МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ

«ГОМЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ ИМЕНИ П. О. СУХОГО»

Факультет автоматизированных и информационных систем

Кафедра «Информатика»

дисциплина «Избранные главы информатики»

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 3

«**Поддержка сохранения состояния. Кэширование**»

Вариант № 16

Выполнил:

студент группы ИП-31, Самусев Р.А.

Принял:

доцент Асенчик О.Д.

Гомель 2018

**Цель работы:** Ознакомиться с возможностями ASP.NET Core с методами сохранению состояния и повышение производительности веб приложений путем использования кэширования.

**Ход работы**

Github: <https://github.com/iorehovski/ChemistShop/tree/master/lab3>

**Задание**:

Используя ранее разработанные объектную модель и простое ASP.NET Core приложение (сайт) в заданной предметной области:

1. Разработать компонент промежуточного уровня (middleware) и включить его в конвейер обработки HTTP-запроса с целью кэширования последней записи из каждой таблицы базы данных заданной предметной области с помощью встроенного инструмента кэширования - объекта IMemoryCache. Данные в кэше хранить неизменными в течении 2\*N+240 секунд, где N- номер вашего варианта.
2. Вывести кэшированные данные таблиц MemoryCache на соответствующие страницы на сайта, генрируемые с использованием представлений (Views).
3. С использованием средств разработчика браузера (Chrome, Firefox) продемонстрировать ускорение обработки запроса при наличии кэширования с использованием MemoryCache.
4. Используя предварительно созданный и сконфигурированный в классе Startup профиль кэширования, подключить кэширование вывода для страниц с использованием атрибута ResponseCache для соответствующих методов контроллера. Данные в кэше хранить неизменными в течении 2\*N+240 секунд, где N- номер вашего варианта.
5. С использованием средств разработчика браузера (Chrome, Firefox) продемонстрировать ускорение обработки запроса при наличии кэширования с использованием атрибута ResponseCache.
6. Реализовать сохранение состояния элементов формы одной страницы с использованием куки.
7. Реализовать сохранение состояния элементов формы одной страницы в виде одного объекта специальной структуры с использованием объекта Session.
8. Осуществить заполнение элементов форм всех страниц при их загрузке данными ранее сохранненными в объекте Session и куки.

**Листинги классов проекта:**

**Startup.cs**

using Microsoft.AspNetCore.Builder;

using Microsoft.AspNetCore.Hosting;

using Microsoft.AspNetCore.Mvc;

using Microsoft.Extensions.Configuration;

using Microsoft.Extensions.DependencyInjection;

using advertisingAgency.Models;

using Microsoft.EntityFrameworkCore;

namespace advertisingAgency

{

public class Startup

{

public Startup(IConfiguration configuration)

{

Configuration = configuration;

}

public IConfiguration Configuration { get; }

public void ConfigureServices(IServiceCollection services)

{

string connection = Configuration.GetConnectionString("SQLConnection");

services.AddDbContext<adAgencyContext>(options => options.UseSqlServer(connection));

services.AddTransient<adAgencyContext>();

services.AddMemoryCache();

services.AddResponseCaching();

services.AddDistributedMemoryCache();

services.AddSession();

services.AddMvc().AddSessionStateTempDataProvider();

services.AddMvc(options =>

{

options.CacheProfiles.Add("Caching",

new CacheProfile()

{

Duration = 2 \* 16 + 240,

Location = ResponseCacheLocation.Any

});

options.CacheProfiles.Add("NoCaching",

new CacheProfile()

{

Location = ResponseCacheLocation.None,

NoStore = true

});

});

}

public void Configure(IApplicationBuilder app, IHostingEnvironment env)

{

app.UseLastElementCache();

app.UseSession();

if (env.IsDevelopment())

{

app.UseBrowserLink();

app.UseDeveloperExceptionPage();

}

else

{

app.UseExceptionHandler("/Home/Error");

