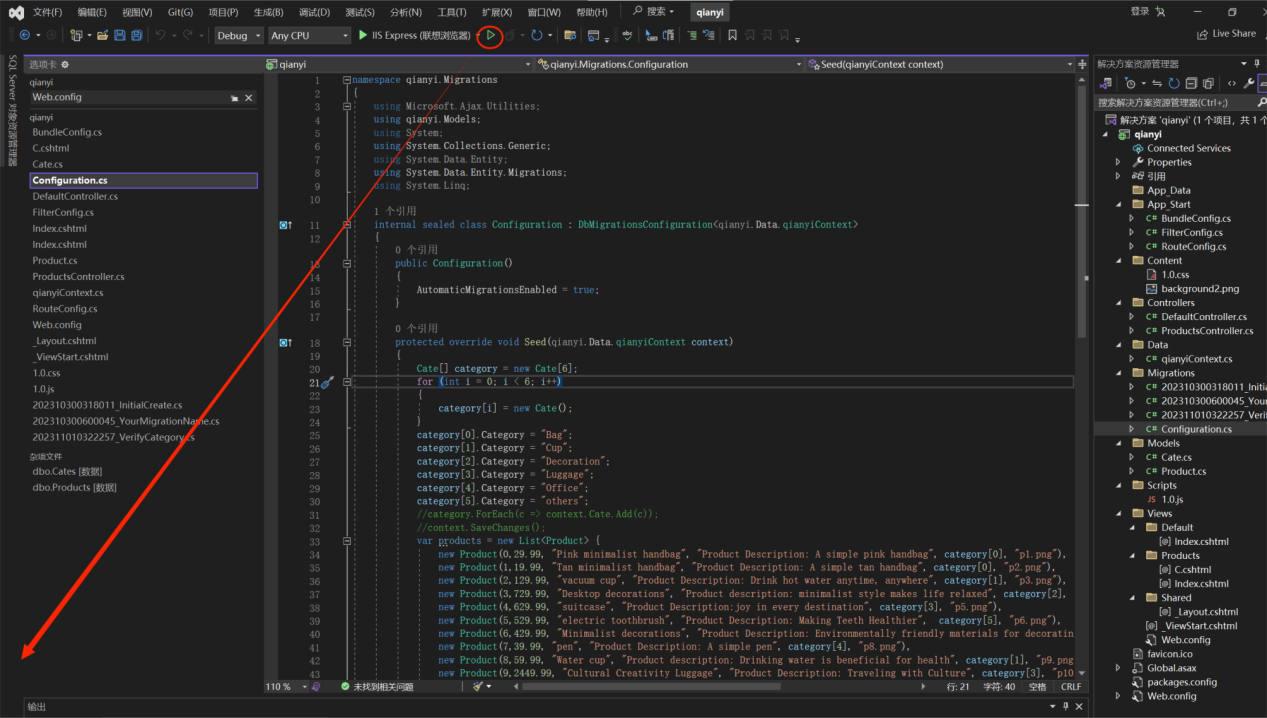
The link to my video：https://www.bilibili.com/video/BV1zH4y1z7h2/?spm\_id\_from=333.999.0.0&vd\_source=16d60c23829eb93a490c2b9a7b647407

Assignment 2 Report

————22009311 Guanghui Pang



1.Open this file.

