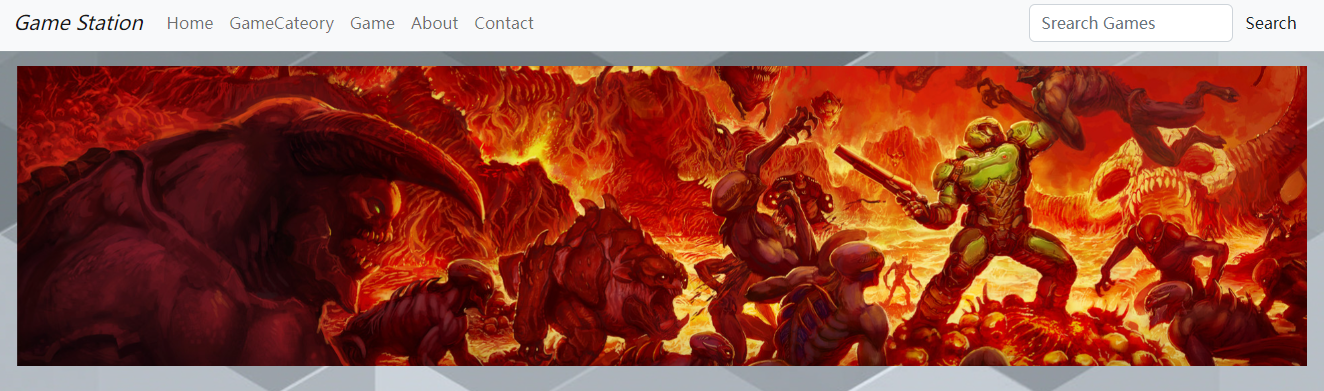
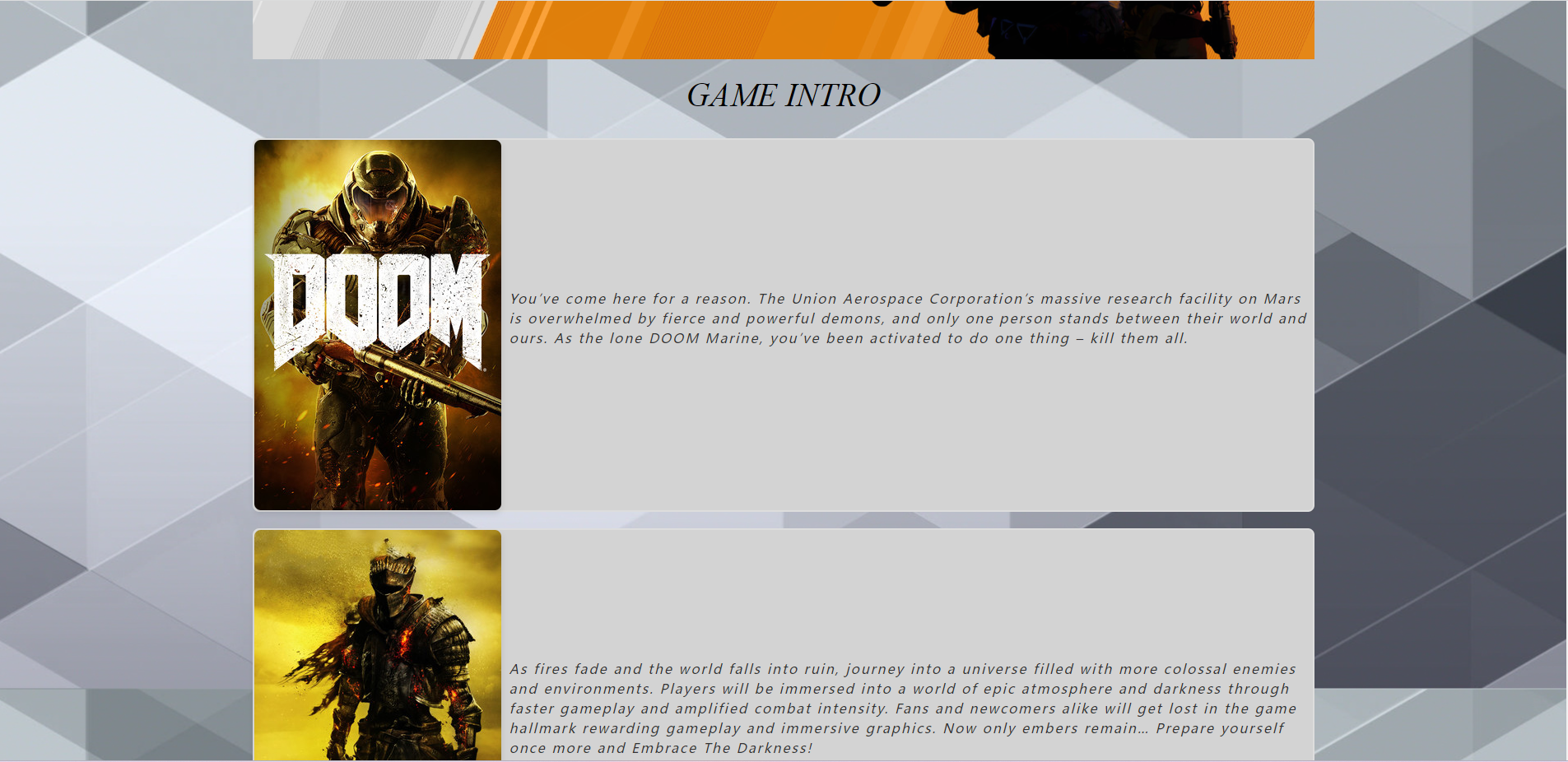
1. **Accessing the Website**:



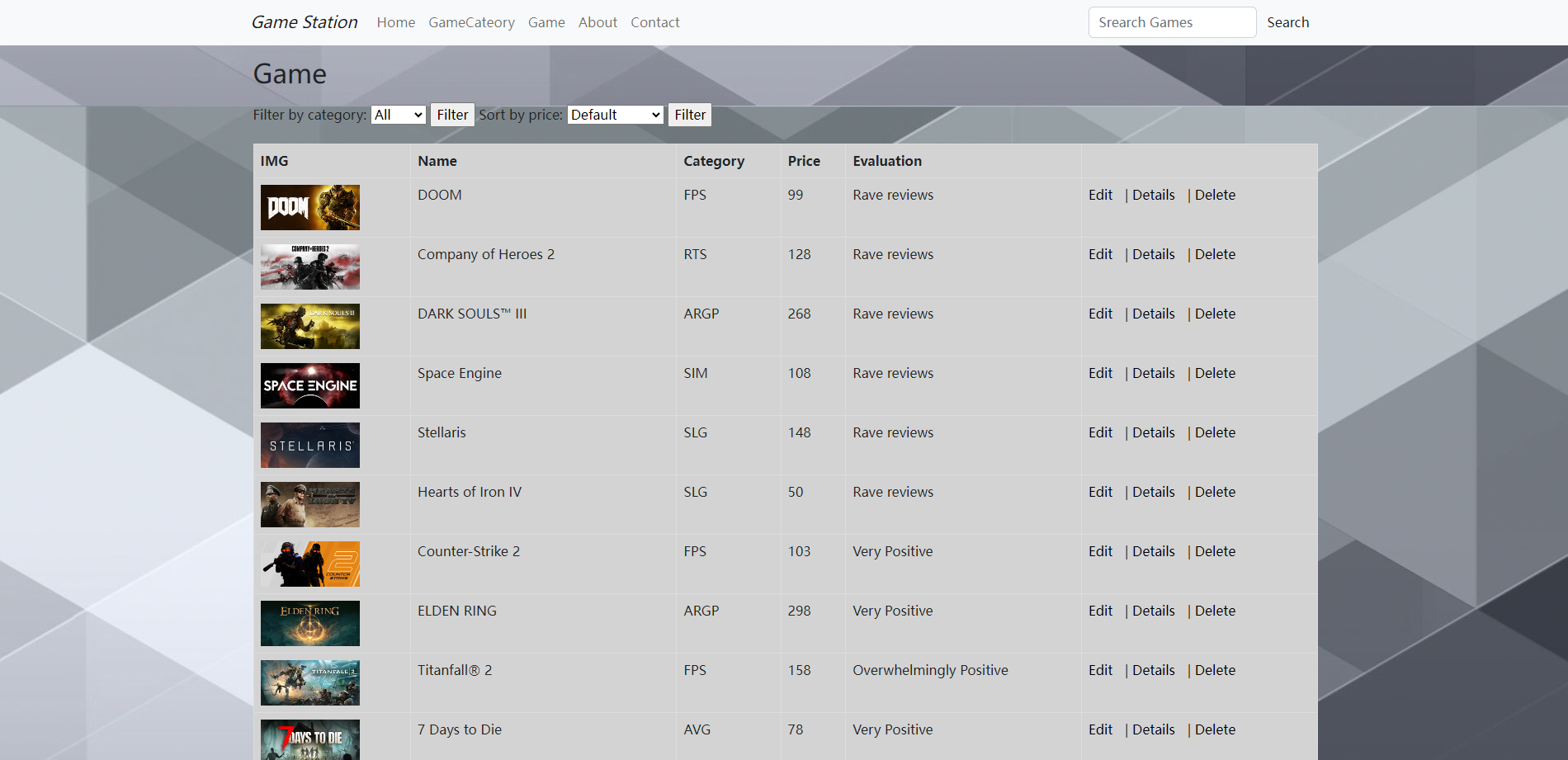
There is a navigation bar at the top of the web page, which is clear and straightforward.

