**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ**

федеральное государственное бюджетное образовательное учреждение

высшего образования

**«УЛЬЯНОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

Факультет  *информационных систем и технологий*

Кафедра  *Измерительно-вычислительные комплексы*

Дисциплина  *Базы данных*

КУРСОВАЯ РАБОТА

Тема  *Автоматизированная информационная система спортивного клуба*

Выполнил студент  */ /*

*подпись инициалы, фамилия*

Курс  *2* Группа  *ИСТбд-21*

Направление  *09.03.02 «Информационные системы и технологии»*

Руководитель  *доцент кафедры ИВК, к.т.н., доцент*

*должность, учёная степень, учёное звание*

*Родионов Виктор Викторович*

*фамилия,имя, отчество*

Дата сдачи:

« » 20 г.

Дата защиты:

« » 20 г.

Оценка:

Ульяновск

20 .

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ**

федеральное государственное бюджетное образовательное учреждение

высшего образования

**«УЛЬЯНОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

Факультет  *информационных систем и технологий*

Кафедра  *Измерительно-вычислительные комплексы*

Дисциплина  *Базы данных*

ЗАДАНИЕ НА КУРСОВУЮ РАБОТУ

студенту  *ИСТбд-21*

*группа фамилия, инициалы*

Тема работы *Автоматизированная информационная система спортивного клуба*

Срок сдачи законченной работы « » 20 г.

Исходные данные к работе *методические указания к выполнению курсовой*

*(базовое предприятие, характер курсовой работы:*

*работы для студентов направления 09.03.03 «Информационные системы*

*задание кафедры, инициативная НИР, рекомендуемая литература, материалы практики)*

*и технологии» по дисциплине «Базы данных» Родионова В.В.*

Содержание пояснительной записки *список использованных обозначений*

*и сокращений, введение, техническое задание, информационное обеспечение*

*системы, алгоритмическое обеспечение системы, прикладное программное*

*обеспечение системы, руководство пользователя, заключение, список*

*использованных источников*

Перечень графического материла

Руководитель  *доцент каф. ИВК / В.В. Родионов /*

*должность подпись инициалы, фамилия*

« » 20 г.

Студент  */ /*

*подпись инициалы, фамилия*

« » 20 г.

Содержание

[Введение……. 4](#_Toc11512419)

[1.Техническое задание 6](#_Toc11512420)

[1.1 Общие сведения 6](#_Toc11512421)

[1.2 Назначение и цели создания системы 6](#_Toc11512422)

[1.3 Характеристика объекта автоматизации 6](#_Toc11512423)

[1.4 Требования к системе 7](#_Toc11512424)

[1.5 Состав и содержание работ по созданию системы 8](#_Toc11512425)

[1.6 Порядок контроля и приёмки системы 8](#_Toc11512426)

[1.7 Требования к документированию 8](#_Toc11512427)

[2. Информационное обеспечение системы 9](#_Toc11512428)

[2.1 Концептуальная схема базы данных 9](#_Toc11512429)

[2.1.1 Модель «сущность-связь» 9](#_Toc11512430)

[2.1.2 Сущности и их атрибуты 9](#_Toc11512431)

[2.1.3 Связи между сущностями 10](#_Toc11512432)

[2.2 Внутренняя схема базы данных 12](#_Toc11512433)

[2.2.1 База данных системы 12](#_Toc11512434)

[3 Прикладное программное обеспечение системы 16](#_Toc11512435)

[3.1 Общая характеристика прикладного программного обеспечения 16](#_Toc11512436)

[3.2 Структура и состав прикладного программного обеспечения 17](#_Toc11512437)

[3.3 Особенности реализации и сопровождения 18](#_Toc11512438)

[4 Руководство пользователя 19](#_Toc11512439)

[4.1 Общие сведения 19](#_Toc11512440)

[4.2 Порядок и особенности работы 19](#_Toc11512441)

[4.3 Исключительные ситуации 25](#_Toc11512442)

[Список использованных источников 26](#_Toc11512443)

[Приложение А. Исходные тексты программных модулей 27](#_Toc11512444)

Лист

№ докум.

Подпись

Дата

Лист

Разраб.

Провер.

Родионов

Реценз.

Н. Контр.

Родионов

Утверд.

Пояснительная записка

Лит.

Листов

ИСТбд-21

## Введение

Лист

№ докум.

Подпись

Дата

Лист

Спортивный клуб - общественная или частная организация, объединяющая спортсменов и любителей спорта. Существуют спортивные клубы общего характера и специализированные (по видам спорта).

Реализация законодательства о физической культуре и спорте, мероприятий государственных программ создает все условия для поступательного развития физкультурно-оздоровительного движения в стране.

Среди основных задач на первое место выдвигается прежде всего привлечение молодежи к спорту, это отражено в программах: «Спорт для всех» – «Необходимо усиление пропаганды «Спорта для Всех» в учебных заведениях, на производстве, в средствах массовой информации и т.д. Соответствующие методические программы создаются при помощи олимпийского комитета, который также включается в дело организации массовых соревнований с большим числом участников.

Невооруженным глазом простого обывателя, каким я и являюсь, видно, что молодежный спорт в России ещё не достиг такой степени развития, регламентированности и общезначимости, как в Европе. Сейчас трудно говорить о каких-то многомиллионных проектах и компаниях в этой сфере, прежде всего потому, что эта проблема видится малозначимой и не требующей поспешного и немедленного решения, по сравнению с такими как наркомания, молодежный алкоголизм и преступность, бедственное существование основного слоя населения и так далее. Но несмотря на это есть люди, которые считают физическое воспитание и спорт важными фактором формирования молодежи. Это фактор здоровья населения.

К сожалению, многие люди не соблюдают самых простейших, обоснованных наукой норм здорового образа жизни. Одни становятся жертвами малоподвижности (гиподинамии), вызывающей преждевременное старение, другие излишествуют в еде с почти неизбежным в этих случаях развитием ожирения, склероза сосудов, а у некоторых - сахарного диабета, третьи не умеют отдыхать, отвлекаться от производственных и бытовых забот, вечно беспокойны, нервны, страдают бессонницей, что в конечном итоге приводит к многочисленным заболеваниям внутренних органов. Некоторые люди, поддаваясь пагубной привычке к курению и алкоголю, активно укорачивают свою жизнь. Активный отдых на фоне гиподинамии может восполнить нехватку двигательной активности, что приводит к улучшению состояния здоровья, поднятию настроения и т.д.

Лист

№ докум.

Подпись

Дата

Лист

Очень популярным видом активного отдыха в выходные дни являются спортивные игры (бадминтон, баскетбол, волейбол, футбол, теннис, хоккей и др.). Они оказывают разностороннее влияние на организм занимающихся. Включая разнообразные формы двигательной деятельности (бег, ходьбу, прыжки, метание, удары, ловлю и броски, различные силовые элементы), спортивные игры развивают глазомер точность и быстроту движений, мышечную силу, способствуют развитию сердечно-сосудистой, нервной, дыхательной систем, улучшению обмена веществ, укреплению опорно-двигательного аппарата.

Физическая культура и спорт относятся к числу важных факторов повышения долголетия, продления жизни населения, активного отдыха. Физкультурно-спортивному движению придан статус одного из главных приоритетов политики государства.

При изучении моей предметной области использовались следующие источники:

1. Статья «Физическая культура и спорт как отрасль социальной сферы» с сайта sporthealth.ru описывала какое место занимает физическая подготовка в нашей стране и какие средства для этого применяются.

2. Статья «Спорт как досуг» с сайта sportschools.ru описывала способы активного отдыха для поддержания здорового образа жизни.

3. Статья «Система физкультуры и спорта в России» с сайта sportfiction.ru продемонстрировала отличие спортивной культуры нашей страны от других стран.

4. Статья «Онлайн-книга Изучаем ASP.NET.MVC4» с сайта metanit.com помогла разобраться в работе с MVC и Entity Framework.

Лист

№ докум.

Подпись

Дата

Лист

5. Статья «Спорт это жизнь» с сайта eurosport.ru описывала, что спорт это здоровье и счастье, благодаря которым человек будет менее подвергнут заражению, в следствии чего человек должен уделять спорту время, чтобы продлить свою жизнь.

## 1.Техническое задание

Лист

№ докум.

Подпись

Дата

Лист

## 1.1 Общие сведения

Автоматизированная система «Спортивный клуб» далее система.

## 1.2 Назначение и цели создания системы

1.2.1 Назначение системы

Область автоматизируемой деятельности: «Спортивный клуб». Система будет доступна узкому кругу людей, имеющих доступ к администрированию спортивного клуба.

1.2.2 Цели создания системы

1. Оптимизировать управление спортивным клубом.

2. Минимизировать возможность возникновения непредвиденных ситуаций путем анализа информации, получаемой из системы.

## 1.3 Характеристика объекта автоматизации

Обьект автоматизации «Спортивный клуб» занимается, организацией тренировок для спортсменов, наймом сотрудников различных направлений и организацией спортивных мероприятий.

Спортивный клуб занимается поиском и наймом работников определенных профессий, которые могут обслуживать, организовывать и управлять организацией, заниматься со спортсменами и тренировать их.

В спортивный клуб могут записаться любой желающий не имеющий противопоказаний по здоровью и согласными на обработку, хранение и использование персональных данных в целях организации. Вступившие в спортивный клуб будут распределены по группам в зависимости от своего возраста, после чего спортсмены могут заниматься на занятиях.

Спортивный клуб и сотрудники могут организовывать мероприятия различных уровней и типов, такие как спортивные соревнования. Соревнования могут быть региональными и федеральными, от этого параметра зависят затраты на организацию и определение с местом проведения мероприятия, для чего необходимо находить и договариваться со спонсорами. В соревнованиях так же могут принимать участие спортсмены различных спортивных клубов, а так же работники данных клубов (тренера).

Лист

№ докум.

Подпись

Дата

Лист

## 1.4 Требования к системе

1.4.1 Требования к системе в целом

1.4.1.1 Требования к структуре и функционированию системы

Определяется общей постановкой задачи задания на курсовую работу.

1.4.1.2 Требования к защите информации он несанкционированного доступа

Требований нет.

1.4.2 Требования к функциям, выполняемым системой

Система должна выполнять следующие функции:

- запись данных о спортивном клубе: название, адрес, доп. услуги и т.д.

- запись персональных и профессиональных данных о нанимаемом сотруднике.

- распределение сотрудников по спортивным клубам

- получение персональных данных и распределение по группам спортсмена, желающего вступить в спортивный клуб.