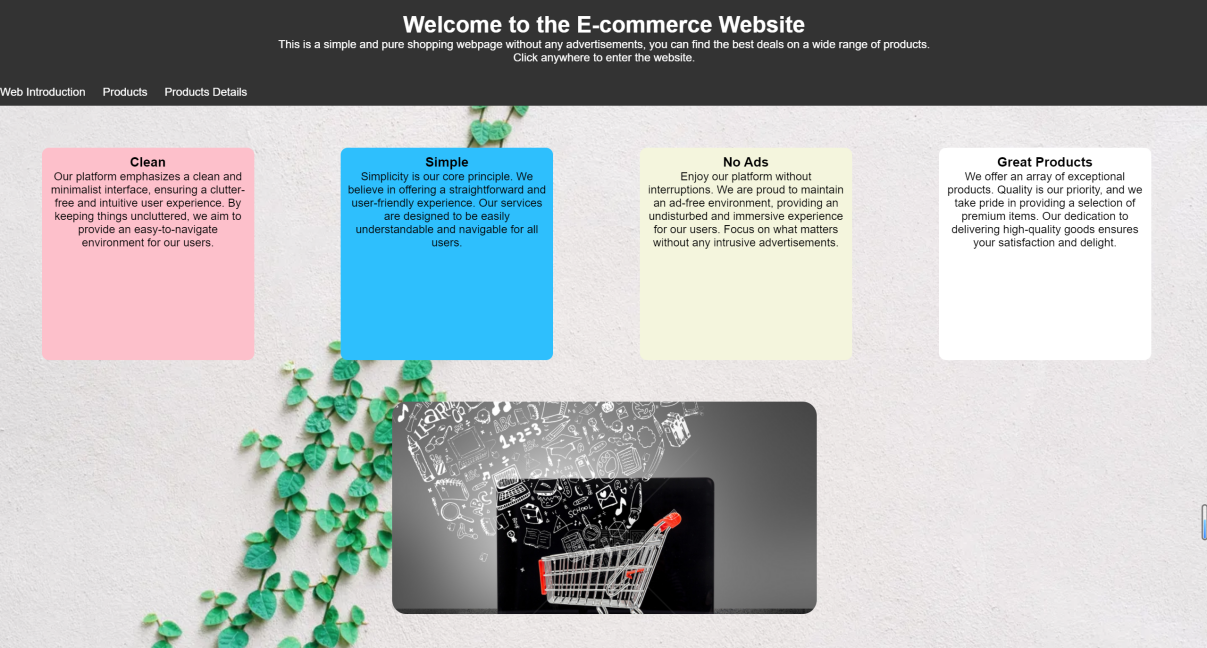


2.Run it in your browser

**2.Website Layout**:

**This is a minimalist shopping website aimed at providing people with a clean shopping environment and high-quality products. It has three parts：Introduction，Products and Products Details.**

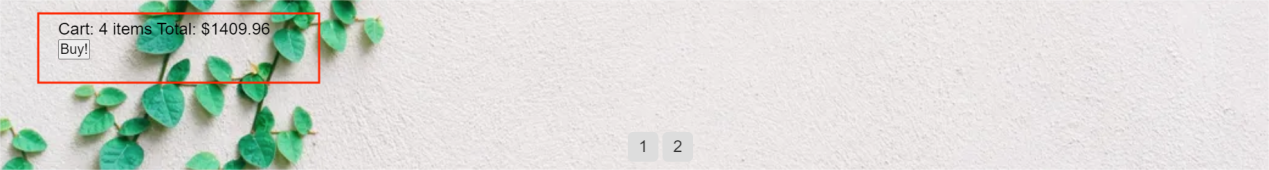
1.**introduction page**



This is the introduction page, which briefly introduces the characteristics of the website: clean and simple. Its navigation bar has three options and there is a product slide at the bottom of the website indicating that this is a product website.

