МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ

«ХАРКІВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»

КАФЕДРА ПРОГРАМНОЇ ІНЖЕНЕРІЇ ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГИЙ УПРАВЛІННЯ

Лабораторна робота № 1

з курсу «Стек технологій .Net ч.2»

Виконав:

студент групи КН 36-а

Кулик В. В.

Перевірила:

Літвінова Ю.С.

ХАРКІВ 2019

Лабораторная работа № 1

WEB-СЕРВИСЫ

**Цель работы**

Познакомиться с технологией создания и использования Web-сервисов.

**Порядок выполнения работы**

1. Создать Web-сервис на основе примера в п. 2.4, который возвращал бы данные, извлеченные из БД (см. лаб. работу № 6).
2. Модифицировать Web-приложение, разработанное в предыдущей работе (Телефонная станция), таким образом, чтобы оно получало данные из сервиса, созданного в предыдущем пункте.

**Ход работы**

Для создания веб-сервиса создадим новы й проект в Visual Studio: ASP.NET Web Service Application.

Изменим код созданного класса сервиса, создав в нем три метода: для чтения, добавления и удаления данных с БД с последующим формированием результирующего DataSet.

using Lab7Service.Models;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.Services;

namespace Lab7Service

{

[WebService(Namespace = "http://lab7.org/")]

[WebServiceBinding(ConformsTo = WsiProfiles.BasicProfile1\_1)]

[System.ComponentModel.ToolboxItem(false)]

public class CallerWebService : System.Web.Services.WebService

{

ProductsEntities productsEntities;

public CallerWebService()

{

productsEntities = new ProductsEntities();

}

[WebMethod]

public List<CallerModel> Callers()

{

return productsEntities.Callers.Select(x => new CallerModel { Id = x.Id, Name = x.Name, Number = x.Number }).ToList();

}

[WebMethod]

public CallerModel Caller(int id)

{

var prod = productsEntities.Callers.Find(id);

if (prod == null)

return null;

return new CallerModel(prod);

}

[WebMethod]

public void DeleteCaller(int id)

{

productsEntities.Callers.Remove(productsEntities.Callers.Find(id));

productsEntities.SaveChanges();

}

[WebMethod]

public void UpdateCaller(int id, CallerModel model)

{

var prod = productsEntities.Callers.Find(id);

if (prod == null)

return;

prod.Name = model.Name;

prod.Number = model.Number;

productsEntities.SaveChanges();

}

[WebMethod]

public void AddCaller(CallerModel model)

{

Caller product = new Caller();

product.Name = model.Name;

product.Number = model.Number;

productsEntities.Callers.Add(product);

productsEntities.SaveChanges();

}

[WebMethod]

public List<CallModel> Calls()

{

return productsEntities.Calls.Select(x => new CallModel { CallerId = x.CallerId, CalledId = x.CalledId, Duration = x.Duration, isActive = x.isActive, Date = x.Date }).ToList();

}

[WebMethod]

public CallModel Call(int callerId, int calledId)

{

var prod = productsEntities.Calls.FirstOrDefault(x => x.CallerId == callerId && x.CalledId == calledId);

if (prod == null)

return null;

return new CallModel(prod);

}

[WebMethod]

public void DeleteCall(int callerId, int calledId)

{

productsEntities.Calls.Remove(productsEntities.Calls.FirstOrDefault(x => x.CallerId == callerId && x.CalledId == calledId));

productsEntities.SaveChanges();

}

[WebMethod]

public void UpdateCall(int callerId, int calledId, CallModel model)

{

var prod = productsEntities.Calls.FirstOrDefault(x => x.CallerId == callerId && x.CalledId == calledId);

if (prod == null)

return;

prod.Date = model.Date;

prod.Duration = model.Duration;

prod.isActive = model.isActive;

productsEntities.SaveChanges();

}

[WebMethod]

public void AddCall(CallModel model)

{

Call product = new Call();

product.CallerId = model.CallerId;

product.CalledId = model.CalledId;

product.Date = model.Date;

product.Duration = model.Duration;

product.isActive = model.isActive;

productsEntities.Calls.Add(product);

productsEntities.SaveChanges();

}

}

}

Листинг 1 – Код сервиса

Протестируем работу сервиса в браузере. Для этого запустим проект и выполним один из описанных выше веб-методов.