}

app.UseStaticFiles();

app.UseResponseCaching();

app.UseMvcWithDefaultRoute();

app.UseMvc(routes =>

{

routes.MapRoute(

name: "default",

template: "{controller=Home}/{action=Index}/{id?}");

});

}

}

}

**Cashelast.cs**(класс для Middleware)

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using Microsoft.AspNetCore.Builder;

using Microsoft.AspNetCore.Http;

using Microsoft.Extensions.Caching.Memory;

using advertisingAgency.Models;

namespace advertisingAgency

{

public class CasheLast

{

private readonly RequestDelegate \_next;

private readonly IMemoryCache \_memoryCache;

public CasheLast(RequestDelegate next, IMemoryCache memCache)

{

\_next = next;

\_memoryCache = memCache;

}

public async Task Invoke(HttpContext context, adAgencyContext dbContext)

{

string path = context.Request.Path.Value.ToLower();

object data = null;

if (path == "/")

data = dbContext.Advedirsments.Last();

if (path == "/home/index")

data = dbContext.Advedirsments.Last();

if (path == "/home/customers")

data = dbContext.Customers.Last();

if (path == "/home/orders")

data = dbContext.Orders.Last();

if (!\_memoryCache.TryGetValue(path, out data))

{

\_memoryCache.Set(path, data, new MemoryCacheEntryOptions().SetAbsoluteExpiration(TimeSpan.FromSeconds(272)));

}

await \_next.Invoke(context);

}

}

public static class DbCacheExtensions

{

public static IApplicationBuilder UseLastElementCache(this IApplicationBuilder builder)

{

return builder.UseMiddleware<CasheLast>();

}

}

}

Класс контроллера

**HomeController.cs**

using System;

using System.Collections.Generic;

using System.Diagnostics;

using System.Linq;

using System.Threading.Tasks;

using Microsoft.AspNetCore.Mvc;

using advertisingAgency.Models;

using Newtonsoft.Json;

using Microsoft.AspNetCore.Http;

using Microsoft.Extensions.Caching.Memory;

namespace advertisingAgency.Controllers

{

public class HomeController : Controller

{

readonly IMemoryCache \_memoryCache;

readonly adAgencyContext \_context;

public HomeController(IMemoryCache memoryCache, adAgencyContext context)

{

\_context = context;

\_memoryCache = memoryCache;

}

[ResponseCache(CacheProfileName = "Caching")]

public IActionResult Index()

{

Advedirsments adFromMemoryCashe = (Advedirsments)\_memoryCache.Get(Request.Path.Value.ToLower());

if (Request.Cookies["coockiesAdvedirsments"] != null)

{

Advedirsments adCoockies = JsonConvert.DeserializeObject<Advedirsments>(Request.Cookies["coockiesAdvedirsments"].ToString());

ViewBag.adCoockies = adCoockies;

}

ViewBag.adFromMemory = adFromMemoryCashe;

ViewBag.advedirsments = \_context.Advedirsments.ToList();

return View();

}

[ResponseCache(CacheProfileName = "Caching")]

public IActionResult Customers()

{

Customers customersFromMemoryCashe = (Customers)\_memoryCache.Get(Request.Path.Value.ToLower());

ViewData["customersFromMemoryCashe"] = customersFromMemoryCashe;

if (HttpContext.Session.Get("CustomersSession") != null)

{

string[] customersSessionArray = HttpContext.Session.GetString("CustomersSession").Split(";");

Customers customerSession = new Customers(customersSessionArray[0], customersSessionArray[1], customersSessionArray[2]);

ViewBag.customersSession = customerSession;

}

ViewBag.customers = \_context.Customers.ToList();

return View();

}

[ResponseCache(CacheProfileName = "Caching")]

public IActionResult Orders()

{

Orders ordersFromMemoryCashe = (Orders)\_memoryCache.Get(Request.Path.Value.ToLower());

if (Request.Cookies["coockiesOrders"] != null)

{

Orders orders = JsonConvert.DeserializeObject<Orders>(Request.Cookies["coockiesOrders"].ToString());

ViewBag.orCoockies = orders;

