Лабораторная работа №5

Передача данных представлению.

1. **Цель работы.**

Дальнейшее изучение механизма передачи данных представлению.

2. **Общие сведения.**

3. **Выполнение работы**

3.1. Исходные данные

Используйте проект из лабораторной работы №4.

3.2. Задание №1

Выберите любую предметную область. Для одной сущности из выбранной предметной области создайте В папке Entities проекта создайте класс, содержащий следующие свойства:

- ID – уникальный номер;
- Название – короткое название конкретного объекта;
- Описание – дополнительное описание конкретного объекта;
- Категория – свойство для объединения объектов в группы;
- Цена/Вес/Расстояние – выберите любой параметр, который можно в дальнейшем обработать математически, например, просуммировать;
- Изображение – имя файла изображения объекта
- Mime тип изображения

В той же папке создайте класс, описывающий категорию объекта. Отношение должно быть один-ко-многим: одна категория описывает много объектов.

3.3. Задание №2

Создайте контроллер с именем Product. Данный контроллер будет отвечать за отображение информации об объектах из задания №1.

Примечание:

Вы можете выбрать другое имя для контроллера, чтобы название соответствовало имени класса объекта из задания 1. В этом случае также измените название контроллера в компоненте меню (см. лабораторную работу №3)

Представление Index контроллера должно выводить список объектов.

3.3.1. Рекомендации к заданию №2

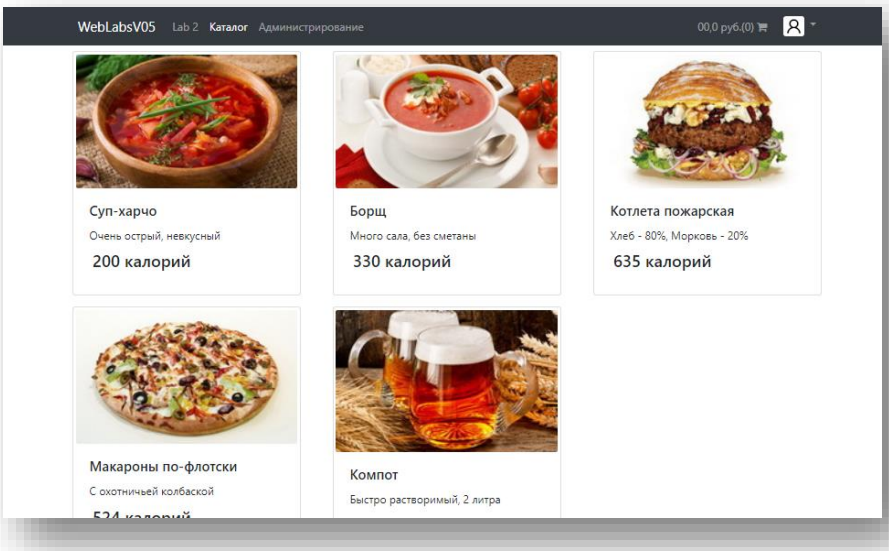
В классе контроллера создайте список типа List<> и проинициализируйте его 4-5 объектами. Этот список используйте в качестве модели представления Index.

3.4. Задание №3

Оформите список с помощью класса bootstrap «card» (см. https://getbootstrap.com/docs/4.3/components/card/#using-grid-markup):

- информация об одном объекте должна занимать 1/3 контейнера экрана
- в верхней части расположить изображение объекта
- название объекта оформить стилем «card-title»
- текст описания объекта оформить стилем «card-text»
- числовую характеристику оформите стилем «card-subtitle badge»

Пример оформления списка:

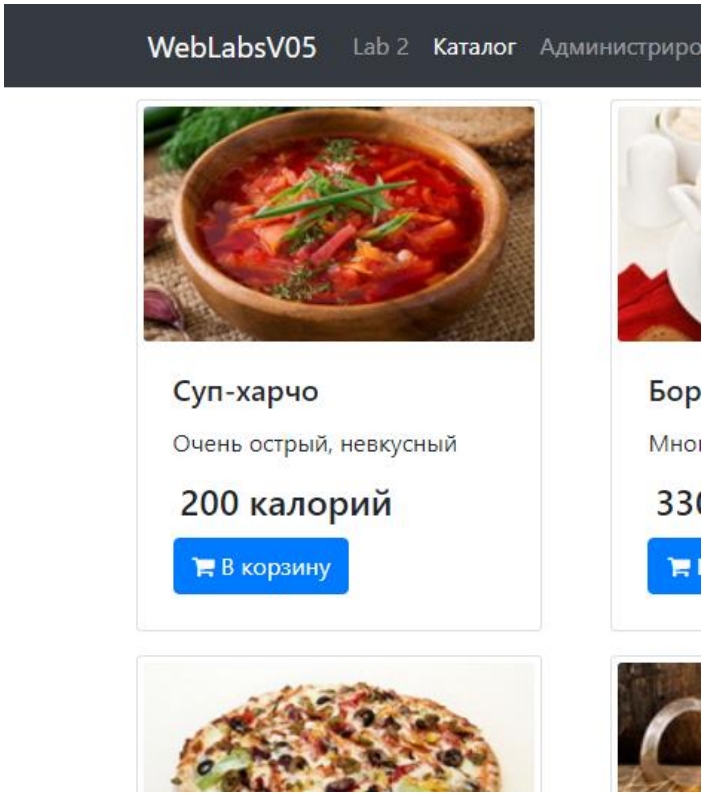


3.5. Задание №4

Для каждого объекта списка добавьте ссылку (тэг <a>) «В корзину».

- ссылка должна адресовать к методу Add контроллера Cart (будут созданы в дальнейшем).
- ссылка должна передавать id добавляемого элемента и адрес текущей страницы для возврата (можно использовать имя returnurl)
- ссылку оформить стилем «btn btn-primary»
- рядом с текстом «В корзину» должна быть иконка sopping-cart из библиотеки font-awesome (см. https://fontawesome.com/icons/shopping-cart?style=solid)

Пример оформления:



3.5.1. Рекомендации к заданию №4

Адрес текущей страницы можно получить так:

```
@{// Получение текущего адреса
  var request = ViewContext.HttpContext.Request;
  var returnUrl = request.Path + request.QueryString.ToUriComponent();
}
```

3.6. Задание №5

Реализуйте разбиение списка товаров на страницы (по 3 товара на странице). При обращении к нужной странице (например ко 2-й) адрес должен иметь вид /Product/Index?pageNo=2.

3.6.1. Рекомендации к заданию №5

В классе контроллера Product создайте свойство _pageSize – количество объектов на странице.

В метод Index контроллера Product передайте номер текущей страницы.

3.7. Задание №6

В папке Models основного проекта создайте модель представления - класс ListViewModel, - имеющий свойства:

- номер текущей страницы;
- общее количество страниц;
- список объектов для отображения на текущей странице.

Класс должен сам делать выборку объектов из общего списка для текущей страницы.

Для создания класса удобно воспользоваться шаблоном проектирования «декоратор»: унаследуйте ваш класс от класса List<T>, добавив функционал для выборки данных и хранения информации о разбиении на страницы.

Для создания объекта класса ListViewModel опишите ***статический*** фабричный метод GetModel, который принимает исходный список объектов, номер текущей страницы и количество объектов на странице. Метод должен вернуть объект класса ListViewModel с уже сформированным списком объектов и значениями номера текущей страницы и общего количества страниц.

3.8. Задание №7

Используйте класс ListViewModel в действии Index контроллера Product.

Измените соответственно модель в представлении Index.

3.9. Задание №8

Оформите разметку одного элемента списка в виде частичного представления

3.10. Задание №9

Добавьте на страницу index возможность выбора категории объектов. При выборе категории в метод Index должен передаваться id выбранной категории.

Выбор категории разместить на странице Index слева от списка объектов.