1. **Website Layout**:

Home

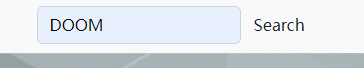


Game

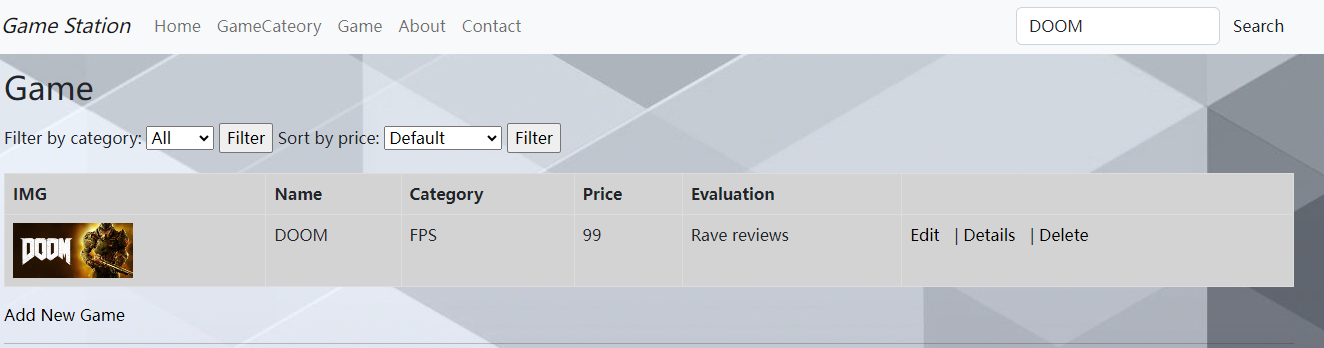


