountAndCurrentLocation = true,

        DisplayItemSliceAndTotal = true

    })

@\*

1.

//@Html.PagedListPager(Model, pageNumber => Url.Action("Index",

//    new {

//        //pageNumber = pageNumber,

//        pageNumber,

//        searchBy = Request.QueryString["searchBy"],

//        searchText = Request.QueryString["searchText"] }))

1.1.

The PagedListPager() 1st parameter is IPagedList

which is the collection of gamers of current page.

1.2.

The PagedListPager() 2nd parameter is Func<int,string> generatePageURL

which we use Url.Action() to generate the Func<int,string>.

In this case, Func<int,string> means that

the input parameter is int, and the output is a string.

1.2.1.

The input parameter is the pageNumber which comes from the query string.

1.2.2.

The output is a string which is the generatePageURL.

In this case, it is Url.Action().

1.2.2.1.

The Url.Action() 1st parameter is action name which is "Index" action.

1.2.2.2.

The Url.Action() 2nd parameter is the route value.

1.2.2.2.1.

searchBy parameter should come from the query string, Request.QueryString["searchBy"].

1.2.2.2.2.

searchText parameter should come from the query string, Request.QueryString["searchText"].

1.3.

It will display the page number even there is only one page.

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

2.

//@Html.PagedListPager(Model, pageNumber => Url.Action("Index",

//    new

//    {

//        //pageNumber = pageNumber,

//        pageNumber,

//        searchBy = Request.QueryString["searchBy"],

//        searchText = Request.QueryString["searchText"]

//    }),

//    new PagedListRenderOptions{ Display = PagedListDisplayMode.IfNeeded })

2.1.

//Display = PagedListDisplayMode.IfNeeded,

It will hide the page number when there is only one page,

because only one page means no need paging.

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

3.

//@Html.PagedListPager(Model, pageNumber => Url.Action("Index",

//    new

//    {

//        //pageNumber = pageNumber,

//        pageNumber,

//        searchBy = Request.QueryString["searchBy"],

//        searchText = Request.QueryString["searchText"]

//    }),

//    new PagedListRenderOptions

//    {

//        Display = PagedListDisplayMode.IfNeeded,

//        DisplayPageCountAndCurrentLocation = true

//    })

3.1.

//Display = PagedListDisplayMode.IfNeeded,

It will hide the page number when there is only one page,

because only one page means no need paging.

3.2.

//DisplayPageCountAndCurrentLocation = true

It will display "Page 1 of 3"

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

4.

//@Html.PagedListPager(Model, pageNumber => Url.Action("Index",

//    new

//    {

//        //pageNumber = pageNumber,

//        pageNumber,

//        searchBy = Request.QueryString["searchBy"],

//        searchText = Request.QueryString["searchText"]

//    }),

//    new PagedListRenderOptions

//    {

//        Display = PagedListDisplayMode.IfNeeded,

//        DisplayPageCountAndCurrentLocation = true,

//        DisplayItemSliceAndTotal = true

//    })

4.1.

//Display = PagedListDisplayMode.IfNeeded,

It will hide the page number when there is only one page,

because only one page means no need paging.

4.2.

//DisplayPageCountAndCurrentLocation = true

It will display "Page 1 of 3"

4.3.

//DisplayItemSliceAndTotal = true

It will display "Showing items 6 through 7 of 7"

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

5.

//@Html.ActionLink("Name", "Index", new

//    {

//        sortBy = ViewBag.NameSort,

//        searchBy = Request.QueryString["searchBy"],

//        searchText = Request.QueryString["searchText"],

//    })

...

//@Html.ActionLink("Gender", "Index", new

//    {

//        sortBy = ViewBag.GenderSort,

//        searchBy = Request.QueryString["searchBy"],

//        searchText = Request.QueryString["searchText"],

//    })

5.1.

When <http://localhost:52319/Gamer/Index>

//<a href="/?sortBy=Name%20desc">Name</a>

//<a href="/?sortBy=Gender">Gender</a>

5.2.

<http://localhost:52319/?searchBy=Gender&searchText=Male>

//<a href="/?sortBy=Name%20desc">Name</a>

//<a href="/?sortBy=Gender">Gender</a>

5.3.