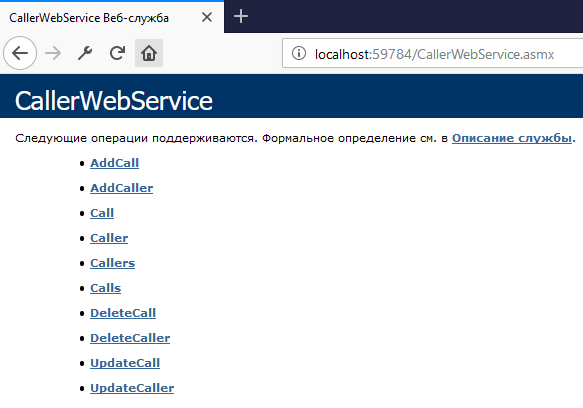


Рисунок 1 – Страница веб-службы

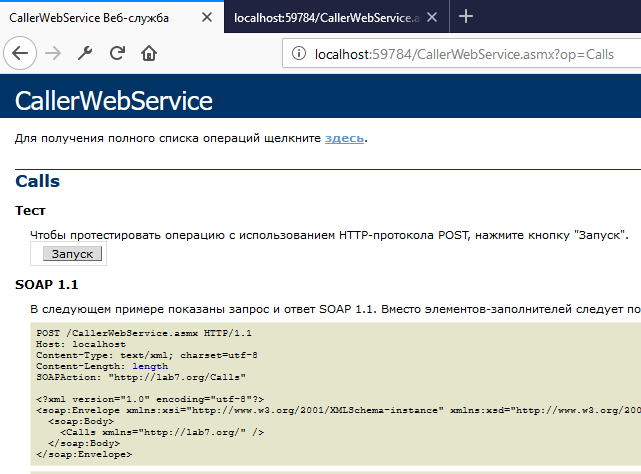


Рисунок 2 – Робота веб-службы

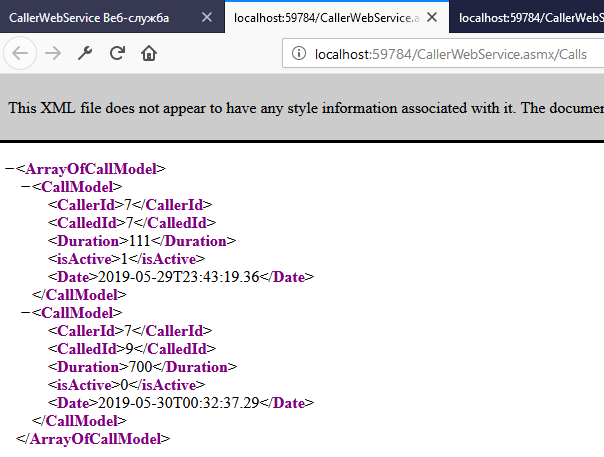


Рисунок 3 – Полученный SOAP файл

Далее следует создать ссылку на сервис в клиентском приложении. В качестве приложения-клиента используем веб-приложение, разработанное в результате предыдущей лабораторной работы.