- организация и проведение соревновательных мероприятий с возможностью участия спортсменов, тренирующихся в спортивном клубе

1.4.3 Требования к видам обеспечения

1.4.3.1 Требования к техническому обеспечению

Процессор – Intel Core i5-6300HQ

Оперативная память – 8 Гб

Видеокарта – NVIDIA GeForce GTX 960

Жесткий диск – WDC WD10SPCX-24HWST1

Монитор – Samsung SyncMaster 963nw

Количество ядер - 4

1.4.3.2 Требования к программному обеспечению

При разработке системы будет использоваться Windows 10.

-Microsoft Visual Studio 2012

Лист

№ докум.

Подпись

Дата

Лист

- Microsoft SQL Server 2019

- ERConstructor 2.0 при создании диаграммы “сущность-связь”

## 1.5 Состав и содержание работ по созданию системы

Определяется этапами выполнения работы задания на курсовую работу.

## 1.6 Порядок контроля и приёмки системы

Определяется порядком защиты и критериями оценки работы задания на курсовую работу.

## 1.7 Требования к документированию

Удаляемые части пояснительной записки:

1. Список использованных обозначений и сокращений;

2. Алгоритмическое обеспечение системы

3. Анализ концептуальной схемы

4. Заключение.

## 2. Информационное обеспечение системы

Лист

№ докум.

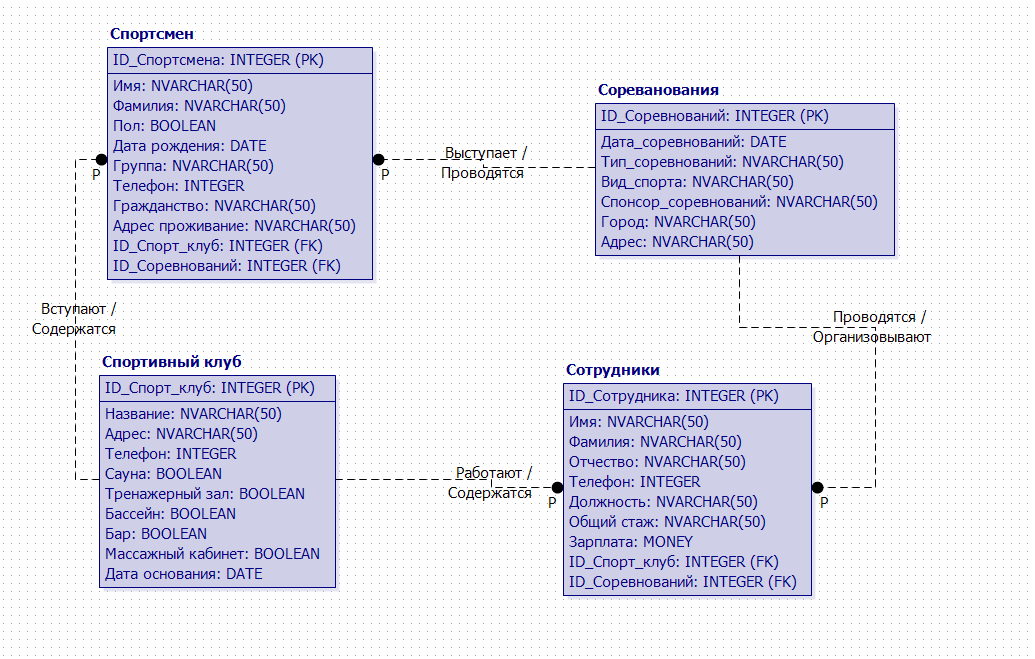
Подпись

Дата

Лист

## 2.1 Концептуальная схема базы данных

## 2.1.1 Модель «сущность-связь»

2.1.2 Сущности и их атрибуты

1.Сущность «Спортивный клуб» включает в себя описание данных о спортивном клубе и его услуги.

2.Сущность «Сотрудники» включает в себя персональные данные работника и его профессиональные характеристики.

3.Сущность «Спортсмен» включает в себя персональные данные спортсмена и группу, в которую он определен.

4.Сущность «Соревнования» включает в себя данные о проводимом соревновании.

Атрибут «Тип соревнований» принимает значения: региональные, федеральные.

## 2.1.3 Связи между сущностями

Лист

№ докум.

Подпись

Дата

Лист

1. Сущность "Спортивный клуб" имеет неидентифицирующую связь 1:N с сущностью "Сотрудник". Минимальное кардинальное число сущности "Спортивный клуб" 1 и максимальное N, потому что как минимум должен быть 1 сотрудник в спортивном клубе и сотрудников может быть много в 1 спортивном клубе. Минимальное и максимальное кардинальное число сущности "Сотрудники" 1, потому что сотрудники могут состоять только в 1 спортивном клубе.

2. Сущность "Спортивный клуб" имеет неидентифицирующую связь 1:N с сущностью "Спортсмен". Минимальное кардинальное число сущности "Спортивный клуб" 1 и максимальное N, потому что как минимум должен быть 1 спортсмен в спортивном клубе и спортсменов может быть много в 1 спортивном клубе. Минимальное и максимальное кардинальное число сущности "Спортсмен" 1, потому что спортсмен может существовать только в 1 спортивном клубе.

3. Сущность «Соревнования» имеет неидентифицирующую связь 1:N с сущностью "Спортсмен". Минимальное кардинальное число сущности «Соревнования» 1 и максимальное N, потому что как минимум должен быть 1 участник в соревновании и в 1 соревновании могут участвовать множество спортсменов. Минимальное и максимальное кардинальное число сущности «Спортсмен» 1, потому что 1 спортсмен может учавствовать только в 1 соревновании.

4. Сущность «Соревнования» имеет неидентифицирующую связь 1:N с сущностью «Сотрудники». Минимальное кардинальное число сущности «Соревнования» 1 и максимальное N, потому что как минимум должен быть 1 сотрудник на соревновании и на 1 соревновании могут принимать участие множество сотрудников. Минимальное и максимальное кардинальное число сущности «Сотрудники» 1, потому что 1 сотрудник может быть на 1 соревновании.

## 2.2 Внутренняя схема базы данных

Лист

№ докум.

Подпись

Дата

Лист

## 2.2.1 База данных системы

1. Таблица «Спортивный клуб»

Создание таблицы «Спортивный клуб» с первичным ключом «ID\_Спорт\_клуба»

CREATE TABLE [dbo].[Спортивный\_клуб](

[ID\_Спорт\_клуба] [int] IDENTITY(1,1) NOT NULL,

[Название] [nvarchar](max) NULL,

[Адрес] [nvarchar](max) NULL,

[Телефон] [int] NULL,

[Сауна] [bit] NULL,

[Тренажерный\_зал] [bit] NULL,

[Бассейн] [bit] NULL,

[Бар] [bit] NULL,

[Массажный\_кабинет] [bit] NULL,

[Дата\_основания] [datetime] NULL,

CONSTRAINT [PK\_dbo.Спортивный\_клуб] PRIMARY KEY CLUSTERED

(

[ID\_Спорт\_клуба] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]

) ON [PRIMARY] TEXTIMAGE\_ON [PRIMARY]

1. Таблица «Сотрудники»

Создание таблицы «Сотрудники» с первычным ключом «ID\_Сотрудника»

CREATE TABLE [dbo].[Сотрудники](

[ID\_Сотрудника] [int] IDENTITY(1,1) NOT NULL,

[Имя] [nvarchar](max) NULL,

[Фамилия] [nvarchar](max) NULL,

[Отчество] [nvarchar](max) NULL,

[Телефон] [int] NULL,

[Должность] [nvarchar](max) NULL,

[Общий\_стаж] [nvarchar](max) NULL,

[Зарплата] [decimal](18, 2) NULL,

[ID\_Спорт\_клуба] [int] NULL,

[Соревнования\_ID\_Соревнований] [int] NULL,

CONSTRAINT [PK\_dbo.Сотрудники] PRIMARY KEY CLUSTERED

(

[ID\_Сотрудника] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]

) ON [PRIMARY] TEXTIMAGE\_ON [PRIMARY]

1. Таблица «Спортсмен»

Создание таблицы «Спортсмен» с первычным ключом «ID\_Спортсмена»

Лист

№ докум.

Подпись

Дата

Лист

CREATE TABLE [dbo].[Спортсмен](

[ID\_Спортсмена] [int] IDENTITY(1,1) NOT NULL,

[Имя] [nvarchar](max) NULL,

[Фамилия] [nvarchar](max) NULL,

[Пол] [bit] NULL,

[Дата\_рождения] [datetime] NULL,

[Группа] [nvarchar](max) NULL,

[Телефон] [int] NULL,

[Гражданство] [nvarchar](max) NULL,

[Адрес\_проживания] [nvarchar](max) NULL,

[Спортивный\_клуб\_ID\_Спорт\_клуба] [int] NULL,

[Соревнования\_ID\_Соревнований] [int] NULL,

CONSTRAINT [PK\_dbo.Спортсмен] PRIMARY KEY CLUSTERED

(

[ID\_Спортсмена] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]

) ON [PRIMARY] TEXTIMAGE\_ON [PRIMARY]

1. Таблица «Соревнования»

Создание таблицы «Соревнования» с первычным ключом «ID\_Соревнований»

CREATE TABLE [dbo].[Соревнования](

[ID\_Соревнований] [int] IDENTITY(1,1) NOT NULL,

[Дата\_соревнований] [datetime] NULL,

[Тип\_соревнований] [nvarchar](max) NULL,

[Вид\_спорта] [nvarchar](max) NULL,

[Спонсор\_соревнований] [nvarchar](max) NULL,

[Город] [nvarchar](max) NULL,

[Адрес] [nvarchar](max) NULL,

CONSTRAINT [PK\_dbo.Соревнования] PRIMARY KEY CLUSTERED

(

[ID\_Соревнований] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]

) ON [PRIMARY] TEXTIMAGE\_ON [PRIMARY]

Хранимая процедура

Данная хранимая процедура срабатывает при добавлении нового спортсмена в таблицу и в зависимости от даты рождения, расчитывается его возраст и сравнивается с числом 18, если его возраст больше 18, то спортсмен попадает в старшую группу, а если же меньше, то попадает в младшую группу.

## CREATE PROCEDURE [dbo].[TestAge]

Лист

№ докум.