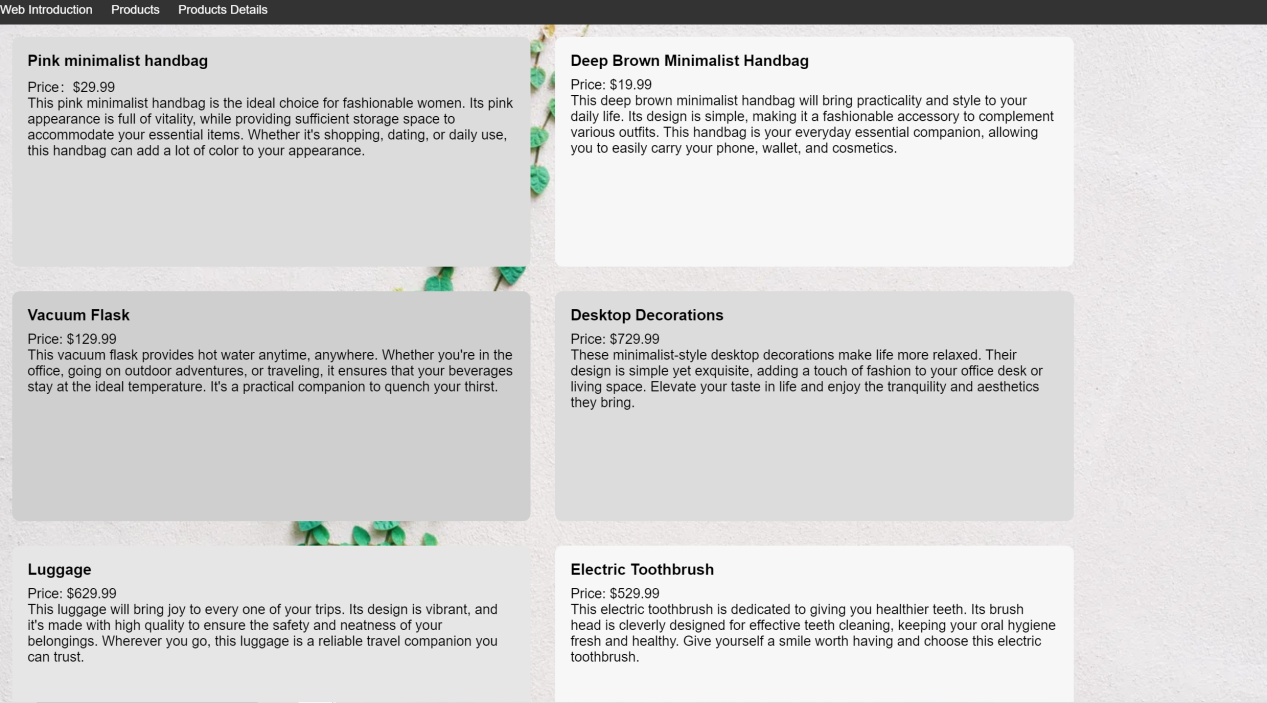
2.**Products Page（main page）**





This is the main page of the website, which has searching，sorting and paging functions. In addition, to meet the needs of the website, I have added a shopping function that can display the number of products added to the shopping cart and the total price.The information of the product is gradually displayed when the mouse moves over the product image

3.**Products Details**



This page displays a detailed introduction to the product, with the text box using the golden ratio and the background box using soft colors to make the introduction comfortable to read.

**3.Utilizing Website Features：**

The initial page is an introduction page, where you can enter the main page by clicking anywhere on the page. You can also enter other pages through the navigation bar on the main page.

On the product page：

1.Search by entering certain words in the product name or description and then clicking the magnifying glass.

2.Sorting by selecting two options in the sort bar and then clicking sort.

3.Paging by click number1 or number2 which are placed on the bottom of the page.

4.Each product has an add to cart option placed to place favorite items in the shopping cart and display them at the bottom.

There is also a navigation bar on the Products Details Page to access other pages.

1. **Technical Demonstrations:**

**1.Models(important):**

Model1：

namespace qianyi.Models

{

public class Product

{

[Key]

public int Id { get; set; } // Assuming an Id field for database primary key

public double Price { get; set; }

public string Name { get; set; }

public string Description { get; set; }

public virtual Cate Category { get; set; }

public string ImageUrl { get; set; }

public Product()

{

this.Id = 0;

this.Name = "默认产品名称";

this.Price = 0;

this.Description = "默认描述";

this.Category = null;

this.ImageUrl = "默认图片地址";

}

public Product(int id,double price,string name, string description, Cate category, string imageUrl)

{

this.Id = id;

this.Name = name;

this.Price = price;

this.Description = description;

this.Category = category;

this.ImageUrl = imageUrl;

}}}

In this model, I created 6 attributes of a product, where Id is used to distinguish products.Price, name, and description are its basic attributes, ImageUrl stores the address of the product image, and Cate is another class used to store its category.

I created a custom constructor and default constructor for it to create objects in the seed.

