**МИНОБРНАУКИ РОССИИ**

Федеральное государственное бюджетное образовательное учреждение высшего образования «Ижевский государственный  
технический университет имени М.Т. Калашникова»

(ФГБОУ ВО «ИжГТУ имени М.Т. Калашникова»)

Кафедра «Вычислительная техника»

Отчет по лабораторной работе №3

по дисциплине «Объектно-ориентированное программирование»

на тему «Проектирование приложений, SOLID, паттерны проектирования»

Вариант 14

Выполнил: студент гр. Б06-781-1

Пермитин И.А.

Проверил: к.т.н., доцент кафедры ВТ

Марков Е.М.

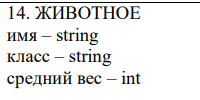
Ижевск 2020

**Задание**

1. Разработать веб-приложение по заданию на основе ASP.NET или любой другой, с использованием MVC.

2. Приложение должно содержать страницу с перечнем созданных записей (объектов) и, хотя бы одно действие для каждого из них, например, «подробнее», «купить» или подобное, открывающее другую страницу.

3. (дополнительно, не обязательно для выполнения) Хранить данные об записях (объектах) в базе данных.



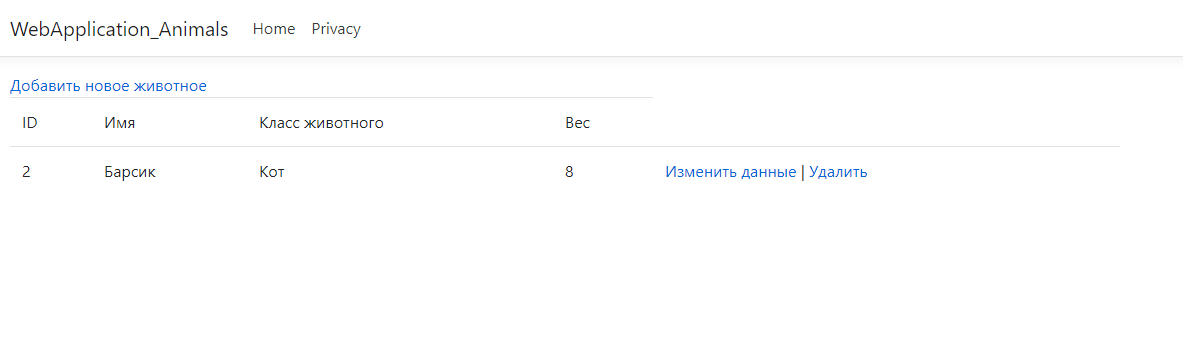
**Описание использованных технологий**

Лабораторная работа выполнялась в IDE «Visual Studio 2019». В качестве шаблона для работы использовался ASP.NET Core Web Application.

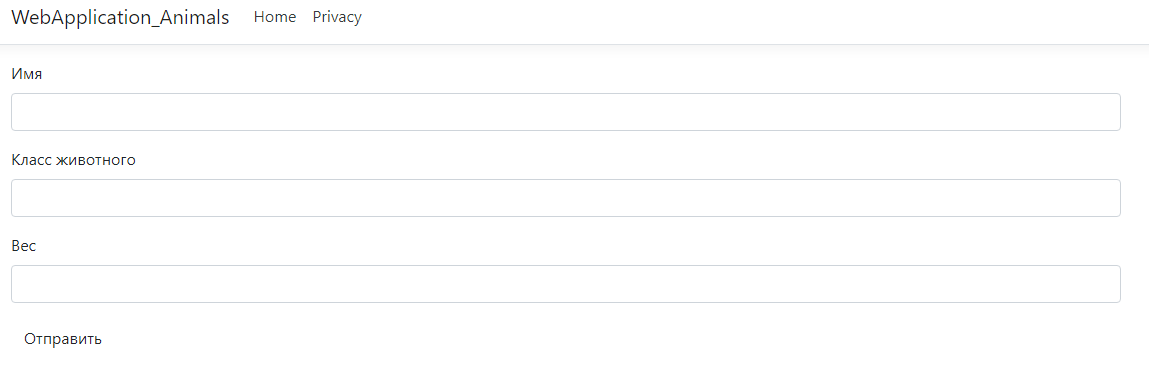
Тип проекта - Web Application (MVC).

**Работа приложения**

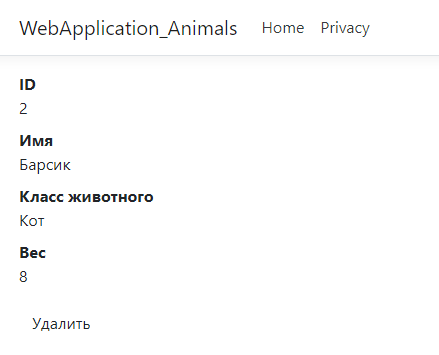
Главная страница веб-приложения. На ней отображаются все записи, находящиеся в БД.