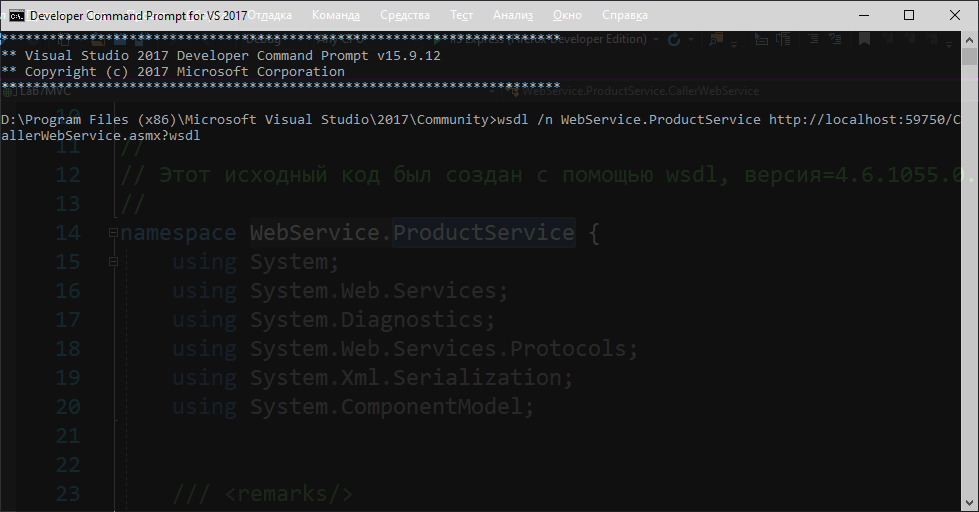


Рисунок 4 – Создание ссылки на веб-сервис

Для этого воспользуемся URL ссылкой веб-сервиса. Во время создания ссылки, сервис должен быть запущен на другом экземпляре Visual Studio.

После создания ссылки, можно создать экземпляр сервиса в программе и модифицировать методы в веб-приложении используя методы веб-сервиса.

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.Mvc;

using WebService.ProductService;

namespace Lab7MVC.Controllers

{

public class CallModelExt

{

public string CallerName { get; set; }

public string CalledName { get; set; }

public int CallerNumber { get; set; }

public int CalledNumber { get; set; }

public int CallerId { get; set; }

public int CalledId { get; set; }

public DateTime Date { get; set; }

public int Duration { get; set; }

public bool isActive { get; set; }

}

public class CallController : Controller

{

CallerWebService service = new CallerWebService();

CallModelExt FromCallModel(CallModel model)

{

CallModelExt res = new CallModelExt();

res.Date = model.Date;

res.isActive = model.isActive != 0;

res.Duration = model.Duration;

res.CallerId = model.CallerId;

res.CallerName = service.Caller(model.CallerId).Name;

res.CalledId = model.CalledId;

var p = service.Caller(model.CalledId);

res.CalledName = p.Name;

res.CalledNumber = p.Number;

res.CallerNumber = service.Caller(model.CallerId).Number;

return res;

}

// GET: Call

public ActionResult Index()

{

return View(service.Calls().Select(x => FromCallModel(x)));

}

private void FillViewBag()

{

ViewBag.Callers = service.Callers().Select(x => new SelectListItem { Text = x.Number.ToString(), Value = x.Id.ToString() });

ViewBag.Calleds = service.Callers().Select(x => new SelectListItem { Text = x.Number.ToString(), Value = x.Id.ToString() });

}

// GET: Call/Details/5

public ActionResult Details(int callerId, int calledId)

{

return View(FromCallModel(service.Call(callerId, calledId)));

}

// GET: Call/Create

public ActionResult Create()

{

FillViewBag();

return View();

}

// POST: Call/Create

[HttpPost]

public ActionResult Create(FormCollection collection)

{

try

{

// TODO: Add insert logic here

CallModel model = new CallModel

{

CallerId = int.Parse(collection["Callers"]),

CalledId = int.Parse(collection["Calleds"]),

Duration = int.Parse(collection["Duration"]),

isActive = collection["isActive"] == "on" ? 1 : 0,

Date = DateTime.Now

};

service.AddCall(model);

return RedirectToAction("Index");

}

catch

{

FillViewBag();

return View();

}

}

// GET: Call/Edit/5

public ActionResult Edit(int callerId, int calledId)

{

FillViewBag();

return View(FromCallModel(service.Call(callerId, calledId)));

}

// POST: Call/Edit/5

[HttpPost]

public ActionResult Edit(int callerId, int calledId, FormCollection collection)

{

try

{

// TODO: Add update logic here

CallModel model = new CallModel

{

CallerId = int.Parse(collection["Callers"]),

CalledId = int.Parse(collection["Calleds"]),

Duration = int.Parse(collection["Duration"]),

isActive = collection["isActive"] == "on" ? 1 : 0,

Date = DateTime.Now

};

service.UpdateCall(callerId, calledId, model);

return RedirectToAction("Index");

}

catch

{

FillViewBag();

return View();

}

}

// GET: Call/Delete/5

public ActionResult Delete(int callerId, int calledId)

{

return View(FromCallModel(service.Call(callerId, calledId)));

}

// POST: Call/Delete/5

[HttpPost]

public ActionResult Delete(int callerId, int calledId, FormCollection collection)