}

ViewBag.ordFromMemory = ordersFromMemoryCashe;

ViewBag.orders = \_context.Orders.ToList();

return View();

}

[HttpPost]

public IActionResult AddType(Advedirsments advedirsment)

{

CookieOptions cookie = new CookieOptions();

cookie.Expires = DateTime.Now.AddMinutes(1);

if (Request.Cookies["coockiesAdvedirsments"] == null)

{

string value = JsonConvert.SerializeObject(advedirsment);

Response.Cookies.Append("coockiesAdvedirsments", value);

}

\_context.Advedirsments.Add(advedirsment);

\_context.SaveChanges();

return RedirectToAction("Index");

}

[HttpPost]

public IActionResult AddCustomer(Customers customer)

{

\_context.Customers.Add(customer);

\_context.SaveChanges();

string sess = customer.CustomerName + ";" + customer.CustomerAddress + ";" + customer.CustomerTelephone;

HttpContext.Session.SetString("CustomersSession", sess);

return RedirectToAction("Customers");

}

[HttpPost]

public IActionResult AddOrder(Orders order)

{

\_context.Orders.Add(order);

\_context.SaveChanges();

return RedirectToAction("Orders");

}

public IActionResult Error()

{

return View(new ErrorViewModel { RequestId = Activity.Current?.Id ?? HttpContext.TraceIdentifier });

}

}

}

Класс контекста данных

**adAgencyContxt.cs**

using Microsoft.EntityFrameworkCore;

namespace advertisingAgency.Models

{

public class adAgencyContext:DbContext

{

public DbSet<Advedirsments> Advedirsments { get; set; }

public DbSet<Customers> Customers { get; set; }

public DbSet<Orders> Orders { get; set; }

public adAgencyContext(DbContextOptions<adAgencyContext> options)

: base(options)

{

}

}

}

Классы моделей

**Advedirsments.cs**

using System.Collections.Generic;

namespace advertisingAgency.Models

{

public class Advedirsments

{

public Advedirsments() { }

public Advedirsments(string AdType, string AdDescription)

{

this.AdType = AdType;

this.AdDescription = AdDescription;

}

public int AdvedirsmentsId { get; set; }

public string AdType { get; set; }

public string AdDescription { get; set; }

public virtual ICollection<Orders> Orders { get; set; }

}

}

**Customers.cs**

using System.Collections.Generic;

namespace advertisingAgency.Models

{

public class Customers

{

public Customers() { }

public Customers(string CustomerName, string CustomerAddress, string CustomerTelephone)

{

this.CustomerName = CustomerName;

this.CustomerAddress = CustomerAddress;

this.CustomerTelephone = CustomerTelephone;

}

public int CustomersId { get; set; }

public string CustomerName { get; set; }

public string CustomerAddress { get; set; }

public string CustomerTelephone { get; set; }

public virtual ICollection<Orders> Orders { get; set; }

}

}

**Orders.cs**

using System;

using System.Collections.Generic;

using System.Text;

namespace advertisingAgency.Models

{

public class Orders

{

public Orders() { }

public Orders(DateTime DateOrder, DateTime DateBegin, DateTime DateEnd, string Location, int OrderCost)

{

this.DateOrder = DateOrder;

this.DateBegin = DateBegin;

this.DateEnd = DateEnd;

this.Location = Location;

this.OrderCost = OrderCost;

}

public int OrdersId { get; set; }

public DateTime DateOrder { get; set; }

public DateTime DateBegin { get; set; }

public DateTime DateEnd { get; set; }

public string Location { get; set; }

public int OrderCost { get; set; }

public bool PaymentStatus { get; set; }

public int? EmployeeId { get; set; }

public int? CustomersId { get; set; }

public int AdvedirsmentsId { get; set; }

public virtual Advedirsments Advedirsments { get; set; }

public virtual Customers Customers { get; set; }

}

}

Классы представлений

**Index.cshtml**

@model advertisingAgency.Models.Advedirsments

@{

ViewData["Title"] = "Advedirsments";