Подпись

Дата

Лист

## AS BEGIN

## DECLARE @id int;

## SET @id = 0;

## SELECT @id = MAX(ID\_Спортсмена) FROM Спортсмен

## DECLARE @age datetime, @year int;

## SELECT @age = Спортсмен.Дата\_рождения FROM Спортсмен WHERE ID\_Спортсмена = @id

## SET @year = YEAR(@age)

## SET @year = 2019 - @year

## IF (@year >= 18)

## BEGIN

## UPDATE Спортсмен SET Группа = 'Старшая' WHERE ID\_Спортсмена = @id;

## GOTO BrancheOne;

## END;

## ELSE

## UPDATE Спортсмен SET Группа = 'Младшая' WHERE ID\_Спортсмена = @id;

## GOTO BrancheTwo;

## BrancheOne:

## BEGIN

## PRINT ('Добавлен в старшую группу');

## SELECT \* FROM Спортсмен WHERE ID\_Спортсмена = @id;

## GOTO Branche3

## END;

## Branche3:

## BEGIN

## RETURN

## END;

## BrancheTwo:

## BEGIN

## PRINT ('Добавлен в младшую группу');

## SELECT \* FROM Спортсмен WHERE ID\_Спортсмена = @id;

## GOTO Branche3

## END;

## END;

## 3 Прикладное программное обеспечение системы

Лист

№ докум.

Подпись

Дата

Лист

## 3.1 Общая характеристика прикладного программного обеспечения

Инструментальная среда разработки - Visual Studio 2019.  
СУБД - Microsoft SQL Server 2012.  
Технология разработки - ASP.NET.MVC.  
Технология доступа к данным - ADO.NET Entity Framework.  
Использование ERConstructor 2.0 при создании диаграммы “сущность-связь”.  
Все операции реализуются корректно и с сохранением ссылочной и логической целостности.   
Размер каждой из рабочих таблиц - не менее 7 записей, справочников – 3 записей.  
Базовые возможности интерфейса для всех таблиц баз данных:  
1. Просмотр данных в виде таблиц  
2. Ввод новых и редактирование уже существующих записей  
3. Удаление записей  
Реализованы 3 фильтрации:   
1. Фильтрация в таблице «Соревнования» по названию города.  
2. Фильтрация в таблице «Спортсмен» по гражданству.  
3. Фильтрация в таблице «Сотрудники» по фамилии.

Реализованы 2 поиска:

1. Поиск по базе данных в таблице «Спортсмен» по фамилии.

2. Поиск по базе данных в таблице «Спортивный клуб» по названию.

Количество элементов в базе данных:

1. Таблиц – 4.

2. Атрибутов – 34.

3. Типов связей – 1.

Лист

№ докум.

Подпись

Дата

Лист

4. Хранимых процедур – 1

## 3.2 Структура и состав прикладного программного обеспечения

Состав и содержание папок:

1. AppStart – папка, которой хранятся настройки системы.
2. Content – папка, содержащая таблицы стилей.
3. Controllers – папка содержит основные программные модули на языке С#.
4. Filters – папка, содержащая фильтры, используемые системой.
5. fonts – папка, содержащая стили шрифтов.
6. Models – папка, содержащая классы сущностей системы.
7. Scripts – папка с JavaScript скриптами.
8. Views – папка, содержащая представления системы.  
   Описание контроллеров:
9. HomeController – контроллер, отвечающий за вывод главной страницы, на которой присутствуют ссылки на все созданные таблицы. Количество значимых строк кода: 1
10. СоревнованияController – контроллер, отвечающий за работу с таблицей «Соревнования». Возможны следующие действия: создание, удаление, редактирование таблицы, подробный просмотр, фильтрация по полю город. Количество значимых строк: 41.
11. СотрудникиController – контроллер, отвечающий за работу с таблицей «Сотрудники». Возможны следующие действия: создание, удаление, редактирование таблицы, подробный просмотр, фильтрация по полю фамилия. Количество значимых строк: 45.

Лист

№ докум.

Подпись

Дата

Лист

1. Спортивный\_клубController – контроллер, отвечающий за работу с таблицей «Спортивный\_клуб». Возможны следующие действия: создание, удаление, редактирование таблицы, подробный просмотр, поиск по полю название. Количество значимых строк: 42.
2. СпортсменController – контроллер, отвечающий за работу с таблицей «Спортсмен». Возможны следующие действия: создание, удаление, редактирование таблицы, подробный просмотр, фильтрация по полю гражданство, поиск по полю фамилия. Количество значимых строк: 40.

## 3.3 Особенности реализации и сопровождения

Особенностей реализации и сопровождения в данной системе отсутствуют.

## 4 Руководство пользователя

Лист

№ докум.

Подпись

Дата

Лист

## 4.1 Общие сведения

Система предназначена для оптимизации управления спортивным клубом.

Система позволяет управлять большим объёмом данных без особых навыков. С помощью данной системы пользователь может создавать, удалять, изменять записи в базе данных. Все манипуляции с базой данных происходят в веб интерфейсе, который прост и интуитивно понятен. Система предназначена для узкого круга людей, кто имеет доступ к администрированию клубом.

## 4.2 Порядок и особенности работы

1.Главная страница (рис.1). На данной странице расположены все таблицы, с которыми может взаимодействовать пользователь.

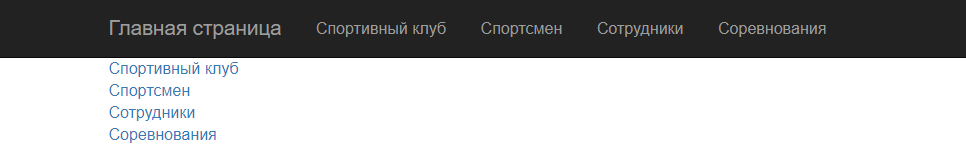


Рисунок 1­­­­­ ­- Главная страница

2.Страница «Спортивный клуб» (рис.2). На данной странице отображается вся информация об спортивном клубе. Есть возможность поиска по названию. Результат поиска продемонстрирован на рисунке 2.1. С правой стороны от каждой записи располагаются ссылки на редактирование, отображении, удаление данной записи.

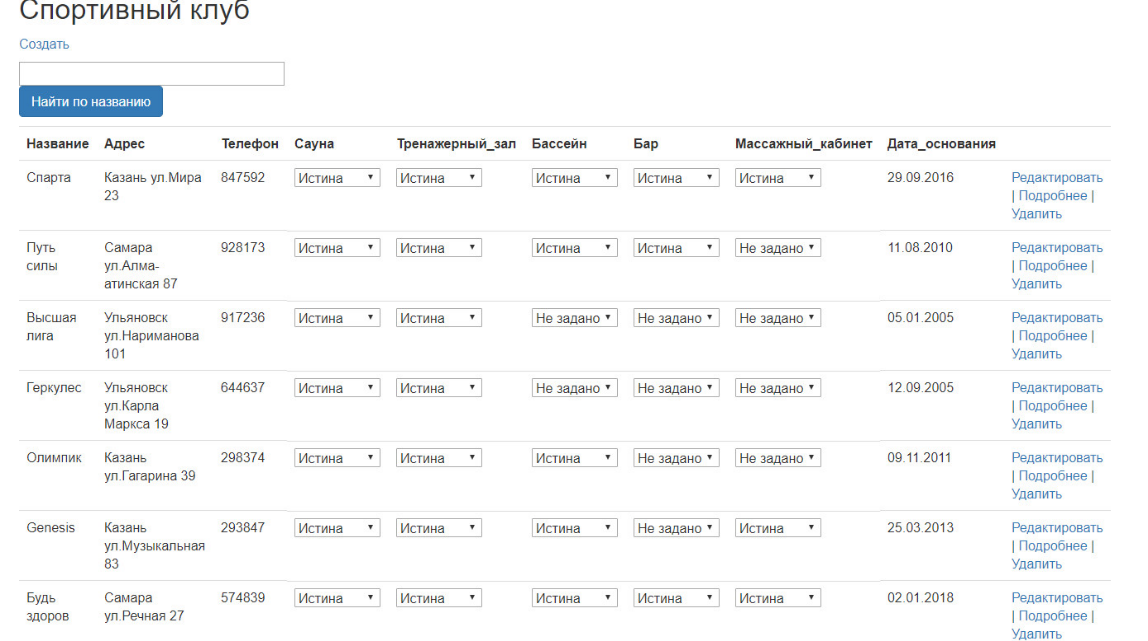


Рисунок 2 - Страница «Спортивный клуб»

Лист

№ докум.

Подпись

Дата

Лист

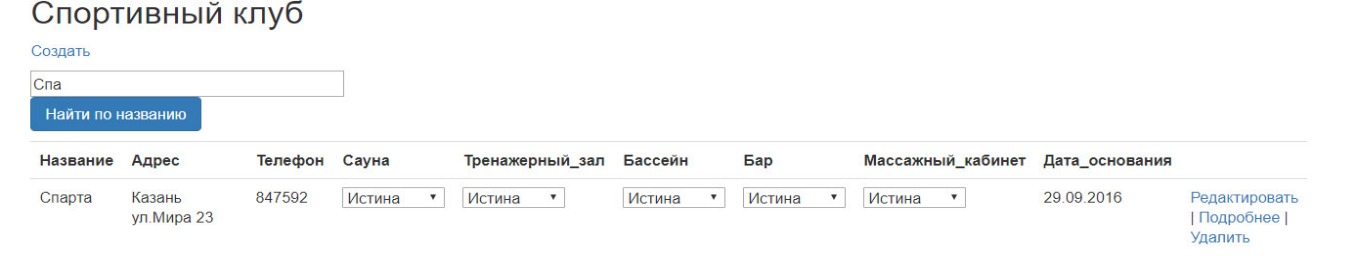


Рисунок 2.1 - Результат поиска по названию

3.При нажатии на ссылку “Создать”, которая находится в верхней части страницы, происходит перенаправление на страницу «Создание» таблицы, в которой вы ранее находились. Для таблицы “Спортивный клуб” процесс создания записи будет выглядеть следующим образом - (рис. 3).