При нажатии на «Добавить новое животное» открывается страница, на которой необходимо заполнить данные новой записи.



При нажатии «Удалить» открывается страница, на которой находится информация о записи, которую вы собираетесь удалить. Кнопка «Удалить» в нижней части страницы выполнит удаление записи из БД.



**Листинг программы**

Animal.cs:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

namespace WebApplication\_Animals.Models

{

public class Animal

{

public int Id { get; set; }

public string name { get; set; }

public string animalClass { get; set; }

public int weight { get; set; }

}

}

ApplicationContext.cs:

using Microsoft.EntityFrameworkCore;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

namespace WebApplication\_Animals.Models

{

public class ApplicationContext : DbContext

{

public DbSet<Animal> animals { get; set; }

public ApplicationContext(DbContextOptions<ApplicationContext> options) : base(options)

{

Database.EnsureCreated();

}

}

}

HomeController.cs:

using System;

using System.Collections.Generic;

using System.Diagnostics;

using System.Linq;

using System.Threading.Tasks;

using Microsoft.AspNetCore.Mvc;

using Microsoft.EntityFrameworkCore;

using Microsoft.Extensions.Logging;

using WebApplication\_Animals.Models;

namespace MyWebApplication.Controllers

{

public class HomeController : Controller

{

private ApplicationContext db;

public HomeController(ApplicationContext context)

{

db = context;

}

public async Task<IActionResult> Index()

{

return View(await db.animals.ToListAsync());

}

public IActionResult Create()

{

return View();

}

[HttpPost]

public async Task<IActionResult> Create(Animal worker)

{

db.animals.Add(worker);

await db.SaveChangesAsync();

return RedirectToAction("Index");

}

public async Task<IActionResult> Details(int? id)

{

if (id != null)

{

Animal worker = await db.animals.FirstOrDefaultAsync(p => p.Id == id);

if (worker != null)

return View(worker);

}

return NotFound();

}

public async Task<IActionResult> Edit(int? id)

{

if (id != null)

{

Animal worker = await db.animals.FirstOrDefaultAsync(p => p.Id == id);

if (worker != null)

return View(worker);

}

return NotFound();

}

[HttpPost]

public async Task<IActionResult> Edit(Animal worker)

{

db.animals.Update(worker);

await db.SaveChangesAsync();

return RedirectToAction("Index");

}

[HttpGet]

[ActionName("Delete")]

public async Task<IActionResult> ConfirmDelete(int? id)

{

if (id != null)

{

Animal animal = await db.animals.FirstOrDefaultAsync(p => p.Id == id);

if (animal != null)

return View(animal);

}

return NotFound();

}

[HttpPost]

public async Task<IActionResult> Delete(int? id)

{

if (id != null)

{

Animal worker = await db.animals.FirstOrDefaultAsync(p => p.Id == id);

if (worker != null)

{

db.animals.Remove(worker);

await db.SaveChangesAsync();

return RedirectToAction("Index");

}

}

return NotFound();

}

}

}

Startup.cs:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using Microsoft.AspNetCore.Builder;

using Microsoft.AspNetCore.Hosting;

using Microsoft.AspNetCore.HttpsPolicy;

using Microsoft.EntityFrameworkCore;

using Microsoft.Extensions.Configuration;

using Microsoft.Extensions.DependencyInjection;

using Microsoft.Extensions.Hosting;

using WebApplication\_Animals.Models;

namespace WebApplication\_Animals

{

public class Startup

{

public Startup(IConfiguration configuration)

{

Configuration = configuration;

}

public IConfiguration Configuration { get; }

// This method gets called by the runtime. Use this method to add services to the container.

public void ConfigureServices(IServiceCollection services)

{

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

services.AddDbContext<ApplicationContext>(options =>

options.UseSqlServer(connection));

services.AddControllersWithViews();

}

// This method gets called by the runtime. Use this method to configure the HTTP request pipeline.

public void Configure(IApplicationBuilder app, IWebHostEnvironment env)

{

if (env.IsDevelopment())

{

app.UseDeveloperExceptionPage();

}

else

{

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

// The default HSTS value is 30 days. You may want to change this for production scenarios, see https://aka.ms/aspnetcore-hsts.

app.UseHsts();

}

app.UseHttpsRedirection();

app.UseStaticFiles();

app.UseRouting();

app.UseAuthorization();

app.UseEndpoints(endpoints =>

{

endpoints.MapControllerRoute(

name: "default",

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

});

}

}

}