Model2：

namespace qianyi.Models

{ public class Cate

{

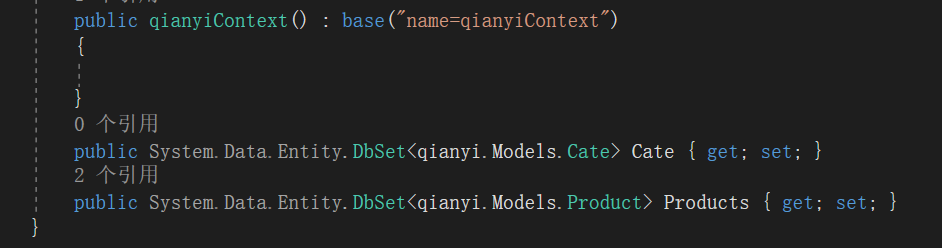
[Key]

public string Category { get; set; }

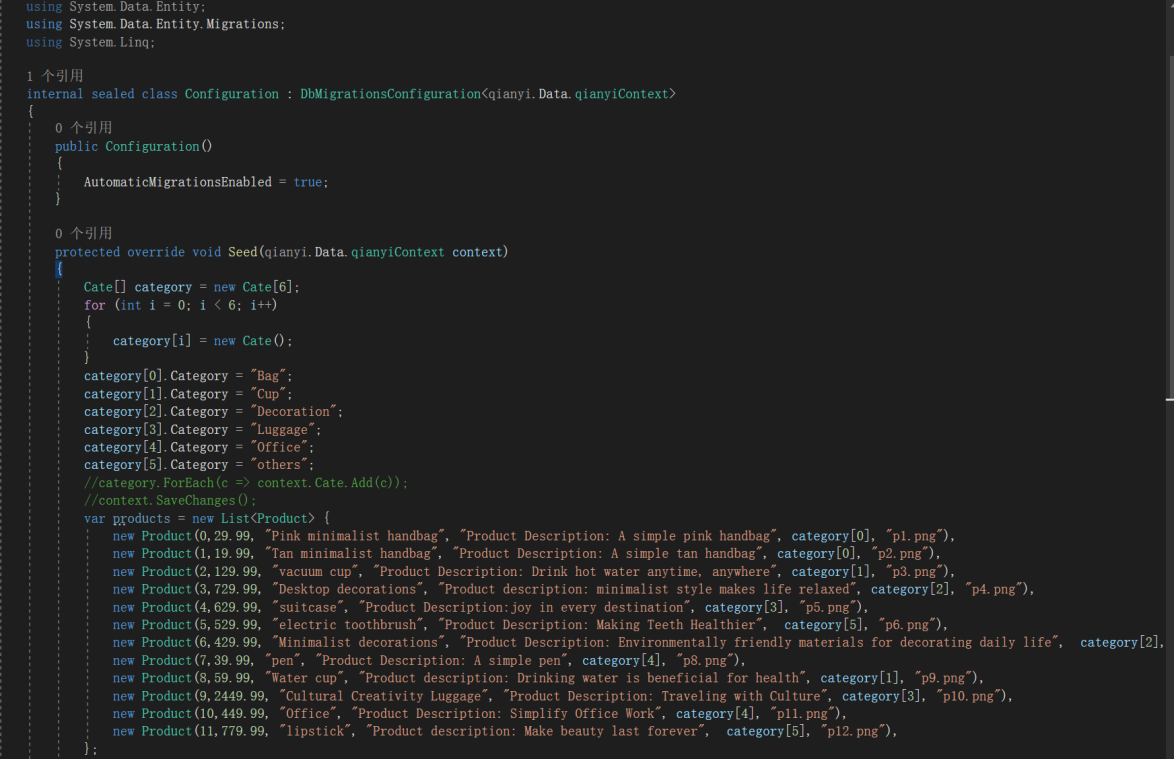
}

}

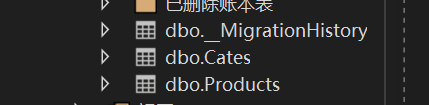
The Cate class is used to store categories in the Product class