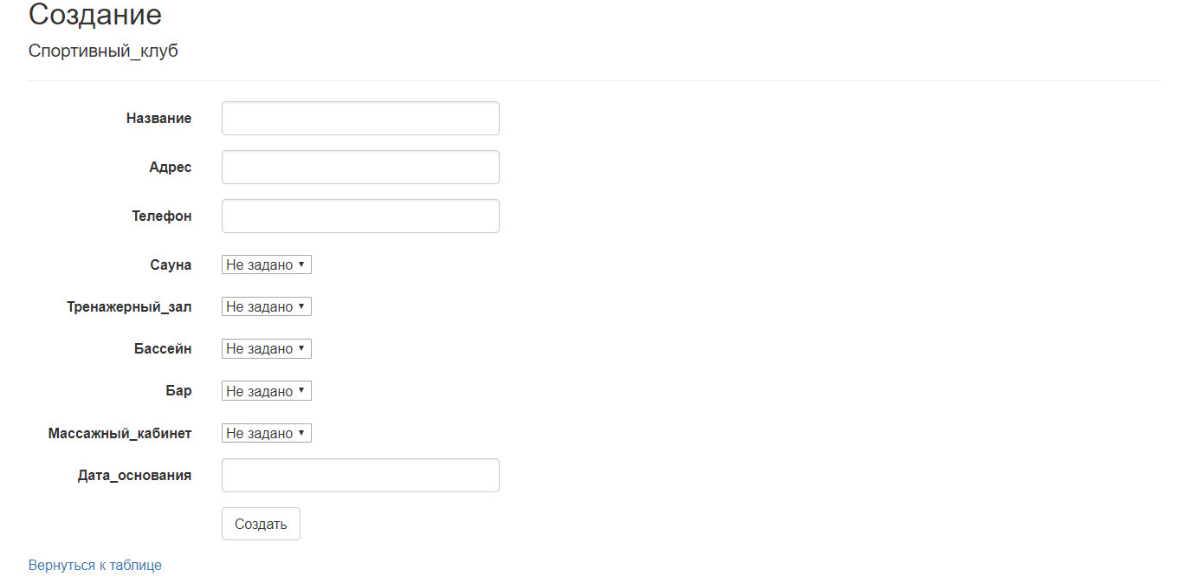
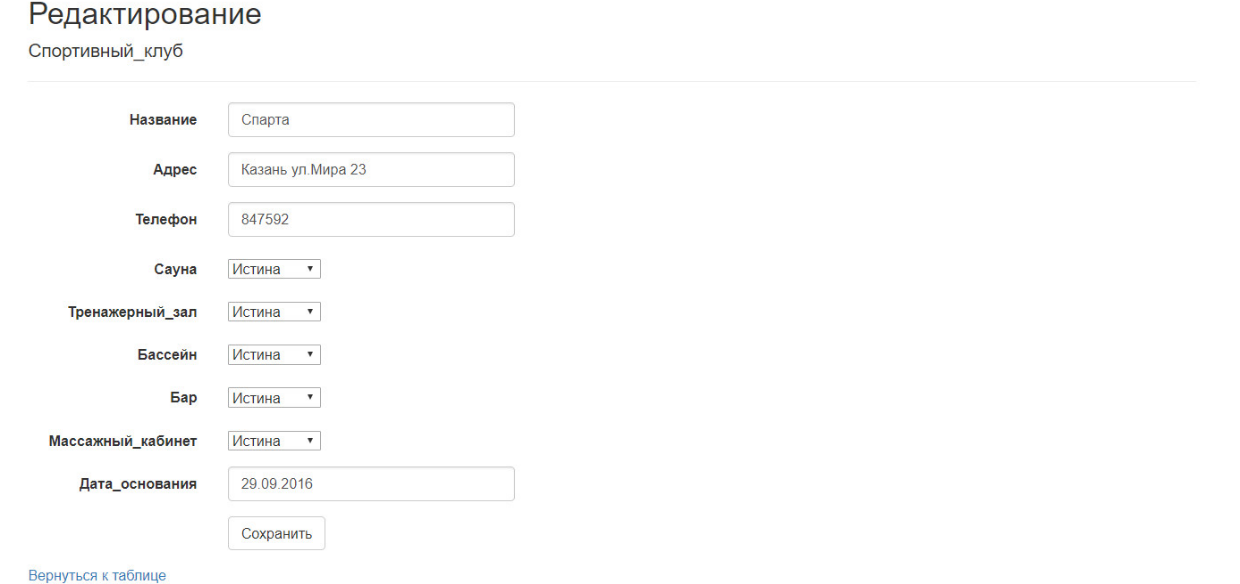


Рисунок 3 - Добавление новой записи в таблицу.

Для остальных таблиц автоматизированной системы добавление записи в таблицу осуществляется аналогичным образом.

4.Страница «Редактирование» таблицы «Спортивный клуб» (рис. 4.). На ней, как и на странице “Подробно”, представлена вся имеющаяся информация о выбранном спортивном клубе, но при этом её можно редактировать. Можно менять как один, так и несколько столбцов. Для сохранения изменений необходимо нажать клавишу “Сохранить”.



Лист

№ докум.

Подпись

Дата

Лист

Рисунок 4 - Изменение записи о спортивном клубе.

Для остальных таблиц автоматизированной системы изменение записи в таблицы осуществляется аналогичным образом.

5.Страница «Удаление» таблицы «Спортивный клуб» (рис. 5). На ней представлена информация о удаляемой записи. Чтобы удалить запись пользователю нужно подтвердить удаление, нажав на кнопку Удалить.

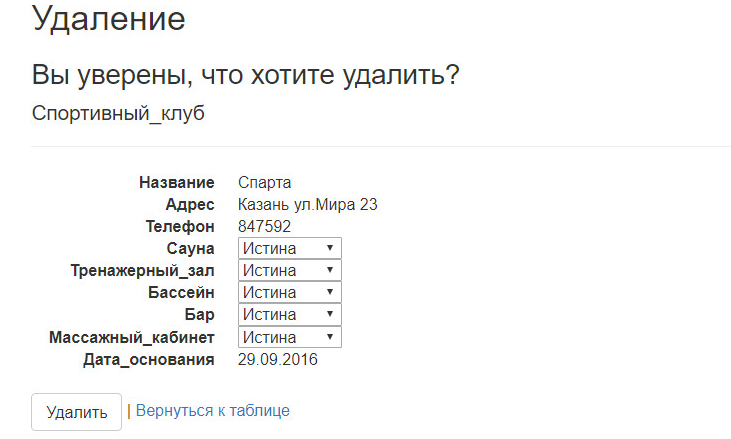


Рисунок 5 - Удаление записи о спортивном клубе.

Для остальных таблиц удаление аналогично.

6.Страница «Подробно» таблицы «Спортивный клуб» (рис 6.) демонстрирует подробную информацию о выбранном спортивном клубе.

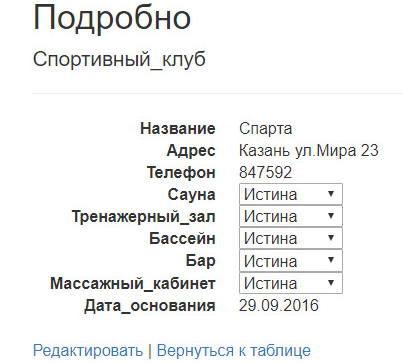


Рисунок 6 - Подробная информация об спортивном клубе.

Для остальных таблиц эта страница выглядит аналогично.

Лист

№ докум.

Подпись

Дата

Лист

7.Страница «Сотрудники» (рис.7). На данной странице отображается вся информация о сотрудниках, которые работают в спортивном клубе. Есть возможность фильтрации по фамилии. Результат фильтрации продемонстрирован на рисунке 7.1.

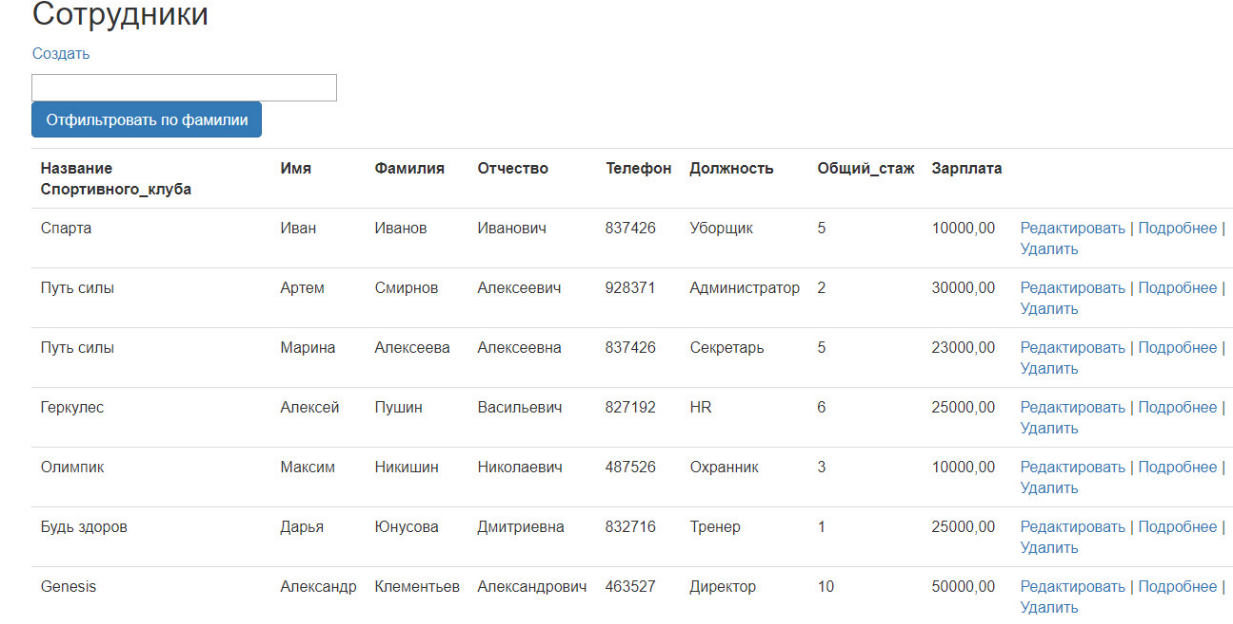


Рисунок 7 - Страница «Сотрудники».

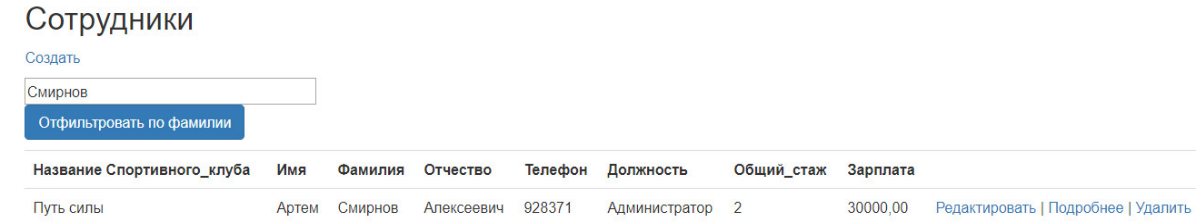


Рисунок 7.1 - Результат фильтрации.

8.Страница «Спортсмен» (рис.8). На данной странице отображается вся информация о спортсменах, которые занимаются в спортивном клубе. Есть возможность фильтрации по гражданству и поиск по фамилии. Результат фильтрации по гражданству показан на рисунке 8.1. Результат поиска продемонстрирован на рисунке 8.2.

Лист

№ докум.