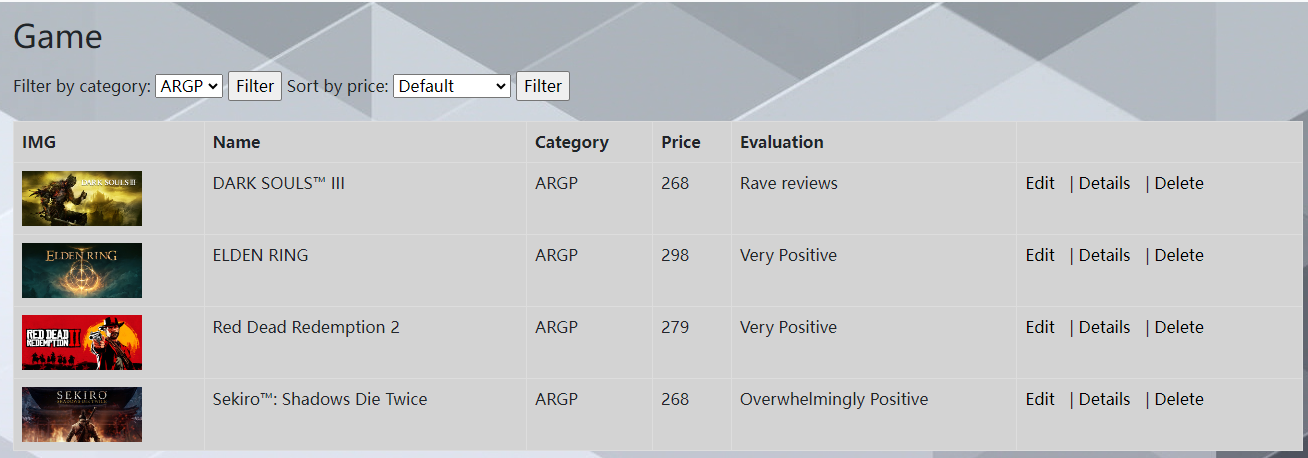
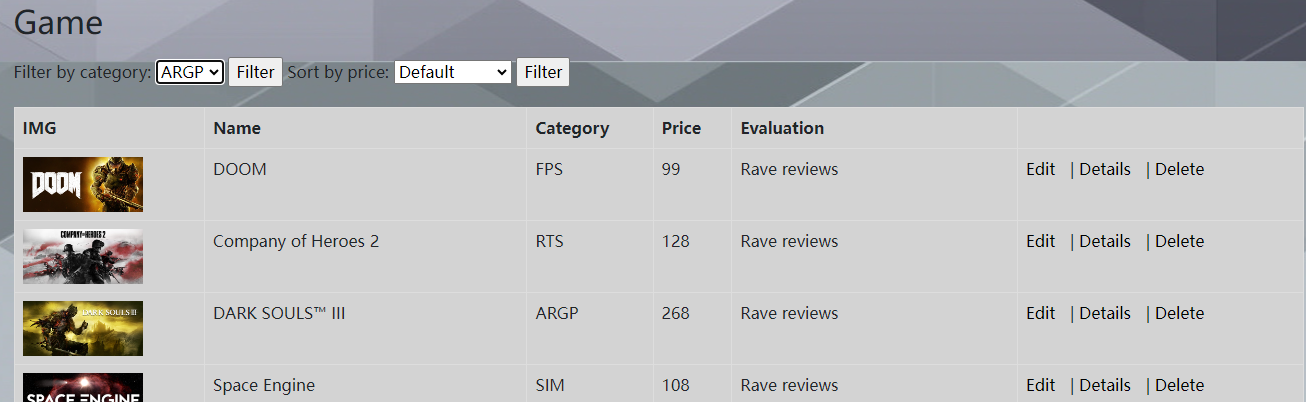