<http://localhost:52319/?sortBy=Name%20desc&searchBy=Gender&searchText=Male>

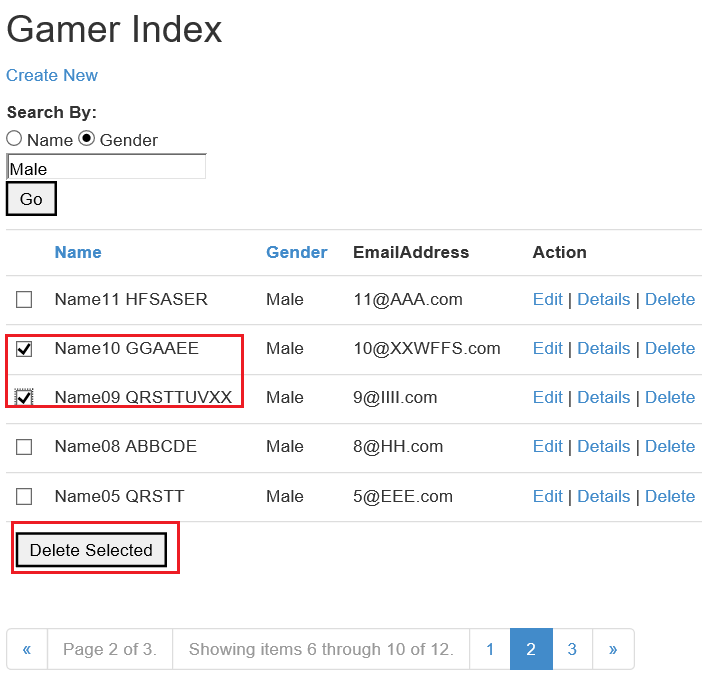
//<a href="/?searchBy=Gender&amp;searchText=Male">Name</a>

//<a href="/?sortBy=Gender&amp;searchBy=Gender&amp;searchText=Male">Gender</a>

\*@

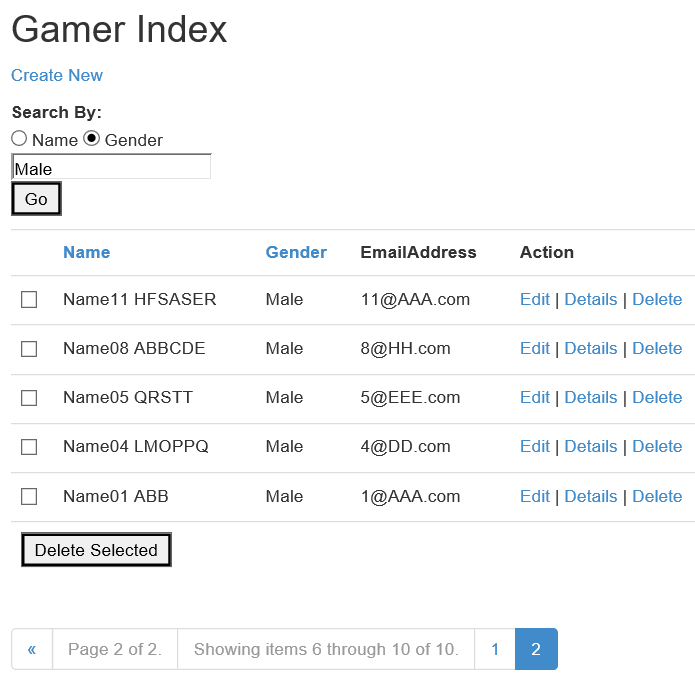
7.3. Run Search Bar, paging, and Sorting

<http://localhost:52319/?pageNumber=2&searchBy=Gender&searchText=Male&sortBy=Name%20desc>



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

<http://localhost:52319/?searchBy=Gender&searchText=Male&pageNumber=2&sortBy=Name%20desc>



8. OnlineGame.Web - Check box delete All

8.1. Views/Gamer/Index.cshtml

@using OnlineGame.Web.Models

@using PagedList

@using PagedList.Mvc

@\*@model IEnumerable<Gamer>\*@

@model IPagedList<Gamer>

@{

    ViewBag.Title = "Gamer Index";

}

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

<script type="text/javascript" language="javascript">

    $(function () {

        $('#SelectAll').click(function () {

            $("input[name='GamerIdsToDelete']").prop("checked", this.checked);

            //1.

            //this.checked means $("#SelectAll").checked

            //if $("#SelectAll").checked==true, then

            //$("input[name='GamerIdsToDelete']").prop("checked", true);

            //if $("#SelectAll").checked==false, then

            //$("input[name='GamerIdsToDelete']").prop("checked", false);

            //2.

            //$("input[name='GamerIdsToDelete']") will select all the elements which name==GamerIdsToDelete.

            //$("input['#GamerIdsToDelete']") will select all only one element which Id==GamerIdsToDelete.

            $("input[name='GamerIdsToDelete']").click(function () {

                if ($("input[name='GamerIdsToDelete']").length === $("input[name='GamerIdsToDelete']:checked").length)

                {

                    $("#SelectAll").prop("checked", "checked");

                }

                else {

                    $("#SelectAll").removeProp("checked");

                }

            });

            //1.

            //When any of name==GamerIdsToDelete elements has been ckicked,

            //if all input[name='GamerIdsToDelete'] have been checked,

            //then the $("#SelectAll") must be checked.

            //Otherwise, the $("#SelectAll") is un-checked.