Layout = "~/Views/Shared/\_Layout.cshtml";

}

<img src="~/images/images.jpg" />

<p></p>

<h1>Виды рекламы</h1>

<table border="1">

<tr>

<th>Тип</th>

<th>Описание</th>

</tr>

@foreach (var item in ViewBag.advedirsments)

{

<tr>

<td>@item.AdType</td>

<td>@item.AdDescription</td>

</tr>

}

</table>

@{

advertisingAgency.Models.Advedirsments adCoockies = (advertisingAgency.Models.Advedirsments)ViewBag.adCoockies;

}

@using (Html.BeginForm("AddType", "Home", FormMethod.Post))

{

<p class="add"> Добавить </p>

<p>

<label>Тип</label>

@{

if (adCoockies != null)

{

@Html.TextBox("AdType", adCoockies.AdType);

}

else

{

@Html.TextBox("AdType");

}

}

</p>

<p>

<label>Описание</label>

@{

if (adCoockies != null)

{

@Html.TextBox("AdDescription", adCoockies.AdDescription);

}

else

{

@Html.TextBox("AdDescription");

}

}

</p>

<p>

<input type="submit" value="Send">

</p>

}

@{

if (ViewBag.adFromMemory != null)

{

advertisingAgency.Models.Advedirsments adFromMemoryCashe = (advertisingAgency.Models.Advedirsments)ViewBag.adFromMemory;

<div class="infoMemory">

@Html.DisplayNameFor(x => x.AdType):&nbsp @adFromMemoryCashe.AdType.ToString();<br />

@Html.DisplayNameFor(x => x.AdDescription):&nbsp @adFromMemoryCashe.AdDescription.ToString();<br />

</div>

}

}

@{

if (ViewBag.adFromMemory != null)

{

advertisingAgency.Models.Advedirsments adFromMemoryCashe = (advertisingAgency.Models.Advedirsments)ViewBag.adFromMemory;

<div class="infoMemory">

<p>MemoryCache</p>

@Html.DisplayNameFor(x => x.AdType):&nbsp @adFromMemoryCashe.AdType.ToString();<br />

@Html.DisplayNameFor(x => x.AdDescription):&nbsp @adFromMemoryCashe.AdDescription.ToString();<br />

</div>

}

}

**Customers.cshtml**

@model advertisingAgency.Models.Customers

@{

ViewData["Title"] = "Customers";

}

<img src="~/images/images.jpg" />

<h1>Заказчики</h1>

<table>

<tr>

<th>Заказчик</th>

<th>Адрес</th>

<th>Телефон</th>

</tr>

@foreach (Customers item in ViewBag.customers)

{

<tr>

<td>@item.CustomerName</td>

<td>@item.CustomerAddress</td>

<td>@item.CustomerTelephone</td>

</tr>

}

</table>

@{

advertisingAgency.Models.Customers customersSession =

(advertisingAgency.Models.Customers)ViewBag.customersSession;

}

@using (Html.BeginForm("AddCustomer", "Home", FormMethod.Post))

{

<p class="add"> Добавить </p>

<p>

<label>Имя заказчика</label>

@{

if (customersSession != null)

{

@Html.TextBox("CustomerName", customersSession.CustomerName)

}

else

{

@Html.TextBox("CustomerName")

}

}

</p>

<p>

<label>Адрес</label>

@{

if (customersSession != null)

{

@Html.TextBox("CustomerAddress", customersSession.CustomerAddress)

}

else

{

@Html.TextBox("CustomerAddress")

}

}

</p>

<p>

<label>Телефон</label>

@{

if (customersSession != null)

{

@Html.TextBox("CustomerTelephone", customersSession.CustomerTelephone)

}

else

{

@Html.TextBox("CustomerTelephone")

}

}

</p>

<p>

<input type="submit" value="Send">

</p>

}

@{

if (@ViewData["customersFromMemoryCashe"] != null)