3.10.1. Рекомендации к заданию №9

Поскольку в метод Index должны передаваться два параметра (id группы и номер страницы), измените тест контроллера Product для передачи этих параметров.

Список категорий представляет собой набор ссылок (тэг <a>) на метод Index контроллера Product с передачей параметра «group» - id выбранной группы.

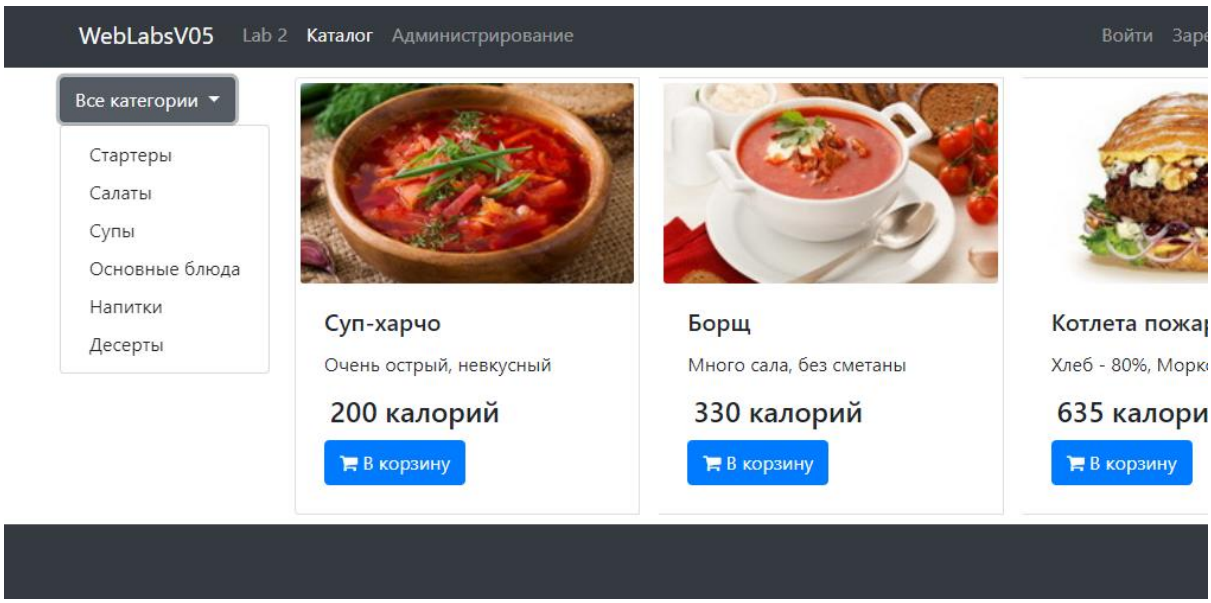
Список категорий можно оформить классом bootstrap «dropdown» (см. https://getbootstrap.com/docs/4.3/components/dropdowns/)

Список объектов категорий можно передать через ViewData. Также для выделения текущего выбора в представление нужно передать id текущей выбранной группы или «0», если выбраны все группы

id выбранной группы можно получить из запроса:

```
HttpContext.Request.Query["group"]
```

Для размещения выбора группы разбейте страницу на две колонки, например, «col-2» и «col-10».



4. Контрольные вопросы

5. Пример выполнения работы

5.1. Предварительная информация

ВНИМАНИЕ:

Проект, используемый в качестве примера, имеет название **WebLabs_BSUIR_V01**. Следовательно, все пространства имен в примерах начинаются с WebLabs_BSUIR_V01, например, WebLabs_BSUIR_V01.Data.

5.2. Создание классов предметной области

ВНИМАНИЕ:

В предлагаемом примере используется предметная область – предприятит питания. Сущность предметной области – Dish (блюдо).

Для выполнения задания добавьте в папку «wwwroot/images» несколько изображений объектов вашей предметной области. Для удобства отображения желательно, чтобы изображения были одного размера или, хотя бы, имели одинаковое соотношение сторон.