        });

        $("#btnDeleteSelected").click(function () {

            var count = $("input[name='GamerIdsToDelete']:checked").length;

            if (count === 0) {

                alert("Please select items to delete.");

                return false;

            }

            else

            {

                return confirm(count + " row(s) will be deleted. Are you sure to continue.");

            }

        });

    });

</script>

<h2>@ViewBag.Title</h2>

<p>

    @Html.ActionLink("Create New", "Create")

</p>

<p>

    @using (Html.BeginForm("Index", "Gamer", FormMethod.Get))

    {

        <b>Search By:</b><br />

        @Html.RadioButton("searchBy", "Name", true) <text>Name</text>

        @Html.RadioButton("searchBy", "Gender") <text>Gender</text><br />

        @Html.TextBox("searchText") <br />

        <input type="submit" value="Go" />

    }

</p>

@using (Html.BeginForm("DeleteMultiple", "Gamer", new

{

    searchBy = Request.QueryString["searchBy"],

    searchText = Request.QueryString["searchText"],

    pageNumber = Request.QueryString["pageNumber"],

    sortBy = Request["sortBy"]  //Request from ViewBag

}, FormMethod.Post))

{

    <table class="table">

        <tr>

            <th>

                <input type="checkbox" name="SelectAll" id="SelectAll" />

            </th>

            <th>

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

                @\*@Html.DisplayNameFor(model => model.First().Name)\*@

                @Html.ActionLink("Name", "Index", new

                {

                    sortBy = ViewBag.NameSort,

                    searchBy = Request.QueryString["searchBy"],

                    searchText = Request.QueryString["searchText"],

                })

                @\*<a href="/?sortBy=Name%20desc">Name</a>\*@

            </th>

            <th>

                @\*@Html.DisplayNameFor(model => model.Gender)\*@

                @\*@Html.DisplayNameFor(model => model.First().Gender)\*@

                @Html.ActionLink("Gender", "Index", new

                {

                    sortBy = ViewBag.GenderSort,

                    searchBy = Request.QueryString["searchBy"],

                    searchText = Request.QueryString["searchText"],

                })

                @\*<a href="/?sortBy=Gender">Gender</a>\*@

            </th>

            <th>

                @\*@Html.DisplayNameFor(model => model.EmailAddress)\*@

                @Html.DisplayNameFor(model => model.First().EmailAddress)

            </th>

            <th>

                Action

            </th>

        </tr>

        @if (!Model.Any())

        {

            <tr>

                <td colspan="5">

                    No matched records.

                </td>

            </tr>

        }

        @foreach (Gamer item in Model)

        {

            <tr>

                <td>

                    <input type="checkbox" name="GamerIdsToDelete" id="GamerIdsToDelete" value="@item.Id" />

                </td>

                <td>

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

                </td>

                <td>

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

                </td>

                <td>

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

                </td>

                <td>

                    @Html.ActionLink("Edit", "Edit", new { id = item.Id }) |

                    @Html.ActionLink("Details", "Details", new { id = item.Id }) |

                    @Html.ActionLink("Delete", "Delete", new { id = item.Id })

                </td>

            </tr>

        }

        <tr>

            <td colspan="5">

                <input type="submit" value="Delete Selected" id="btnDeleteSelected" name="btnDeleteSelected"/>

            </td>

        </tr>

    </table>

}

@\*@Html.PagedListPager(Model, pageNumber => Url.Action("Index",

    new

    {

        //pageNumber = pageNumber,

        pageNumber,

        searchBy = Request.QueryString["searchBy"],

        searchText = Request.QueryString["searchText"]

    }))\*@

@\*@Html.PagedListPager(Model, pageNumber => Url.Action("Index",

    new

    {

        //pageNumber = pageNumber,

        pageNumber,

        searchBy = Request.QueryString["searchBy"],

        searchText = Request.QueryString["searchText"]

    }),

    new PagedListRenderOptions{ Display = PagedListDisplayMode.IfNeeded })\*@

@\*@Html.PagedListPager(Model, pageNumber => Url.Action("Index",

    new

    {

        //pageNumber = pageNumber,

        pageNumber,

        searchBy = Request.QueryString["searchBy"],

        searchText = Request.QueryString["searchText"]

    }),

    new PagedListRenderOptions

    {

        Display = PagedListDisplayMode.IfNeeded,

        DisplayPageCountAndCurrentLocation = true

    })\*@

@Html.PagedListPager(Model, pageNumber => Url.Action("Index",

    new

    {

//pageNumber = pageNumber,

pageNumber,

        searchBy = Request.QueryString["searchBy"],

        searchText = Request.QueryString["searchText"],

        sortBy = Request["sortBy"]  //Request from ViewBag

}),

    new PagedListRenderOptions

    {

        Display = PagedListDisplayMode.IfNeeded,

        DisplayPageCountAndCurrentLocation = true,

        DisplayItemSliceAndTotal = true

    })

Graphical user interface, application

Description automatically generated

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

Table

Description automatically generated

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

Graphical user interface, table

Description automatically generated

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

Graphical user interface, application

Description automatically generated

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