{

advertisingAgency.Models.Customers custFromMemoryCashe = (advertisingAgency.Models.Customers)ViewData["customersFromMemoryCashe"];

<div class="infoMemory">

<p>MemoryCache</p>

@Html.DisplayNameFor(x => x.CustomerName):&nbsp @custFromMemoryCashe.CustomerName.ToString();<br />

@Html.DisplayNameFor(x => x.CustomerAddress):&nbsp @custFromMemoryCashe.CustomerAddress.ToString();<br />

@Html.DisplayNameFor(x => x.CustomerTelephone):&nbsp @custFromMemoryCashe.CustomerTelephone.ToString();<br />

</div>

}

}

**Orders.cshtml**

@model advertisingAgency.Models.Orders

@{

ViewData["Title"] = "Orders";

}

<img src="~/images/images.jpg" />

<h1>Заказы</h1>

<table>

<tr>

<th>Дата заключения договора</th>

<th>Срок действия</th>

<th>Место расположения</th>

<th>Цена услуги</th>

</tr>

@foreach (Orders item in ViewBag.orders)

{

<tr>

<td>@item.DateOrder.ToShortDateString()</td>

<td>@item.DateBegin.ToShortDateString() - @item.DateEnd.ToShortDateString()</td>

<td>@item.Location</td>

<td>@item.OrderCost</td>

</tr>

}

</table>

@using (Html.BeginForm("AddOrder", "Home", FormMethod.Post))

{

<p class="add"> Добавить </p>

<p>

<label>Дата заключения договора</label>

<input type="date" name="DateOrder" />

</p>

<p>

<label>Начало</label>

<input type="date" name="DateBegin" />

</p>

<p>

<label>Конец</label>

<input type="date" name="DateEnd" />

</p>

<p>

<label>Место расположения</label>

@Html.EditorFor(order => order.Location)

</p>

<p>

<label>Цена услуги</label>

@Html.EditorFor(order => order.OrderCost)

</p>

<p>

<input type="submit" value="Send">

</p>

}

@{

if (ViewBag.ordFromMemory != null)

{

advertisingAgency.Models.Orders ordersFromMemoryCashe = (advertisingAgency.Models.Orders)ViewBag.ordFromMemory;

<div class="infoMemory">

<p>MemoryCache</p>

@Html.DisplayNameFor(x => x.DateOrder):&nbsp @ordersFromMemoryCashe.DateOrder.ToShortDateString();<br />

@Html.DisplayNameFor(x => x.DateBegin):&nbsp @ordersFromMemoryCashe.DateBegin.ToShortDateString();<br />

@Html.DisplayNameFor(x => x.DateBegin):&nbsp @ordersFromMemoryCashe.DateEnd.ToShortDateString();<br />

@Html.DisplayNameFor(x => x.Location):&nbsp @ordersFromMemoryCashe.Location.ToString();<br />

@Html.DisplayNameFor(x => x.OrderCost):&nbsp @ordersFromMemoryCashe.OrderCost.ToString();<br />