Классы предметной области поместите в папку Entities проекта.

Пример класса Dish:

```
public class Dish
    {
        public int DishId { get; set; } // id блюда
        public string DishName { get; set; } // название блюда
        public string Description { get; set; } // описание блюда
        public int Calories { get; set; } // кол. калорий на порцию
        public string Image { get; set; } // имя файла изображения

        // Навигационные свойства
       /// <summary>
       /// группа блюд (например, супы, напитки и т.д.)
       /// </summary>
        public int DishGroupId { get; set; }
        public DishGroup Group { get; set; }
}
```

Пример класса DishGroup:

```
public class DishGroup
    {
        public int DishGroupId { get; set; }
        public string GroupName { get; set; }
        /// <summary>
        /// Навигационное свойство 1-ко-многим
        /// </summary>
        public List<Dish> Dishes { get; set; }
}
```

5.3. Вывод списка объектов на страницу

5.3.1. Создание контроллера

В папке Controllers основного проекта создайте пустой контроллер Product (имя класса контроллера – ProductController).

Опишите приватные поля, содержащие список объектов и список групп объектов, например:

```
List<Dish> _dishes;
List<DishGroup> _dishGroups;
```

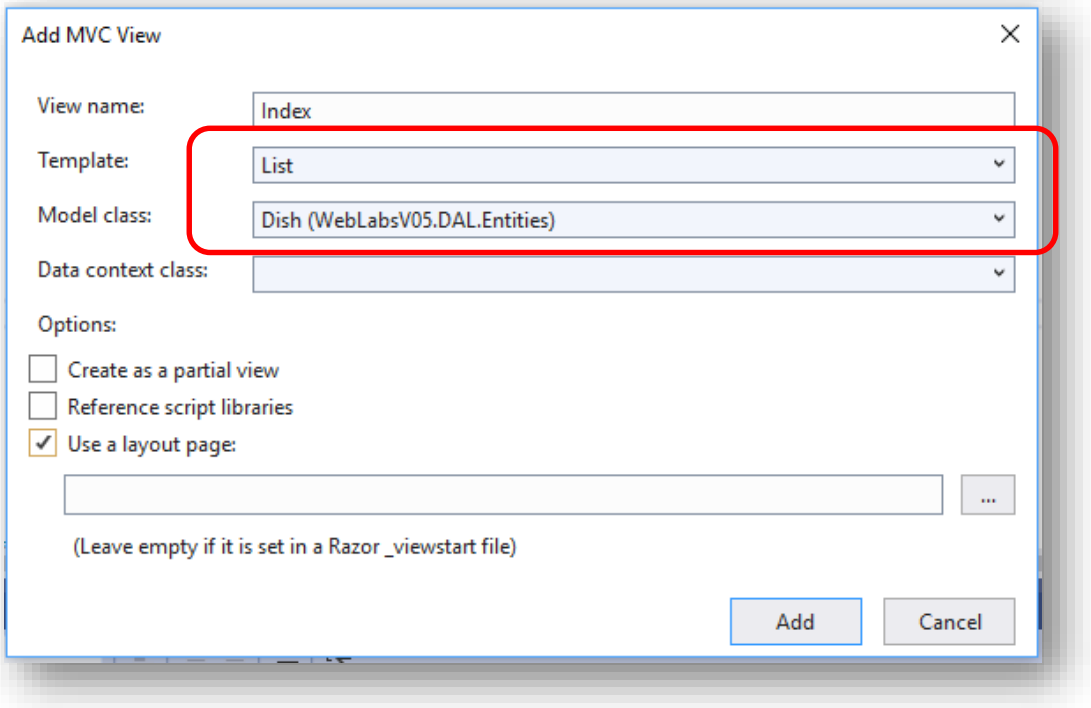
В отдельном методе выполните заполнение списков. Количество элементов в списке объектов должно быть 4-5. Количество элементов в списке групп значения не имеет. Вызовите метод в конструкторе контроллера.