Подпись

Дата

Лист

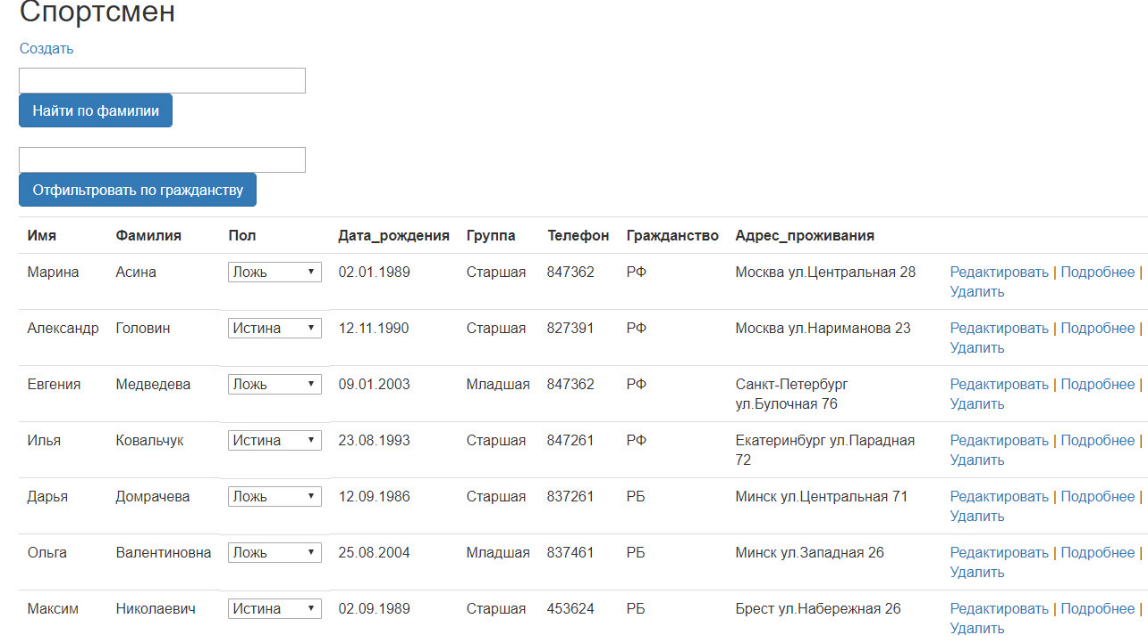


Рисунок 8 - Страница «Спортсмен».



Рисунок 8.1 – Результат фильтрации.

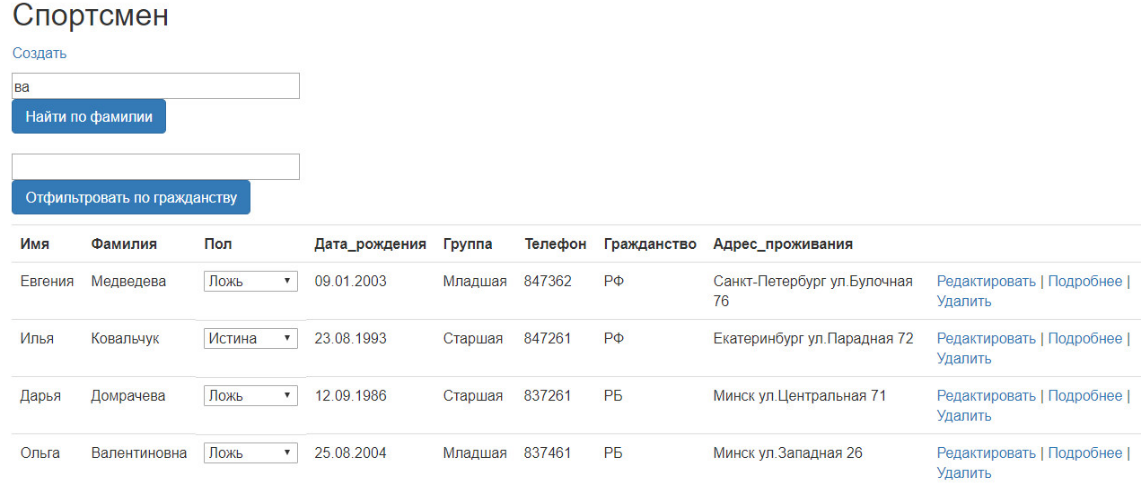


Рисунок 8.2 - Результат поиска.

## 4. 3 Исключительные ситуации

1. Проверка корректности ввода данных пользователем в любую таблицу.

2. Проверка на связанность полей таблицы спортивный клуб с таблицей сотрудники при удалении записи из спортивного клуба.

## Список использованных источников

Лист

№ докум.

Подпись

Дата

Лист

1. Онлайн-книга Изучаем ASP.NET.MVC4 [Электронный ресурс] // Сайт о программировании «Metanit.com». - Электронные данн. - Режим доступа:

<https://metanit.com/sharp/mvc/>

1. Онлайн-книга Работа с Entity Framework 6 [Электронный ресурс] // Сайт о программировании «professorweb.ru». - Электронные данн. - Режим доступа: <https://professorweb.ru/>
2. Начало работы с Entity Framework 6 Code First с помощью MVC 5 [Электронный ресурс] // Сайт «docs.microsoft.com» — Электронные данн. —Режим доступа: <https://docs.microsoft.com/ru-ru/aspnet/mvc/overview/getting-started/getting-started-with-ef-using-mvc/creating-an-entity-framework-data-model-for-an-asp-net-mvc-application>
3. Фильтры в ASP.NET Core [Электронный ресурс] // Сайт «docs.microsoft.com» — Электронные данн. — Режим доступа: <https://docs.microsoft.com/ru-ru/aspnet/core/mvc/controllers/filters?view=aspnetcore-2.2>
4. Физическая культура и спорт как отрасль социальной сферы. [Электронный ресурс] // Президент России - Электронные данн. - Режим доступа: <http://kremlin.ru/events/president/transcripts/21920>
5. Спорт как досуг. [Электронный ресурс] // ИЦРОН - Электронные данн. - Режим доступа: [http://izron.ru/articles/pedagogika-psikhologiya-i-obrazovanie-ot-teorii-k-praktike-sbornik-nauchnykh-trudov-po-itogam-mezhdu/](http://izron.ru/articles/pedagogika-psikhologiya-i-obrazovanie-ot-teorii-k-praktike-sbornik-nauchnykh-trudov-po-itogam-mezhdu/sektsiya-4-teoriya-i-metodika-fizicheskogo-vospitaniya-sportivnoy-trenirovki-ozdorovitelnoy-i-adapti/sport-kak-chast-dosuga-molodezhi/)
6. Система физкультуры и спорта в России [Электронный ресурс] // Теория и практика физической культуры - Электронные данн. - Режим доступа: <http://sportlib.info/Press/TPFK/2007N7/p77-79.htm>
7. Спорт это жизнь [Электронный ресурс] // Пышлицкое - Электронные данн. - Режим доступа:<http://pyshlitskoe.shatura.ru/sport-eto-zhizn-eto-dvizhenie-eto-zdorove.php>

## Приложение А. Исходные тексты программных модулей

HomeController

using System.Web.Mvc;

namespace WebApplication1.Controllers

{

public class HomeController : Controller

{

public ActionResult Index()

{

return View();

}

}

}

СоревнованияController

using System;

using System.Collections.Generic;

using System.Data;

using System.Data.Entity;

using System.Linq;

using System.Net;

using System.Web;

using System.Web.Mvc;

using WebApplication1.Models;

namespace WebApplication1.Controllers

{

public class СоревнованияController : Controller

{

private DBConnection db = new DBConnection();

public ActionResult Index(string filter)

{

if (!String.IsNullOrWhiteSpace(filter))

{

var text = db.Соревнования.AsEnumerable();

text = text.Where(x => x.Город == filter);

return View(text.ToList());

}

return View(db.Соревнования.ToList());

}

// GET: Соревнования

//public ActionResult Index()

//{

// return View(db.Соревнования.ToList());

//}

// GET: Соревнования/Details/5

public ActionResult Details(int? id)

{

if (id == null)

{

return new HttpStatusCodeResult(HttpStatusCode.BadRequest);

}

Соревнования соревнования = db.Соревнования.Find(id);

if (соревнования == null)

{

return HttpNotFound();

}

return View(соревнования);

}

// GET: Соревнования/Create

public ActionResult Create()

{

return View();

}

// POST: Соревнования/Create

// Чтобы защититься от атак чрезмерной передачи данных, включите определенные свойства, для которых следует установить привязку. Дополнительные

// сведения см. в статье https://go.microsoft.com/fwlink/?LinkId=317598.

[HttpPost]

[ValidateAntiForgeryToken]

public ActionResult Create([Bind(Include = "ID\_Соревнований,Дата\_соревнований,Тип\_соревнований,Вид\_спорта,Спонсор\_соревнований,Город,Адрес")] Соревнования соревнования)

{

if (ModelState.IsValid)

{

db.Соревнования.Add(соревнования);

db.SaveChanges();

return RedirectToAction("Index");

}

return View(соревнования);

}

// GET: Соревнования/Edit/5

public ActionResult Edit(int? id)

{

if (id == null)

{

return new HttpStatusCodeResult(HttpStatusCode.BadRequest);

}

Соревнования соревнования = db.Соревнования.Find(id);

if (соревнования == null)

{

return HttpNotFound();

}

return View(соревнования);

}

// POST: Соревнования/Edit/5

// Чтобы защититься от атак чрезмерной передачи данных, включите определенные свойства, для которых следует установить привязку. Дополнительные

// сведения см. в статье https://go.microsoft.com/fwlink/?LinkId=317598.

[HttpPost]

[ValidateAntiForgeryToken]

public ActionResult Edit([Bind(Include = "ID\_Соревнований,Дата\_соревнований,Тип\_соревнований,Вид\_спорта,Спонсор\_соревнований,Город,Адрес")] Соревнования соревнования)

{

if (ModelState.IsValid)

{

db.Entry(соревнования).State = EntityState.Modified;

db.SaveChanges();

return RedirectToAction("Index");

}

return View(соревнования);

}

// GET: Соревнования/Delete/5

public ActionResult Delete(int? id)

{

if (id == null)

{

return new HttpStatusCodeResult(HttpStatusCode.BadRequest);

}

Соревнования соревнования = db.Соревнования.Find(id);

if (соревнования == null)

{

return HttpNotFound();

}

return View(соревнования);

}

// POST: Соревнования/Delete/5

[HttpPost, ActionName("Delete")]

[ValidateAntiForgeryToken]

public ActionResult DeleteConfirmed(int id)

{

Соревнования соревнования = db.Соревнования.Find(id);

db.Соревнования.Remove(соревнования);

db.SaveChanges();

return RedirectToAction("Index");

}

protected override void Dispose(bool disposing)

{

if (disposing)

{

db.Dispose();

}

base.Dispose(disposing);

}

}

}

СотрудникиController

using System;

using System.Collections.Generic;

using System.Data;

using System.Data.Entity;

using System.Linq;

using System.Net;

using System.Web;

using System.Web.Mvc;

using WebApplication1.Models;

namespace WebApplication1.Controllers

{

public class СотрудникиController : Controller

{

private DBConnection db = new DBConnection();

public ActionResult Index(string filter)

{

if (!String.IsNullOrWhiteSpace(filter))

{

var temp = db.Сотрудники.AsEnumerable();

temp = temp.Where(x => x.Фамилия == filter);

return View(temp.ToList());

}

var сотрудники = db.Сотрудники.Include(с => с.Спортивный\_клуб);

return View(сотрудники.ToList());

}

// GET: Сотрудники

//public ActionResult Index()

//{

// var сотрудники = db.Сотрудники.Include(с => с.Спортивный\_клуб);

// return View(сотрудники.ToList());

//}

// GET: Сотрудники/Details/5

public ActionResult Details(int? id)

{

if (id == null)

{

return new HttpStatusCodeResult(HttpStatusCode.BadRequest);

}

Сотрудники сотрудники = db.Сотрудники.Find(id);

if (сотрудники == null)

{

return HttpNotFound();

}

return View(сотрудники);

}

// GET: Сотрудники/Create

public ActionResult Create()

{

ViewBag.ID\_Спорт\_клуба = new SelectList(db.Спортивный\_клуб, "ID\_Спорт\_клуба", "Название");

return View();

}

// POST: Сотрудники/Create

// Чтобы защититься от атак чрезмерной передачи данных, включите определенные свойства, для которых следует установить привязку. Дополнительные

// сведения см. в статье https://go.microsoft.com/fwlink/?LinkId=317598.