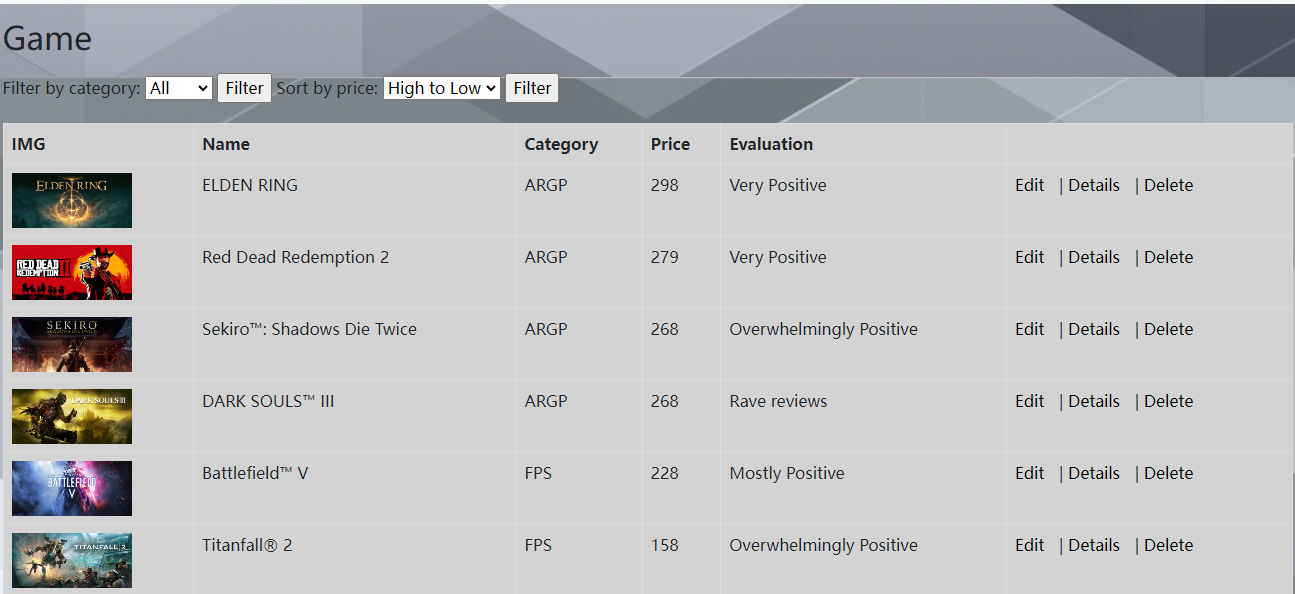
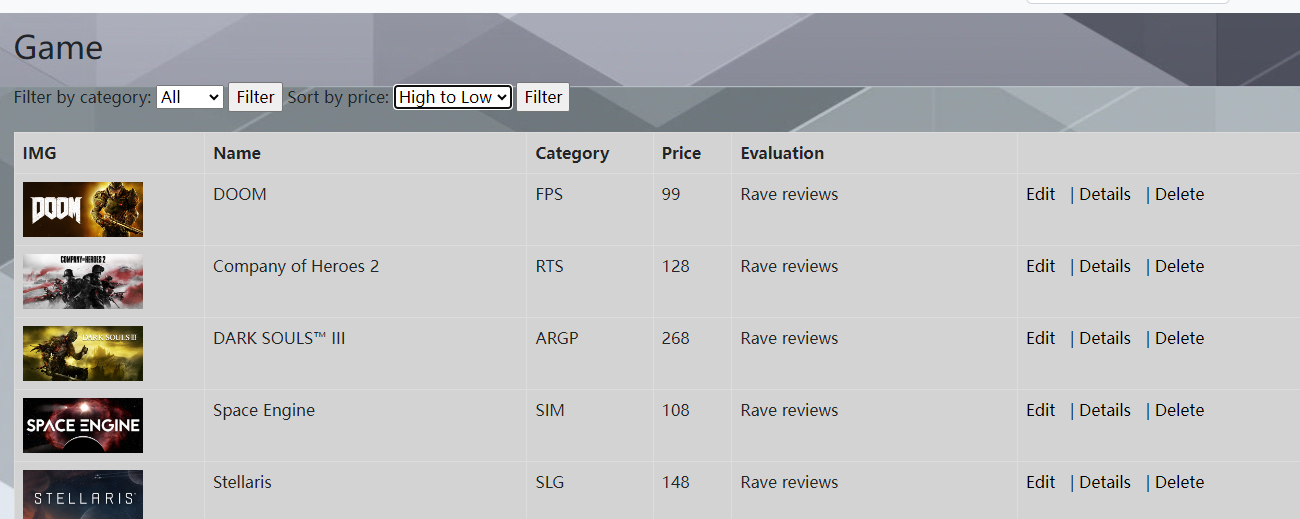
The main page layout is just like these two.