Передайте список объектов в качестве модели представлению:

```
namespace WebLabsV05.Controllers
{
    public class ProductController : Controller
    {
        List<Dish> _dishes;
        List<DishGroup> _dishGroups;

        public ProductController()
        {
            SetupData();
        }

        public IActionResult Index()
        {
            return View(_dishes);
        }

        /// <summary>
        /// Инициализация списков
        /// </summary>
        private void SetupData()
        {
            _dishGroups = new List<DishGroup>
            {
                new DishGroup {DishGroupId=1, GroupName="Стартеры"},
                new DishGroup {DishGroupId=2, GroupName="Салаты"},
                new DishGroup {DishGroupId=3, GroupName="Супы"},
                new DishGroup {DishGroupId=4, GroupName="Основные
блюда"},
                new DishGroup {DishGroupId=5, GroupName="Напитки"},
                new DishGroup {DishGroupId=6, GroupName="Десерты"}
            };

            _dishes = new List<Dish>
            {
                new Dish {DishId = 1, DishName="Суп-харчо",
                Description="Очень острый, невкусный",
                Calories =200, DishGroupId=3, Image="Суп.jpg" },
                new Dish { DishId = 2, DishName="Борщ",
                Description="Много сала, без сметаны",
                Calories =330, DishGroupId=3, Image="Борщ.jpg" },
                new Dish { DishId = 3, DishName="Котлета пожарская",
                Description="Хлеб - 80%, Морковь - 20%",
                Calories =635, DishGroupId=4, Image="Котлета.jpg" },
                new Dish { DishId = 4, DishName="Макароны по-флотски",
                Description="С охотничьей колбаской",
                Calories =524, DishGroupId=4, Image="Макароны.jpg" },
                new Dish { DishId = 5, DishName="Компот",
                Description="Быстро растворимый, 2 литра",
                Calories =180, DishGroupId=5, Image="Компот.jpg" }
            };
```

```
        }
    }
}
```

5.3.2. Создание представления с помощью scaffolding



В представлении описана модель:

```
@model IEnumerable<XXX.Entities.Dish>
```

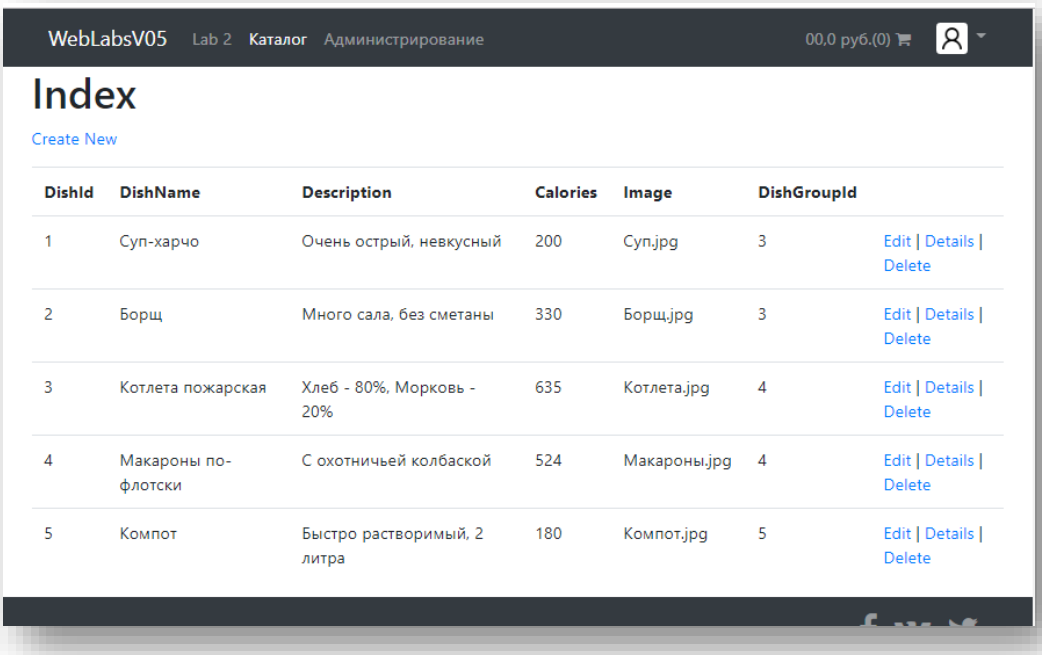
В файл _ViewImports добавьте:

```
@using XXX.Entities
```