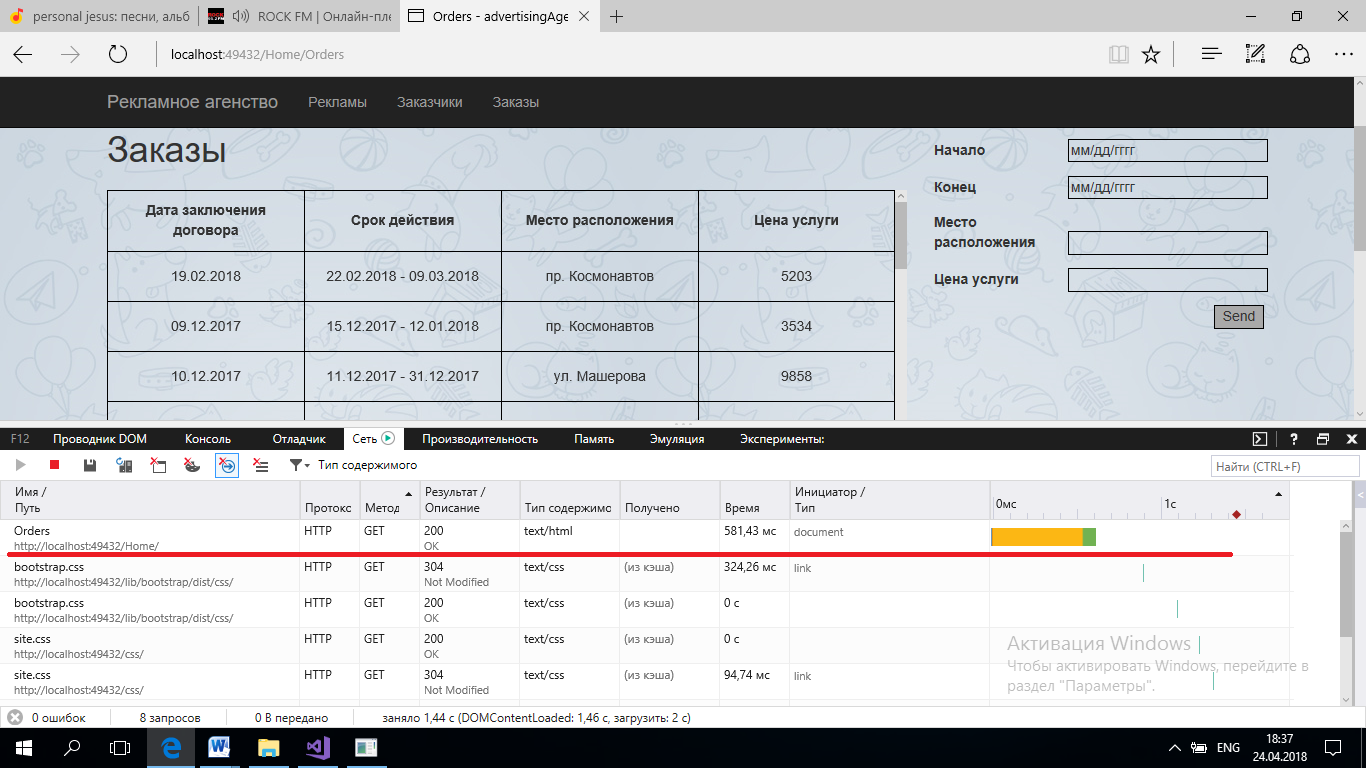
</div>

}

}

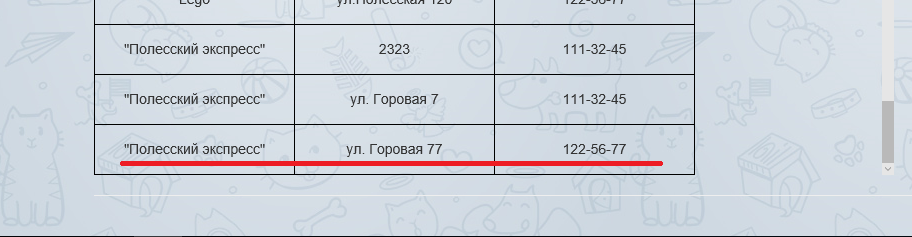
**Визуальные представления HTML страниц и представлений в браузере**:

Пример работы ResponseCache(Orders.cshtml):