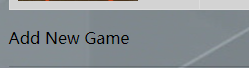
1. **Utilizing Website Features**:
2. Search



Click Search.



1. Filter
2. Pricesort
3. Add new game

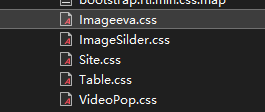


1. **Technical Demonstrations**:
2. Models



The GameCategory and Game models define data structures for game categories and individual games, establishing relationships between them in an ASP.NET MVC application.

1. CSS

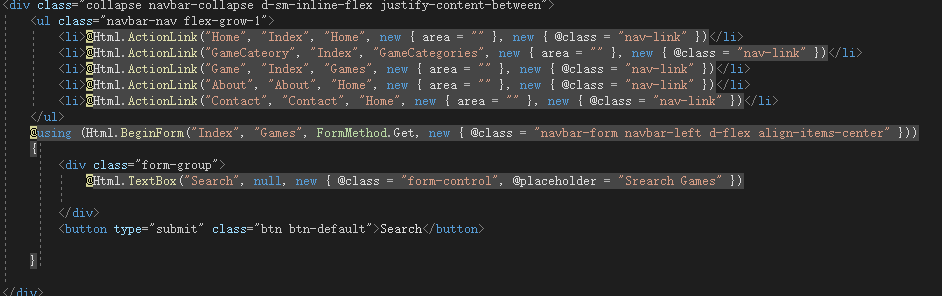
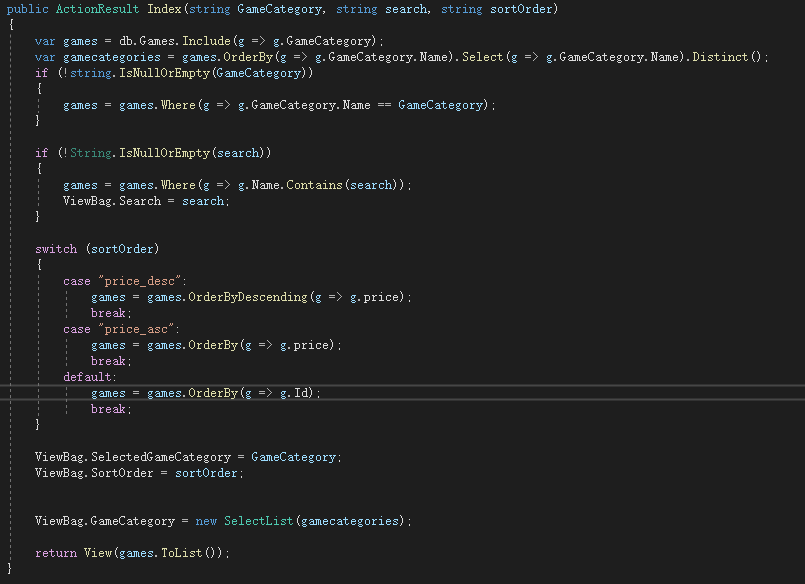


I wrote some of my own CSS styles for the pictures, pop-ups, and views of the game model.

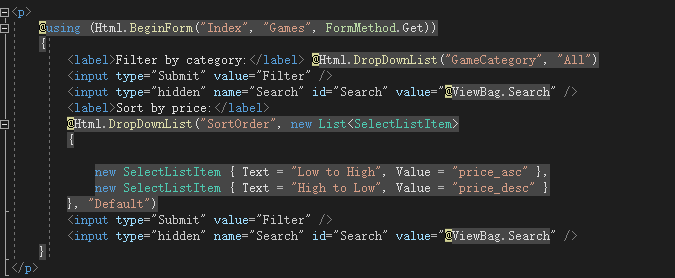
1. JavaScript



I wrote the script for my feature click on the image video pop-up and the image slide.

1. Feature Implementation

1. Search Functionality:use Html.BeginForm to create a form that submits to the Index action of the Games controller. Specifically, use a text box @Html.TextBox("Search", null, new { @class = "form-control", @placeholder = "Search Games" }). The value of this text box is passed as the search parameter to the Index action method. In the Index method, if the search parameter is not empty, LINQ queries are used to filter games that contain the search keyword.



2. Sorting Functionality: use a dropdown list @Html.DropDownList("SortOrder", ...) for selecting the sort order. In the Index method, determine the user's selected sorting order based on the sortOrder parameter and accordingly sort the games. If the user chooses to sort by ascending or descending price, LINQ queries are used for sorting.