Измените модель в представлении Index:

```
@model IEnumerable<Dish>
```

Запустите проект. Выберите пункт меню «Каталог». Убедитесь, что выводится список объектов:



5.3.3. Оформление представления Index

Уберите всю разметку из представления Index, кроме:

```
@model IEnumerable<Dish>

@{
    ViewData["Title"] = "Index";
}
```

Измените значение Title на более понятное название, например:

```
@{
    ViewData["Title"] = "Меню";
}
```

Используйте компонент card библиотеки bootstrap (https://getbootstrap.com/docs/4.3/components/card/):

```
@model IEnumerable<Dish>

@{
    ViewData["Title"] = "Меню";
}

<div class="row">
    <div class="card-deck">
        @foreach (var item in Model)
        {
```

```
            <div class='card m-2 p-1 text-center col-4'>
    <img src="~/Images/@item.Image"
         alt='NoImage'
         class="card-img-top" />
    <div class="card-body">
        <h5 class="card-title">
            @item.DishName
        </h5>
        <p class="card-text">
            @item.Description
        </p>
        <div class="card-text badge badge-secondary">
            <h6>@item.Calories калорий</h6>
        </div>
        @{ // Получение текущего адреса
            var request = ViewContext.HttpContext.Request;
            var returnUrl = request.Path +
request.QueryString.ToUriComponent();
        }
        <!--Разметка кнопки добавления в корзину-->
        <p class="mt-2">
            <a asp-action="Add"
               asp-controller="Cart"
               asp-route-id="@item.DishId"
               asp-route-returnUrl="@returnUrl"
               class="btn btn-primary">
                <i class="fa fa-shopping-cart"></i> В корзину
            </a>
        </p>
    </div>
</div>
        }
    </div>
</div>
```

Запустите проект и проверьте результат.

5.1. Вывод по три объекта на страницу

5.1.1. Изменения в классе контроллера

```
public class ProductController : Controller
    {
        List<Dish> _dishes;
        List<DishGroup> _dishGroups;

        int _pageSize;

        public ProductController()
```

```
        {
            _pageSize = 3;
            SetupData();
        }

        public IActionResult Index(int pageNo=1)
        {
            var items = _dishes
                        .Skip((pageNo - 1)*_pageSize)
                        .Take(_pageSize)
                        .ToList();
            return View(items);
        }
    . . .
}
```

Запустите проект. На страницу должны выводиться 3 первых объекта.

Добавьте в строку запроса «?pageNo=2». Запустите проект. Должен выводиться список объектов 2-й страницы.

5.2. Создание модели представления

В паке Models основного проекта добавьте класс ListViewModel.

Унаследуйте класс от класса List<T>

Добавьте свойства CurrentPage (номер текущей страницы) и TotalPages (общее количество страниц).