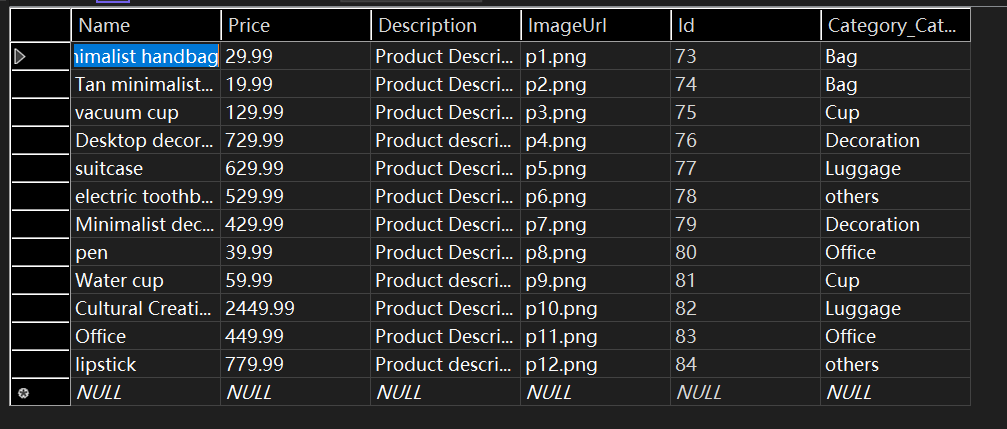
Configure them in the database



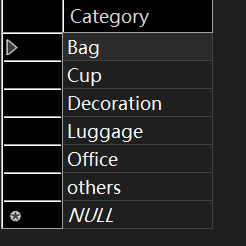
I have a seed method in Configuration to memory the basic data in my database and there is some pictures to show my database



The first one:

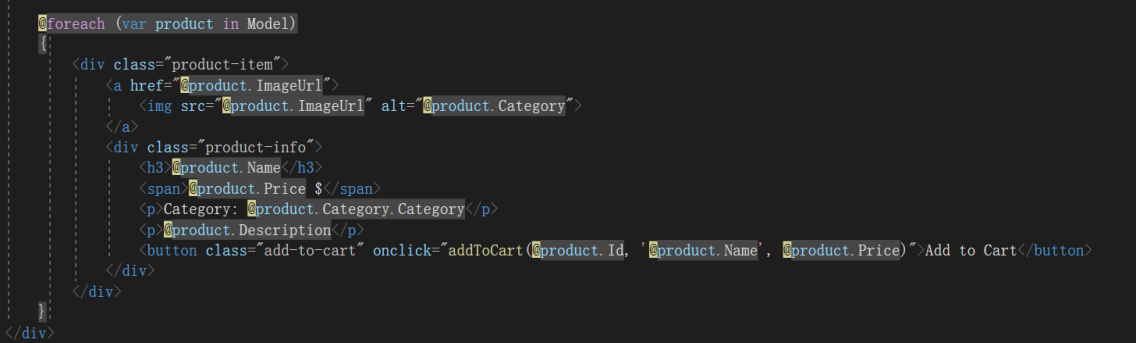
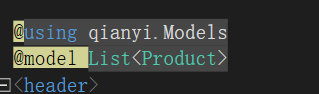


The second one:



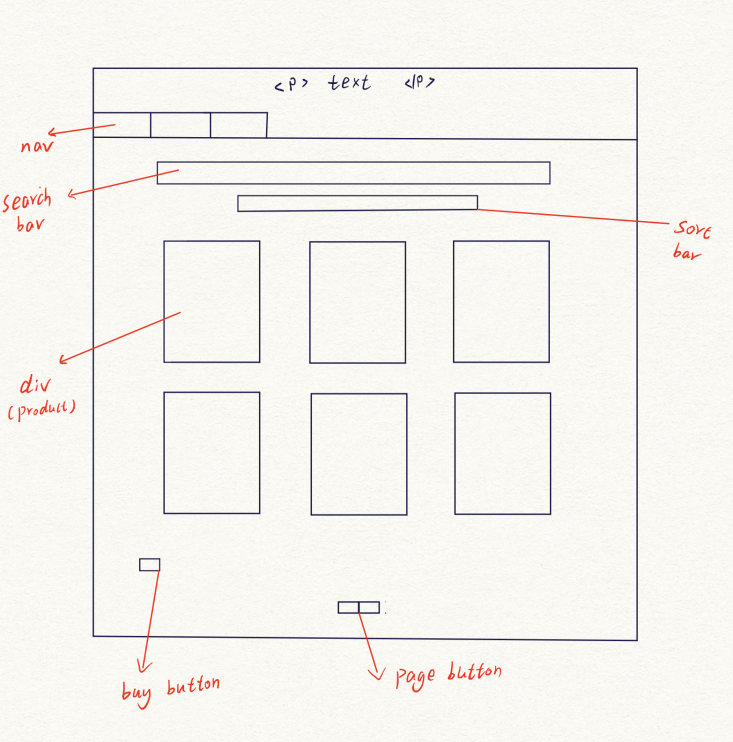
So i can use my database by @ in my html page.