[HttpPost]

[ValidateAntiForgeryToken]

public ActionResult Create([Bind(Include = "ID\_Сотрудника,Имя,Фамилия,Отчество,Телефон,Должность,Общий\_стаж,Зарплата,ID\_Спорт\_клуба")] Сотрудники сотрудники)

{

if (ModelState.IsValid)

{

db.Сотрудники.Add(сотрудники);

db.SaveChanges();

return RedirectToAction("Index");

}

ViewBag.ID\_Спорт\_клуба = new SelectList(db.Спортивный\_клуб, "ID\_Спорт\_клуба", "Название", сотрудники.ID\_Спорт\_клуба);

return View(сотрудники);

}

// GET: Сотрудники/Edit/5

public ActionResult Edit(int? id)

{

if (id == null)

{

return new HttpStatusCodeResult(HttpStatusCode.BadRequest);

}

Сотрудники сотрудники = db.Сотрудники.Find(id);

if (сотрудники == null)

{

return HttpNotFound();

}

ViewBag.ID\_Спорт\_клуба = new SelectList(db.Спортивный\_клуб, "ID\_Спорт\_клуба", "Название", сотрудники.ID\_Спорт\_клуба);

return View(сотрудники);

}

// POST: Сотрудники/Edit/5

// Чтобы защититься от атак чрезмерной передачи данных, включите определенные свойства, для которых следует установить привязку. Дополнительные

// сведения см. в статье https://go.microsoft.com/fwlink/?LinkId=317598.

[HttpPost]

[ValidateAntiForgeryToken]

public ActionResult Edit([Bind(Include = "ID\_Сотрудника,Имя,Фамилия,Отчество,Телефон,Должность,Общий\_стаж,Зарплата,ID\_Спорт\_клуба")] Сотрудники сотрудники)

{

if (ModelState.IsValid)

{

db.Entry(сотрудники).State = EntityState.Modified;

db.SaveChanges();

return RedirectToAction("Index");

}

ViewBag.ID\_Спорт\_клуба = new SelectList(db.Спортивный\_клуб, "ID\_Спорт\_клуба", "Название", сотрудники.ID\_Спорт\_клуба);

return View(сотрудники);

}

// GET: Сотрудники/Delete/5

public ActionResult Delete(int? id)

{

if (id == null)

{

return new HttpStatusCodeResult(HttpStatusCode.BadRequest);

}

Сотрудники сотрудники = db.Сотрудники.Find(id);

if (сотрудники == null)

{

return HttpNotFound();

}

return View(сотрудники);

}

// POST: Сотрудники/Delete/5

[HttpPost, ActionName("Delete")]

[ValidateAntiForgeryToken]

public ActionResult DeleteConfirmed(int id)

{

Сотрудники сотрудники = db.Сотрудники.Find(id);

db.Сотрудники.Remove(сотрудники);

db.SaveChanges();

return RedirectToAction("Index");

}

protected override void Dispose(bool disposing)

{

if (disposing)

{

db.Dispose();

}

base.Dispose(disposing);

}

}

}

Спортивный\_клубController

using System;

using System.Collections.Generic;

using System.Data;

using System.Data.Entity;

using System.Linq;

using System.Net;

using System.Web;

using System.Web.Mvc;

using WebApplication1.Models;

namespace WebApplication1.Controllers

{

public class Спортивный\_клубController : Controller

{

private DBConnection db = new DBConnection();

public interface IExceptionFilter

{

void OnException(ExceptionContext filterContext);

}

// GET: Спортивный\_клуб

public ActionResult Index(string search)

{

return View(db.Спортивный\_клуб.Where(x => x.Название.Contains(search) || search == null).ToList());

//return View(db.Спортивный\_клуб.ToList());

}

// GET: Спортивный\_клуб/Details/5

public ActionResult Details(int? id)

{

if (id == null)

{

return new HttpStatusCodeResult(HttpStatusCode.BadRequest);

}

Спортивный\_клуб спортивный\_клуб = db.Спортивный\_клуб.Find(id);

if (спортивный\_клуб == null)

{

return HttpNotFound();

}

return View(спортивный\_клуб);

}

// GET: Спортивный\_клуб/Create

public ActionResult Create()

{

return View();

}

// POST: Спортивный\_клуб/Create

// Чтобы защититься от атак чрезмерной передачи данных, включите определенные свойства, для которых следует установить привязку. Дополнительные

// сведения см. в статье https://go.microsoft.com/fwlink/?LinkId=317598.

[HttpPost]

[ValidateAntiForgeryToken]

public ActionResult Create([Bind(Include = "ID\_Спорт\_клуба,Название,Адрес,Телефон,Сауна,Тренажерный\_зал,Бассейн,Бар,Массажный\_кабинет,Дата\_основания")] Спортивный\_клуб спортивный\_клуб)

{

if (ModelState.IsValid)

{

db.Спортивный\_клуб.Add(спортивный\_клуб);

db.SaveChanges();

return RedirectToAction("Index");

}

return View(спортивный\_клуб);

}

// GET: Спортивный\_клуб/Edit/5

public ActionResult Edit(int? id)

{

if (id == null)

{

return new HttpStatusCodeResult(HttpStatusCode.BadRequest);

}

Спортивный\_клуб спортивный\_клуб = db.Спортивный\_клуб.Find(id);

if (спортивный\_клуб == null)

{

return HttpNotFound();

}

return View(спортивный\_клуб);

}

// POST: Спортивный\_клуб/Edit/5

// Чтобы защититься от атак чрезмерной передачи данных, включите определенные свойства, для которых следует установить привязку. Дополнительные

// сведения см. в статье https://go.microsoft.com/fwlink/?LinkId=317598.

[HttpPost]

[ValidateAntiForgeryToken]

public ActionResult Edit([Bind(Include = "ID\_Спорт\_клуба,Название,Адрес,Телефон,Сауна,Тренажерный\_зал,Бассейн,Бар,Массажный\_кабинет,Дата\_основания")] Спортивный\_клуб спортивный\_клуб)

{

if (ModelState.IsValid)

{

db.Entry(спортивный\_клуб).State = EntityState.Modified;

db.SaveChanges();

return RedirectToAction("Index");

}

return View(спортивный\_клуб);

}

// GET: Спортивный\_клуб/Delete/5

[HandleError(ExceptionType = typeof(DeletedRowInaccessibleException), View = "~/Error")]

public ActionResult Delete(int? id)

{

if (id == null)

{

return new HttpStatusCodeResult(HttpStatusCode.BadRequest);

}

Спортивный\_клуб спортивный\_клуб = db.Спортивный\_клуб.Find(id);

if (спортивный\_клуб == null)

{

return HttpNotFound();

}

return View(спортивный\_клуб);

}

// POST: Спортивный\_клуб/Delete/5

[HttpPost, ActionName("Delete")]

[ValidateAntiForgeryToken]

[HandleError(ExceptionType = typeof(DeletedRowInaccessibleException), View = "~/Error")]

public ActionResult DeleteConfirmed(int id)

{

try {

Спортивный\_клуб спортивный\_клуб = db.Спортивный\_клуб.Find(id);

db.Спортивный\_клуб.Remove(спортивный\_клуб);

db.SaveChanges();

return RedirectToAction("Index");

} catch

{

return Redirect("~/Спортивный\_клуб/Error/");

}

}

public ActionResult Error()

{

return View();

}

protected override void Dispose(bool disposing)

{

if (disposing)

{

db.Dispose();

}

base.Dispose(disposing);

}

}

}

СпортсменController

using System;

using System.Collections.Generic;

using System.Data;

using System.Data.Entity;

using System.Linq;

using System.Net;

using System.Web;

using System.Web.Mvc;

using WebApplication1.Models;

namespace WebApplication1.Controllers

{

public class СпортсменController : Controller

{

private DBConnection db = new DBConnection();

// GET: Спортсмен

public ActionResult Index(string search, string filter)

{

if (!String.IsNullOrWhiteSpace(filter))

{

var temp = db.Спортсмен.AsEnumerable();

temp = temp.Where(x => x.Гражданство == filter);

return View(temp.ToList());

}

return View(db.Спортсмен.Where(x => x.Фамилия.Contains(search) || search == null).ToList());

// return View(db.Спортсмен.ToList());

}

// GET: Спортсмен/Details/5

public ActionResult Details(int? id)

{

if (id == null)

{

return new HttpStatusCodeResult(HttpStatusCode.BadRequest);

}

Спортсмен спортсмен = db.Спортсмен.Find(id);

if (спортсмен == null)

{

return HttpNotFound();

}

return View(спортсмен);

}

// GET: Спортсмен/Create

public ActionResult Create()

{

return View();

}

// POST: Спортсмен/Create

// Чтобы защититься от атак чрезмерной передачи данных, включите определенные свойства, для которых следует установить привязку. Дополнительные

// сведения см. в статье https://go.microsoft.com/fwlink/?LinkId=317598.