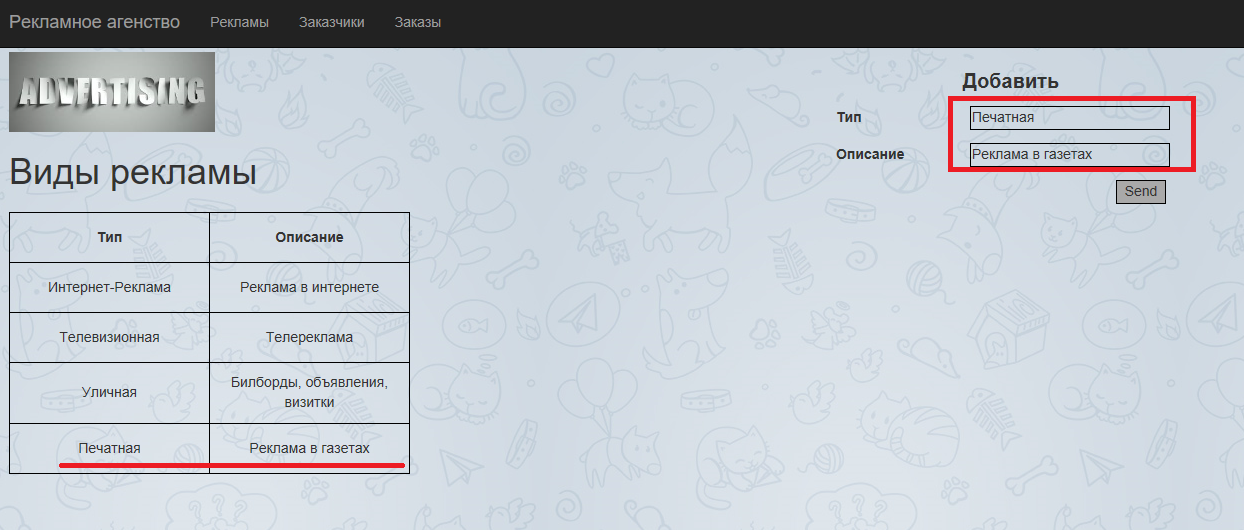
Пример работы Session(Customers.cshtml):



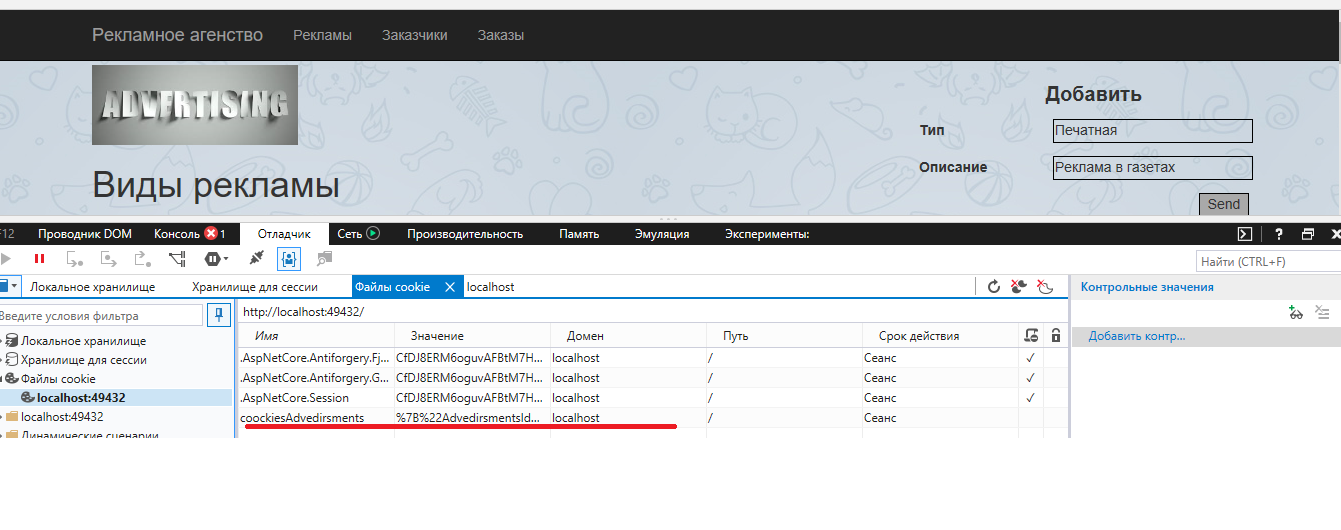


Как видно из скриншотов, данные добавляются в объект в session после добавления в базу и заполняют форму страницы при её загрузке