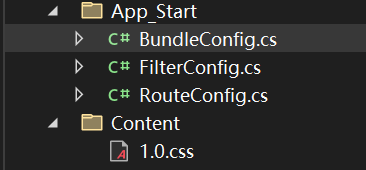
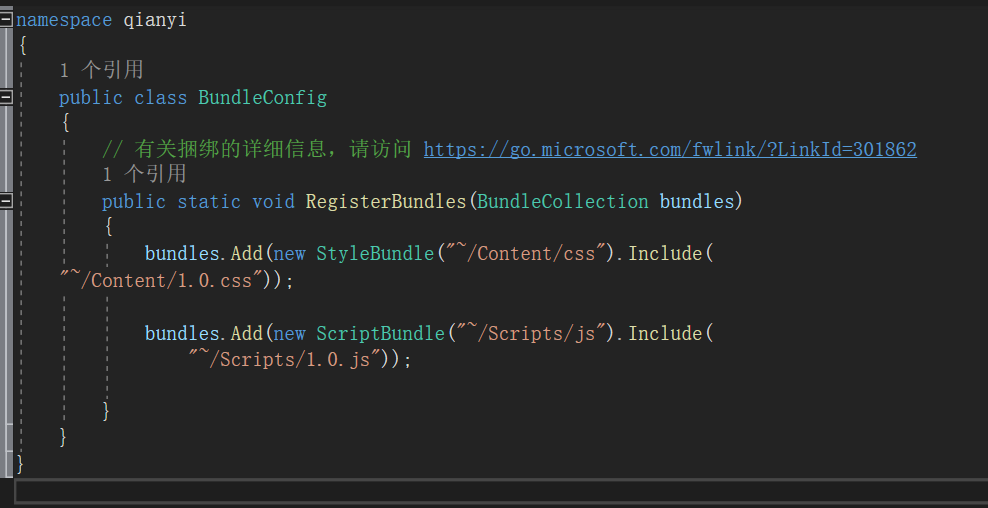