Добавьте статический метод GetModel. Метод должен принимать список объектов, номер текущей страницы и количество объектов на странице, а возвращать объект ListViewModel<T>, содержащий список из объектов для текущей страницы и значения CurrentPage и TotalPages:

```
public class ListViewModel<T> : List<T> where T:class
{
    public int TotalPages { get; set; }
    public int CurrentPage { get; set; }
    private ListViewModel(IEnumerable<T> items,
                          int total,
                          int current):base(items)
    {
        TotalPages = total;
        CurrentPage = current;
    }
    /// <summary>
    /// Получить модель представления списка объектов
    /// </summary>
```

```
        /// <param name="list">исходный список объектов</param>
        /// <param name="current">номер текущей страницы</param>
        /// <param name="itemsPerPage">кол. объектов на странице</param>
        /// <returns>объект класса ListViewModel</returns>
        public static ListViewModel<T> GetModel(IEnumerable<T> list,
                                        int current, int itemsPerPage)
        {
            var items = list
                            .Skip((current - 1)*itemsPerPage)
                            .Take(itemsPerPage)
                            .ToList();
            var total = (int)Math.Ceiling((double)list.Count() /
        itemsPerPage);
            return new ListViewModel<T>(items, total, current);
        }
}
```

5.2.1. Изменения в классе ProductController

В методе Index используйте класс ListViewModel<Dish>:

```
public IActionResult Index(int pageNo=1)
{
    return View(ListViewModel<Dish>.GetModel(_dishes, pageNo,
_pageSize));
}
```

В представлении Index измените модель:

```
@model ListViewModel<Dish>
```

Запустите проект. Проверьте, что список выводится правильно.

5.3. Оформление разметки списка

5.3.1. Создание частичного представления

В папке Views/Product создайте частичное представление _ListItemPartial.cshtml. В качестве модели укажите класс Dish. Скопируйте разметку карточки (и т.д.). Замените @item на @Model:

```
@model Dish

<div class='card m-2 p-1 text-center col-4'>
    <img src="~/Images/@Model.Image"
         alt='NoImage'
         class="card-img-top" />
    <div class="card-body">
```

```
            <h5 class="card-title">
                @Model.DishName
            </h5>
            <p class="card-text">
                @Model.Description
            </p>
            <div class="card-text badge badge-secondary">
                <h6>@Model.Calories калорий</h6>
            </div>
            @{ // Получение текущего адреса
                var request = ViewContext.HttpContext.Request;
                var returnUrl = request.Path +
request.QueryString.ToUriComponent();
            }
            <!--Разметка кнопки добавления в корзину-->
            <p class="mt-2">
                <a asp-action="Add"
                   asp-controller="Cart"
                   asp-route-id="@Model.DishId"
                   asp-route-returnUrl="@returnUrl"
                   class="btn btn-primary">
                    <i class="fa fa-shopping-cart"></i> В корзину
                </a>
            </p>
        </div>
</div>
```

5.3.2. Изменение представления Index

Вместо разметки элемента списка поместите вызов частичного представления _ListItemPartial:

```
<div class="row">
    <div class="card-deck col">
        @foreach (var item in Model)
        {
            <partial name="_ListItemPartial" model="@item" />
        }
    </div>
</div>
```

Запустите проект, проверьте результат.

5.4. Выбор группы объектов

5.4.1. Разметка меню выбора группы

На страницу Index добавьте разметку:

```
@model ListViewModel<Dish>
```

```
@{
    ViewData["Title"] = "Меню";

    var categories = ViewData["Groups"] as IEnumerable<DishGroup>;
    int currentGroup = (int)ViewData["CurrentGroup"];

    var text = currentGroup !=0
        ? categories
            .FirstOrDefault(g => g.DishGroupId == currentGroup)?
            .GroupName
        : "Все";
}

<div class="row">
    <div class="col-2">
        <div class="dropdown mt-2">
            <a class="btn btn-secondary dropdown-toggle"
               asp-action="Index" asp-controller="Product"
               role="button"
               id="dropdownMenuLink"
               data-toggle="dropdown"
              aria-haspopup="true" aria-expanded="false">
                @text
            </a>

            <div class="dropdown-menu"
                 aria-labelledby="dropdownMenuLink">
                <a class="dropdown-item"
                   asp-action="Index"
                   asp-controller="Product">Все</a>

                @foreach (var item in categories)
                {
                    <a class="dropdown-item"
                       asp-action="Index"
                       asp-controller="Product"
                       asp-route-group="@item.DishGroupId"
                      asp-route-pageNo="1"
                       >@item.GroupName</a>
                }
            </div>
        </div>
    </div>
    <div class="col-10">
        <div class="card-deck">
            @foreach (var item in Model)
            {
                <partial name="_ListItemPartial" model="@item" />
            }
        </div>
```

