@\* Kategorileri eklemek için SelectList kullanıldı. \*@

<**select** **name**="category" class="form-select"

**asp-items**="@(new SelectList(Model.Categories, "CategoryId", "Name","Model.SelectedCategory"))">

<**option** **value**="0">Hepsi</**option**>

</**select**>

</div>

<div class="col-md-11">

<button class="btn btn-primary">

Filtrele

</button>

@\* İlgili Product sayfasında yer alan Display(Name) özelliği kullanılarak tablo başlıkları doğrudan buraya geliyor \*@

<th>@Html.DisplayNameFor(m => m.Products[0].ProductId)</th>

<th>@Html.DisplayNameFor(m => m.Products[0].Name)</th>

<th>@Html.DisplayNameFor(m => m.Products[0].Price)</th>

// Tüm ürünleri ve kategorileri tek yerden yönetmek için bir class oluşturduk

public class ProductViewModel

{

public List<Product>? Products { get; set; }

public List<Category>? Categories { get; set; }

public int SelectedCategory { get; set; }

}

// category null veya boş değilse eğer bu koşul çalışıyor

// category 0 değilse eğer bu koşul çalışıyor

// category değerine göre products listesi filtreleniyor

if (!String.IsNullOrEmpty(category) && category != "0")

{

products = products.Where(p => p.CategoryId == int.Parse(category)).ToList();

}

// bundan sonra tarayıcıda linkin sonuna ?searchString=iphone yazarak arama yapabiliriz

// model adında bir değişken oluşturuyoruz

// bu değişkenin içerisine products ve categories listesini atıyoruz

// ve bu değişkeni view sayfasına gönderiyoruz

var model = new ProductViewModel

{

Products = products,

Categories = Repository.Categories,

SelectedCategory = int.Parse(category)

};

return View(model);