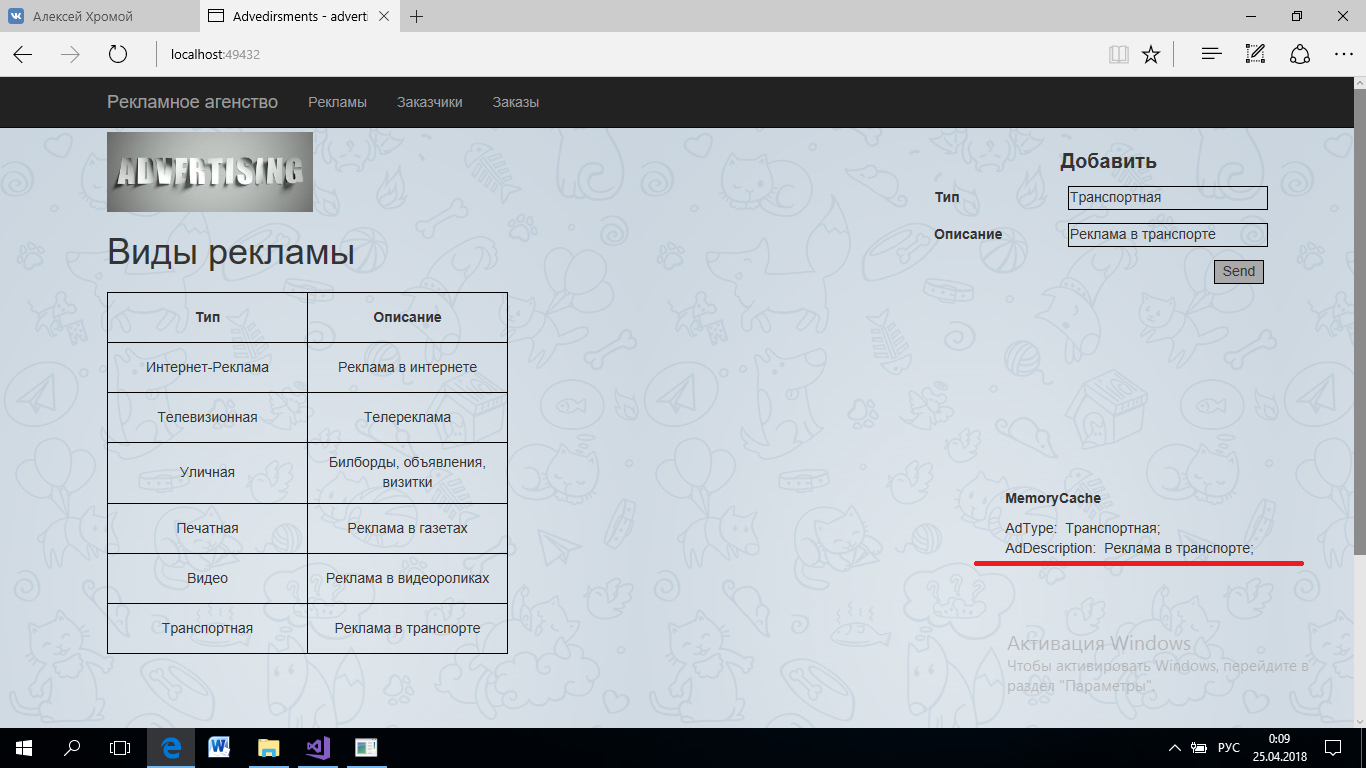
Пример работы Cookies(Index.cshtml):



Как видим, сохраняется состояние элементов формы страницы с использованием куки. В данном случае используется кука coockiesAdvedirsments(рисунок ниже):



Пример работы IMemoryCache(Index.cshtml)



**Вывод:** В ходе выполнения данной лабораторной работы были получены навыки в работе с Microsoft ASP.NET Core. Разобрался с кеширование данных и сохранением данных в cookie, научился работать в MemoryCache и ResponseCache. Задания лабораторной работы выполнены полностью