1. **CSS:**

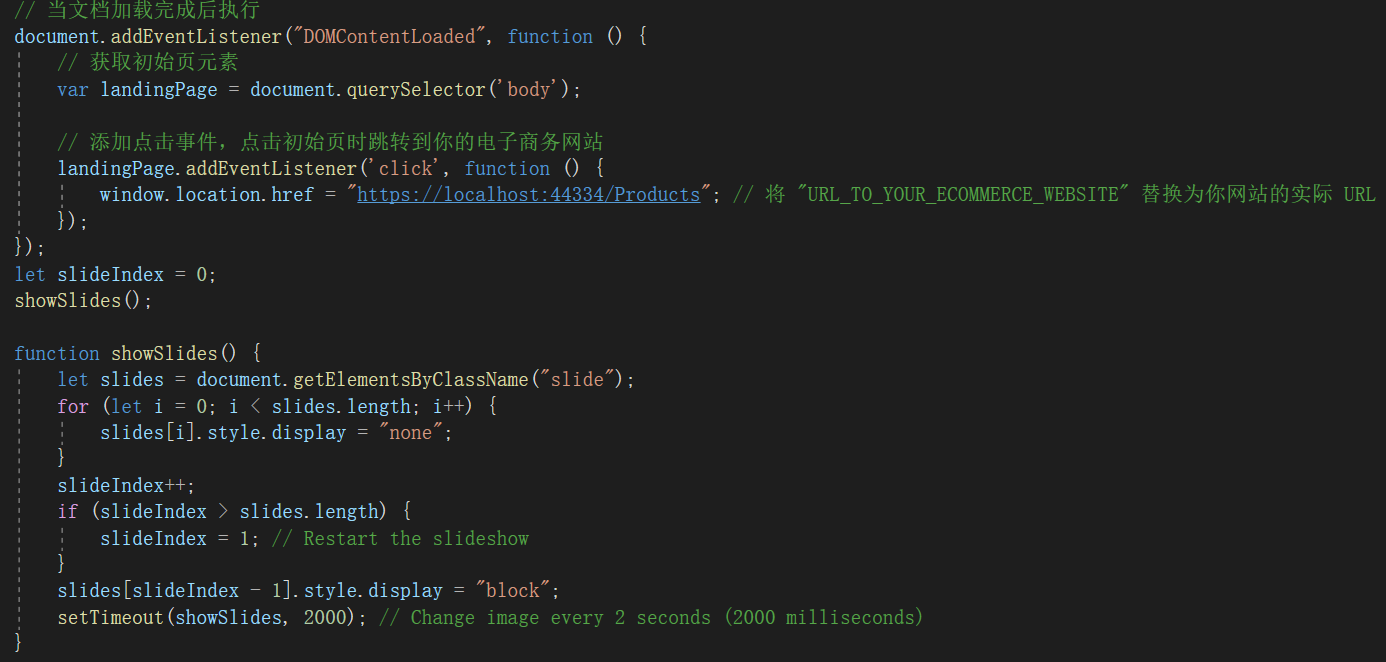
Because the CSS file is too long, I drew a brief image to represent my Product page layout

****

Connecting css with html by BundleConfig so style could be separated from the html.

1. **JavaScript:**



The first EventListener is to enable the introduction page to enter the product page when clicked；

ShowSlides is to achieve the slides would be automatically showed.



This code creates a shopping cart and displays the quantity and total price of the products in the cart