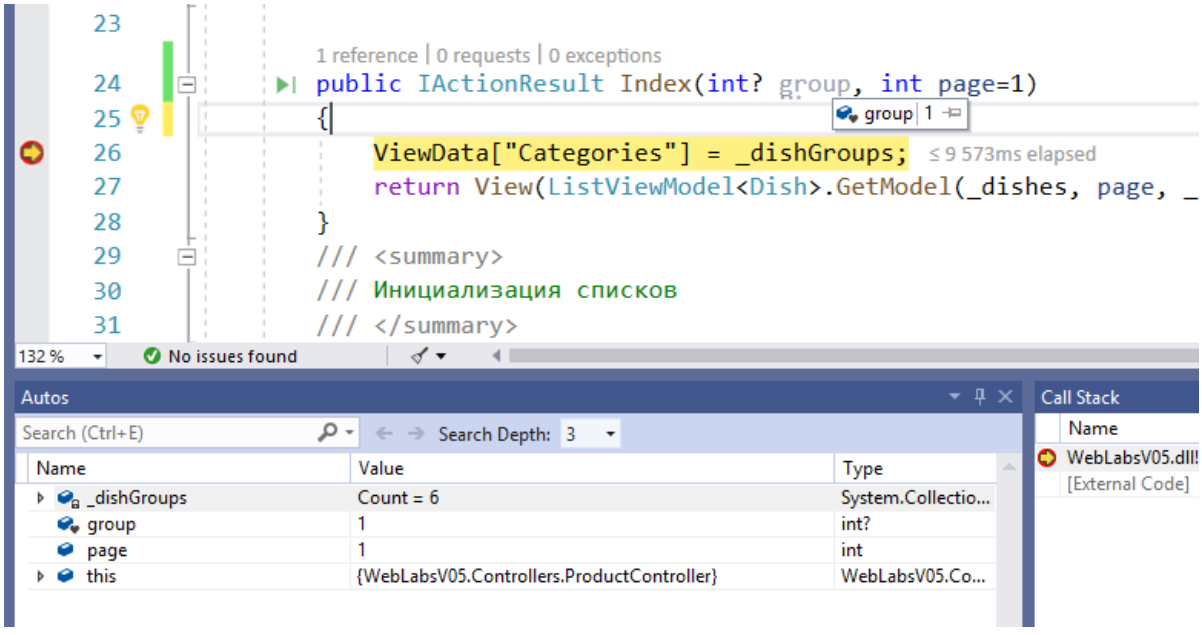
```
    </div>
</div>
```

5.4.2. Изменения в методе Index контроллера Product

Добавьте еще один параметр в метод. Передайте в представление список групп и номер текущей группы.

```
public IActionResult Index(int? group, int pageNo=1)
{
    // Поместить список групп во ViewData
        ViewData["Groups"] = _dishGroups;

    // Получить id текущей группы и поместить в TempData
        ViewData["CurrentGroup"] = group ?? 0;. . .
}
```

Запустите проект в режиме отладки. Сделайте точку останова внутри метода Index. Убедитесь, что параметр «group» передается в контроллер при выборе группы в списке:



Сформируйте список объектов, отфильтрованный по группе и передайте его методу GetModel:

```
public IActionResult Index(int? group, int pageNo=1)
{
    var dishesFiltered = _dishes
    .Where(d => !group.HasValue || d.DishGroupId == group.Value);
```

```
        ViewData["Groups"] = _dishGroups;
          . . .
        return View(ListViewModel<Dish>.GetModel(dishesFiltered,
                                                 pageNo, _pageSize));
}
```

Запустите проект и проверьте, что отображаются объекты из выбранной группы.