[HttpPost]

[ValidateAntiForgeryToken]

public ActionResult Create([Bind(Include = "ID\_Спортсмена,Имя,Фамилия,Пол,Дата\_рождения,Группа,Телефон,Гражданство,Адрес\_проживания")] Спортсмен спортсмен)

{

if (ModelState.IsValid)

{

db.Спортсмен.Add(спортсмен);

db.SaveChanges();

db.Database.ExecuteSqlCommand("EXEC TestAge");

return RedirectToAction("Index");

}

return View(спортсмен);

}

// GET: Спортсмен/Edit/5

public ActionResult Edit(int? id)

{

if (id == null)

{

return new HttpStatusCodeResult(HttpStatusCode.BadRequest);

}

Спортсмен спортсмен = db.Спортсмен.Find(id);

if (спортсмен == null)

{

return HttpNotFound();

}

return View(спортсмен);

}

// POST: Спортсмен/Edit/5

// Чтобы защититься от атак чрезмерной передачи данных, включите определенные свойства, для которых следует установить привязку. Дополнительные

// сведения см. в статье https://go.microsoft.com/fwlink/?LinkId=317598.

[HttpPost]

[ValidateAntiForgeryToken]

public ActionResult Edit([Bind(Include = "ID\_Спортсмена,Имя,Фамилия,Пол,Дата\_рождения,Группа,Телефон,Гражданство,Адрес\_проживания")] Спортсмен спортсмен)

{

if (ModelState.IsValid)

{

db.Entry(спортсмен).State = EntityState.Modified;

db.SaveChanges();

return RedirectToAction("Index");

}

return View(спортсмен);

}

// GET: Спортсмен/Delete/5

public ActionResult Delete(int? id)

{

if (id == null)

{

return new HttpStatusCodeResult(HttpStatusCode.BadRequest);

}

Спортсмен спортсмен = db.Спортсмен.Find(id);

if (спортсмен == null)

{

return HttpNotFound();

}

return View(спортсмен);

}

// POST: Спортсмен/Delete/5

[HttpPost, ActionName("Delete")]

[ValidateAntiForgeryToken]

public ActionResult DeleteConfirmed(int id)

{

Спортсмен спортсмен = db.Спортсмен.Find(id);

db.Спортсмен.Remove(спортсмен);

db.SaveChanges();

return RedirectToAction("Index");

}

protected override void Dispose(bool disposing)

{

if (disposing)

{

db.Dispose();

}

base.Dispose(disposing);

}

}

}

Процедура SQL

USE [Sportclub]

GO

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[TestAge] Script Date: 10.10.2019 8:38:39 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

ALTER PROCEDURE [dbo].[TestAge]

AS BEGIN

DECLARE @id int;

SET @id = 0;

SELECT @id = MAX(ID\_Спортсмена) FROM Спортсмен

DECLARE @age datetime, @year int;

SELECT @age = Спортсмен.Дата\_рождения FROM Спортсмен WHERE ID\_Спортсмена = @id

SET @year = YEAR(@age)

PRINT(@year)

SET @year = 2019 - @year

PRINT(@year)

IF (@year >= 18)

BEGIN

UPDATE Спортсмен SET Группа = 'Старшая' WHERE ID\_Спортсмена = @id;

GOTO BrancheOne;

END;

ELSE

UPDATE Спортсмен SET Группа = 'Младшая' WHERE ID\_Спортсмена = @id;

GOTO BrancheTwo;

BrancheOne:

BEGIN

PRINT ('Добавлен в старшую группу');

SELECT \* FROM Спортсмен WHERE ID\_Спортсмена = @id;

GOTO Branche3

END;

Branche3:

BEGIN

RETURN

END;

BrancheTwo:

BEGIN

PRINT ('Добавлен в младшую группу');

SELECT \* FROM Спортсмен WHERE ID\_Спортсмена = @id;

GOTO Branche3

END;

END;

Views Home Index

@{

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

}

<a href="@Url.Action("Index", "Спортивный\_клуб")">Спортивный клуб</a><br />

<a href="@Url.Action("Index", "Спортсмен")">Спортсмен</a><br />

<a href="@Url.Action("Index", "Сотрудники")">Сотрудники</a><br />

<a href="@Url.Action("Index", "Соревнования")">Соревнования</a><br />

Shared \_Layout

<!DOCTYPE html>

<html>

<head>

<meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>

<meta charset="utf-8" />

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>@ViewBag.Title - My ASP.NET Application</title>

@Styles.Render("~/Content/css")

@Scripts.Render("~/bundles/modernizr")

</head>

<body>

<div class="navbar navbar-inverse navbar-fixed-top">

<div class="container">

<div class="navbar-header">

<button type="button" class="navbar-toggle" data-toggle="collapse" data-target=".navbar-collapse">

<span class="icon-bar"></span>

<span class="icon-bar"></span>

<span class="icon-bar"></span>

</button>

@Html.ActionLink("Главная страница", "Index", "Home", new { area = "" }, new { @class = "navbar-brand" })

</div>

<div class="navbar-collapse collapse">

<ul class="nav navbar-nav">

<li>@Html.ActionLink("Спортивный клуб", "Index", "Спортивный\_клуб")</li>

<li>@Html.ActionLink("Спортсмен", "Index", "Спортсмен")</li>

<li>@Html.ActionLink("Сотрудники", "Index", "Сотрудники")</li>

<li>@Html.ActionLink("Соревнования", "Index", "Соревнования")</li>

</ul>

</div>

</div>

</div>

<div class="container body-content">

@RenderBody()

<hr />

<footer>

<p>&copy; @DateTime.Now.Year - мое приложение ASP.NET</p>

</footer>

</div>

@Scripts.Render("~/bundles/jquery")

@Scripts.Render("~/bundles/bootstrap")

@RenderSection("scripts", required: false)

</body>

</html>

Соревнования Index

@model IEnumerable<WebApplication1.Models.Соревнования>

@{

ViewBag.Title = "Index";

}

<h2>Соревнования</h2>

<p>

@Html.ActionLink("Создать", "Create")

@using (@Html.BeginForm("Index", "Соревнования"))

{

@Html.TextBox("filter", "", new { style = "width:100%" })

<br />

<input type="submit" class="btn btn-primary" value="Отфильтровать по городу" />

}

</p>

<table class="table">

<tr>

<th>

@Html.DisplayNameFor(model => model.Дата\_соревнований)

</th>

<th>

@Html.DisplayNameFor(model => model.Тип\_соревнований)

</th>

<th>

@Html.DisplayNameFor(model => model.Вид\_спорта)

</th>

<th>

@Html.DisplayNameFor(model => model.Спонсор\_соревнований)

</th>

<th>

@Html.DisplayNameFor(model => model.Город)

</th>

<th>

@Html.DisplayNameFor(model => model.Адрес)

</th>

<th></th>

</tr>

@foreach (var item in Model) {

<tr>

<td>

@Html.DisplayFor(modelItem => item.Дата\_соревнований)

</td>

<td>

@Html.DisplayFor(modelItem => item.Тип\_соревнований)

</td>

<td>

@Html.DisplayFor(modelItem => item.Вид\_спорта)

</td>

<td>

@Html.DisplayFor(modelItem => item.Спонсор\_соревнований)

</td>

<td>

@Html.DisplayFor(modelItem => item.Город)

</td>

<td>

@Html.DisplayFor(modelItem => item.Адрес)

</td>

<td>

@Html.ActionLink("Редактировать", "Edit", new { id=item.ID\_Соревнований }) |

@Html.ActionLink("Подробнее", "Details", new { id=item.ID\_Соревнований }) |

@Html.ActionLink("Удалить", "Delete", new { id=item.ID\_Соревнований })

</td>

</tr>

}

</table>

Соревнования Create

@model WebApplication1.Models.Соревнования

@{

ViewBag.Title = "Create";

}

<h2>Создание</h2>

@using (Html.BeginForm())

{

@Html.AntiForgeryToken()

<div class="form-horizontal">

<h4>Соревнования</h4>

<hr />

@Html.ValidationSummary(true, "", new { @class = "text-danger" })

<div class="form-group">

@Html.LabelFor(model => model.Дата\_соревнований, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.Дата\_соревнований, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.Дата\_соревнований, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.Тип\_соревнований, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.Тип\_соревнований, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.Тип\_соревнований, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.Вид\_спорта, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.Вид\_спорта, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.Вид\_спорта, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.Спонсор\_соревнований, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.Спонсор\_соревнований, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.Спонсор\_соревнований, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.Город, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.Город, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.Город, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.Адрес, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.Адрес, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.Адрес, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

<div class="col-md-offset-2 col-md-10">

<input type="submit" value="Создать" class="btn btn-default" />

</div>

</div>

</div>

}

<div>

@Html.ActionLink("Вернуться к таблице", "Index")

</div>

Соревнования Delete

@model WebApplication1.Models.Соревнования

@{

ViewBag.Title = "Delete";

}

<h2>Удаление</h2>

<h3>Вы уверены, что хотите удалить?</h3>

<div>

<h4>Соревнования</h4>

<hr />

<dl class="dl-horizontal">

<dt>

@Html.DisplayNameFor(model => model.Дата\_соревнований)

</dt>

<dd>

@Html.DisplayFor(model => model.Дата\_соревнований)

</dd>

<dt>

@Html.DisplayNameFor(model => model.Тип\_соревнований)

</dt>

<dd>

@Html.DisplayFor(model => model.Тип\_соревнований)

</dd>

<dt>

@Html.DisplayNameFor(model => model.Вид\_спорта)

</dt>

<dd>

@Html.DisplayFor(model => model.Вид\_спорта)

</dd>

<dt>

@Html.DisplayNameFor(model => model.Спонсор\_соревнований)

</dt>

<dd>

@Html.DisplayFor(model => model.Спонсор\_соревнований)

</dd>

<dt>

@Html.DisplayNameFor(model => model.Город)

</dt>

<dd>

@Html.DisplayFor(model => model.Город)

</dd>

<dt>

@Html.DisplayNameFor(model => model.Адрес)

</dt>

<dd>

@Html.DisplayFor(model => model.Адрес)

</dd>

</dl>

@using (Html.BeginForm()) {

@Html.AntiForgeryToken()

<div class="form-actions no-color">

<input type="submit" value="Удалить" class="btn btn-default" /> |

@Html.ActionLink("Вернуться к таблице", "Index")

</div>

}

</div>

Соревнования Edit

@model WebApplication1.Models.Соревнования

@{

ViewBag.Title = "Edit";

}

<h2>Редактирование</h2>

@using (Html.BeginForm())

{

@Html.AntiForgeryToken()

<div class="form-horizontal">

<h4>Соревнования</h4>

<hr />

@Html.ValidationSummary(true, "", new { @class = "text-danger" })

@Html.HiddenFor(model => model.ID\_Соревнований)

<div class="form-group">

@Html.LabelFor(model => model.Дата\_соревнований, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.Дата\_соревнований, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.Дата\_соревнований, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.Тип\_соревнований, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.Тип\_соревнований, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.Тип\_соревнований, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.Вид\_спорта, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.Вид\_спорта, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.Вид\_спорта, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.Спонсор\_соревнований, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.Спонсор\_соревнований, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.Спонсор\_соревнований, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.Город, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.Город, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.Город, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.Адрес, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.Адрес, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.Адрес, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