{

try

{

// TODO: Add delete logic here

service.DeleteCall(callerId, calledId);

return RedirectToAction("Index");

}

catch

{

return View();

}

}

}

}

Листинг 2 – Контроллер веб-страницы

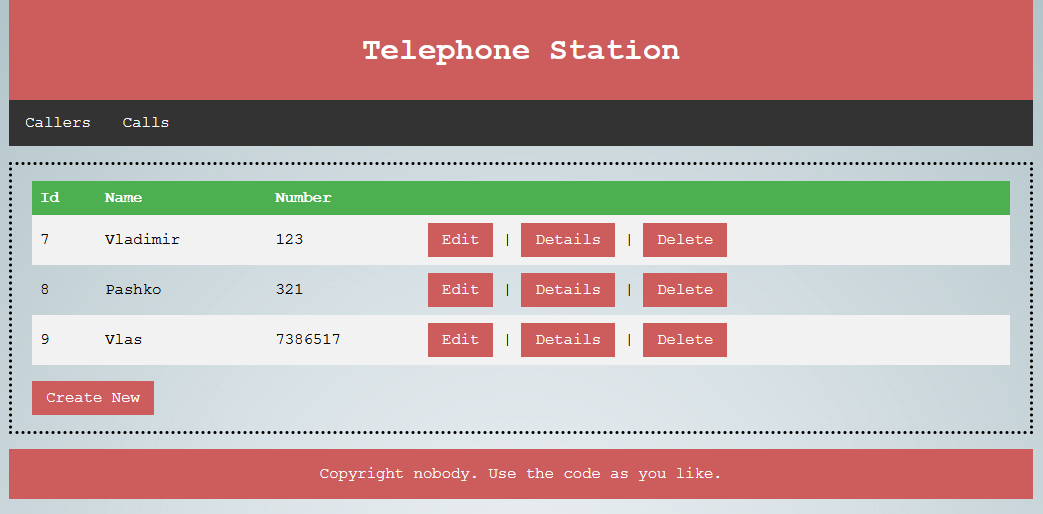


Рисунок 5 – Результат работы веб-приложения

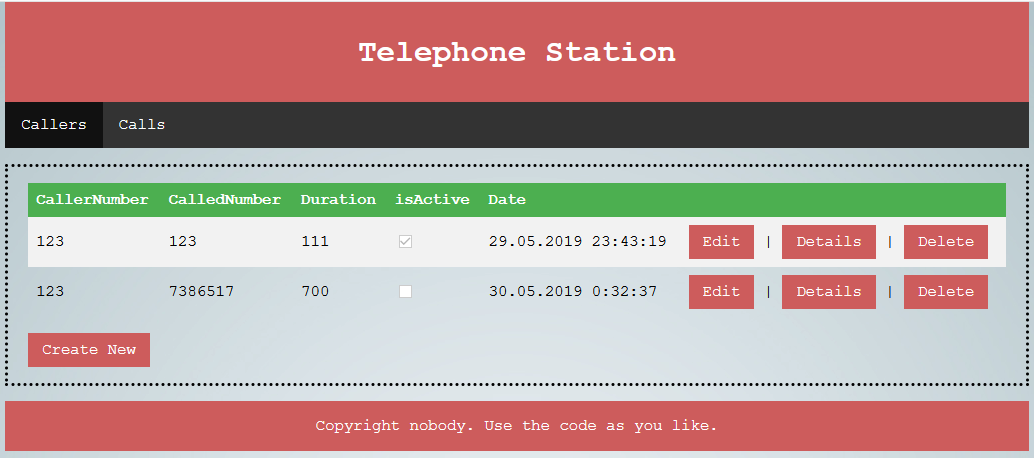


Рисунок 6 – Результат работы веб-приложения



Рисунок 7 – Результат работы веб-приложения

**Выводы**

На данной лабораторной работе были изучены основные особенности использования веб-сервисов ASP.NET и интеграция их в клиентские приложения. Было усовершенствованно веб-приложение, разработанное на прошлой лабораторной работе, путем интеграции в него разработанного веб-сервиса. Разработанный веб-сервис реализует методы чтения, добавления и удаления данных в БД.