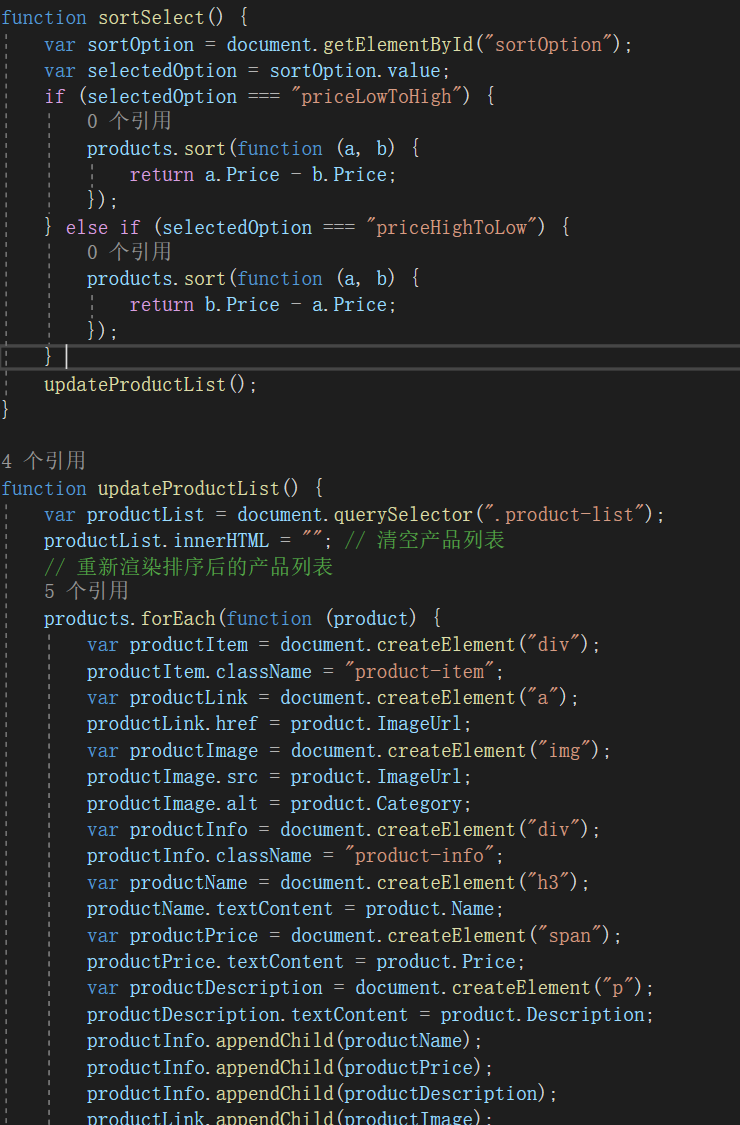
1. **Feature Implementation:**

For searching：



I use document. querySelector (". product list") to retrieve the product list in the HTML, clear it, and then convert user input to lowercase. Through products. filter (function (product)) find the value I need, in this function, I return a value represented by | | to achieve the search function. Then, I fill the filtereProducts back into the empty container, so that my page only displays the search content

For Sorting:



Next, I will use annotation to explain my code. The black one is the annotation, and the red one is the code:

//Get Selection Box Elements

Var sortOption=document. getElementById ("sortOption");

//Get the value of the selection box

Var selectedOption=sortOption. value;

//Sort the product array

If (selectedOption==="priceLowToHigh"){

//Sort by price from low to high

Products. sort (function (a, b){

Return a.Price - b. Price;

});

}Else if (selectedOption=="priceHighToLow"){

//Sort by price from high to low

Products. sort (function (a, b){

Return b. Price - a. Price;

});

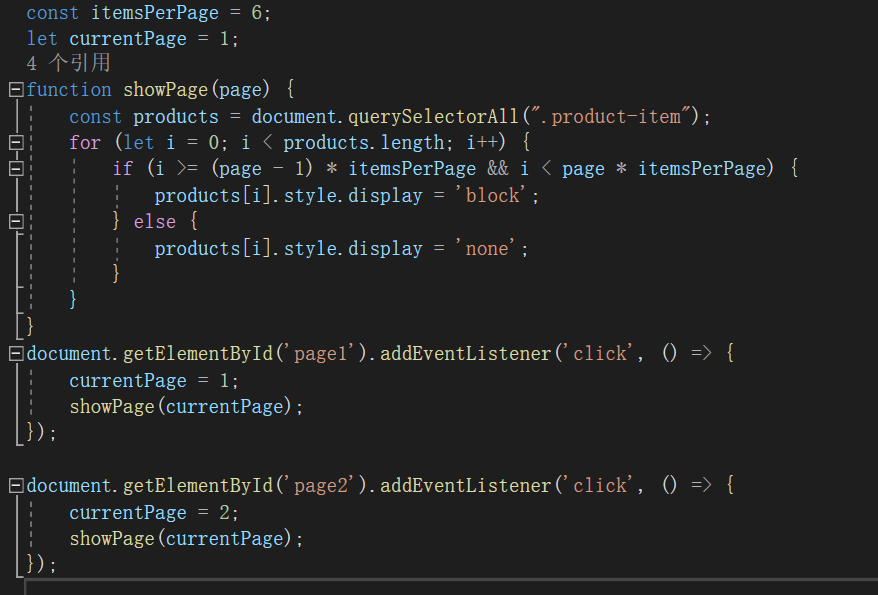
}

//Update the product list on the page

UpdateProductList();

UpdateProductList() here; The function is similar to the search’update, except that after sorting, an array that meets the requirements will appear, and then this array will be filled into my HTML container

For Paging:



If the index of an element is within the range of the current page

(page -1) \* itemsPerPage<=i<page \* itemsPerPage), set the display style of the element to 'block' to display it. The code then listens for two buttons. When the button is clicked, set the currentPage variable to 1 and 2 respectively, and call the showPage (currentPage) function to display the corresponding product items based on the current number of pages.

This way, I can divide my twelve products into two groups for display