<div class="col-md-offset-2 col-md-10">

<input type="submit" value="Сохранить" class="btn btn-default" />

</div>

</div>

</div>

}

<div>

@Html.ActionLink("Вернуться к таблице", "Index")

</div>

Соревнования Details

@model WebApplication1.Models.Соревнования

@{

ViewBag.Title = "Details";

}

<h2>Подробно</h2>

<div>

<h4>Соревнования</h4>

<hr />

<dl class="dl-horizontal">

<dt>

@Html.DisplayNameFor(model => model.Дата\_соревнований)

</dt>

<dd>

@Html.DisplayFor(model => model.Дата\_соревнований)

</dd>

<dt>

@Html.DisplayNameFor(model => model.Тип\_соревнований)

</dt>

<dd>

@Html.DisplayFor(model => model.Тип\_соревнований)

</dd>

<dt>

@Html.DisplayNameFor(model => model.Вид\_спорта)

</dt>

<dd>

@Html.DisplayFor(model => model.Вид\_спорта)

</dd>

<dt>

@Html.DisplayNameFor(model => model.Спонсор\_соревнований)

</dt>

<dd>

@Html.DisplayFor(model => model.Спонсор\_соревнований)

</dd>

<dt>

@Html.DisplayNameFor(model => model.Город)

</dt>

<dd>

@Html.DisplayFor(model => model.Город)

</dd>

<dt>

@Html.DisplayNameFor(model => model.Адрес)

</dt>

<dd>

@Html.DisplayFor(model => model.Адрес)

</dd>

</dl>

</div>

<p>

@Html.ActionLink("Редактировать", "Edit", new { id = Model.ID\_Соревнований }) |

@Html.ActionLink("Вернуться к таблице", "Index")

</p>

Сотрудники Index

@model IEnumerable<WebApplication1.Models.Сотрудники>

@{

ViewBag.Title = "Index";

}

<h2>Сотрудники</h2>

<p>

@Html.ActionLink("Создать", "Create")

@using (@Html.BeginForm("Index", "Сотрудники"))

{

@Html.TextBox("filter", "", new { style = "width:100%" })

<br />

<input type="submit" class="btn btn-primary" value="Отфильтровать по фамилии" />

}

</p>

<table class="table">

<tr>

<th>

@\*@Html.DisplayNameFor(model => model.Спортивный\_клуб.Название)\*@

Название Спортивного\_клуба

</th>

<th>

@Html.DisplayNameFor(model => model.Имя)

</th>

<th>

@Html.DisplayNameFor(model => model.Фамилия)

</th>

<th>

@Html.DisplayNameFor(model => model.Отчество)

</th>

<th>

@Html.DisplayNameFor(model => model.Телефон)

</th>

<th>

@Html.DisplayNameFor(model => model.Должность)

</th>

<th>

@Html.DisplayNameFor(model => model.Общий\_стаж)

</th>

<th>

@Html.DisplayNameFor(model => model.Зарплата)

</th>

<th></th>

</tr>

@foreach (var item in Model) {

<tr>

<td>

@Html.DisplayFor(modelItem => item.Спортивный\_клуб.Название)

</td>

<td>

@Html.DisplayFor(modelItem => item.Имя)

</td>

<td>

@Html.DisplayFor(modelItem => item.Фамилия)

</td>

<td>

@Html.DisplayFor(modelItem => item.Отчество)

</td>

<td>

@Html.DisplayFor(modelItem => item.Телефон)

</td>

<td>

@Html.DisplayFor(modelItem => item.Должность)

</td>

<td>

@Html.DisplayFor(modelItem => item.Общий\_стаж)

</td>

<td>

@Html.DisplayFor(modelItem => item.Зарплата)

</td>

<td>

@Html.ActionLink("Редактировать", "Edit", new { id=item.ID\_Сотрудника }) |

@Html.ActionLink("Подробнее", "Details", new { id=item.ID\_Сотрудника }) |

@Html.ActionLink("Удалить", "Delete", new { id=item.ID\_Сотрудника })

</td>

</tr>

}

</table>

Сотрудники Create

@model WebApplication1.Models.Сотрудники

@{

ViewBag.Title = "Create";

}

<h2>Создание</h2>

@using (Html.BeginForm())

{

@Html.AntiForgeryToken()

<div class="form-horizontal">

<h4>Сотрудники</h4>

<hr />

@Html.ValidationSummary(true, "", new { @class = "text-danger" })

<div class="form-group">

@Html.LabelFor(model => model.Имя, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.Имя, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.Имя, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.Фамилия, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.Фамилия, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.Фамилия, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.Отчество, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.Отчество, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.Отчество, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.Телефон, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.Телефон, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.Телефон, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.Должность, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.Должность, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.Должность, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.Общий\_стаж, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.Общий\_стаж, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.Общий\_стаж, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.Зарплата, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.Зарплата, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.Зарплата, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.ID\_Спорт\_клуба, "ID\_Спорт\_клуба", htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.DropDownList("ID\_Спорт\_клуба", null, htmlAttributes: new { @class = "form-control" })

@Html.ValidationMessageFor(model => model.ID\_Спорт\_клуба, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

<div class="col-md-offset-2 col-md-10">

<input type="submit" value="Создать" class="btn btn-default" />

</div>

</div>

</div>

}

<div>

@Html.ActionLink("Вернуться к таблице", "Index")

</div>

Сотрудники Delete

@model WebApplication1.Models.Сотрудники

@{

ViewBag.Title = "Delete";

}

<h2>Удаление</h2>

<h3>Вы уверены, что хотите удалить?</h3>

<div>

<h4>Сотрудники</h4>

<hr />

<dl class="dl-horizontal">

<dt>

@Html.DisplayNameFor(model => model.Спортивный\_клуб.Название)

</dt>

<dd>

@Html.DisplayFor(model => model.Спортивный\_клуб.Название)

</dd>

<dt>

@Html.DisplayNameFor(model => model.Имя)

</dt>

<dd>

@Html.DisplayFor(model => model.Имя)

</dd>

<dt>

@Html.DisplayNameFor(model => model.Фамилия)

</dt>

<dd>

@Html.DisplayFor(model => model.Фамилия)

</dd>

<dt>

@Html.DisplayNameFor(model => model.Отчество)

</dt>

<dd>

@Html.DisplayFor(model => model.Отчество)

</dd>

<dt>

@Html.DisplayNameFor(model => model.Телефон)

</dt>

<dd>

@Html.DisplayFor(model => model.Телефон)

</dd>

<dt>

@Html.DisplayNameFor(model => model.Должность)

</dt>

<dd>

@Html.DisplayFor(model => model.Должность)

</dd>

<dt>

@Html.DisplayNameFor(model => model.Общий\_стаж)

</dt>

<dd>

@Html.DisplayFor(model => model.Общий\_стаж)

</dd>

<dt>

@Html.DisplayNameFor(model => model.Зарплата)

</dt>

<dd>

@Html.DisplayFor(model => model.Зарплата)

</dd>

</dl>

@using (Html.BeginForm()) {

@Html.AntiForgeryToken()

<div class="form-actions no-color">

<input type="submit" value="Удалить" class="btn btn-default" /> |

@Html.ActionLink("Вернуться к таблице", "Index")

</div>

}

</div>

Сотрудники Edit

@model WebApplication1.Models.Сотрудники

@{

ViewBag.Title = "Edit";

}

<h2>Редактирование</h2>

@using (Html.BeginForm())

{

@Html.AntiForgeryToken()

<div class="form-horizontal">

<h4>Сотрудники</h4>

<hr />

@Html.ValidationSummary(true, "", new { @class = "text-danger" })

@Html.HiddenFor(model => model.ID\_Сотрудника)

<div class="form-group">

@Html.LabelFor(model => model.Имя, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.Имя, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.Имя, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.Фамилия, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.Фамилия, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.Фамилия, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.Отчество, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.Отчество, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.Отчество, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.Телефон, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.Телефон, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.Телефон, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.Должность, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.Должность, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.Должность, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.Общий\_стаж, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.Общий\_стаж, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.Общий\_стаж, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.Зарплата, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.Зарплата, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.Зарплата, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.ID\_Спорт\_клуба, "ID\_Спорт\_клуба", htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.DropDownList("ID\_Спорт\_клуба", null, htmlAttributes: new { @class = "form-control" })

@Html.ValidationMessageFor(model => model.ID\_Спорт\_клуба, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

<div class="col-md-offset-2 col-md-10">

<input type="submit" value="Сохранить" class="btn btn-default" />

</div>

</div>

</div>

}

<div>

@Html.ActionLink("Вернуться к таблице", "Index")

</div>

Сотрудники Details

@model WebApplication1.Models.Сотрудники

@{

ViewBag.Title = "Details";

}

<h2>Подробно</h2>

<div>

<h4>Сотрудники</h4>

<hr />

<dl class="dl-horizontal">

<dt>

@Html.DisplayNameFor(model => model.Спортивный\_клуб.Название)

</dt>

<dd>

@Html.DisplayFor(model => model.Спортивный\_клуб.Название)

</dd>

<dt>

@Html.DisplayNameFor(model => model.Имя)

</dt>

<dd>

@Html.DisplayFor(model => model.Имя)

</dd>

<dt>

@Html.DisplayNameFor(model => model.Фамилия)

</dt>

<dd>

@Html.DisplayFor(model => model.Фамилия)

</dd>

<dt>

@Html.DisplayNameFor(model => model.Отчество)

</dt>

<dd>

@Html.DisplayFor(model => model.Отчество)

</dd>

<dt>

@Html.DisplayNameFor(model => model.Телефон)

</dt>

<dd>

@Html.DisplayFor(model => model.Телефон)

</dd>

<dt>

@Html.DisplayNameFor(model => model.Должность)

</dt>

<dd>

@Html.DisplayFor(model => model.Должность)

</dd>

<dt>

@Html.DisplayNameFor(model => model.Общий\_стаж)

</dt>

<dd>

@Html.DisplayFor(model => model.Общий\_стаж)

</dd>

<dt>

@Html.DisplayNameFor(model => model.Зарплата)

</dt>

<dd>

@Html.DisplayFor(model => model.Зарплата)

</dd>

</dl>

</div>

<p>

@Html.ActionLink("Редактировать", "Edit", new { id = Model.ID\_Сотрудника }) |

@Html.ActionLink("Вернуться к таблице", "Index")

</p>

Спортивный клуб Index

@model IEnumerable<WebApplication1.Models.Спортивный\_клуб>

@{

ViewBag.Title = "Index";

}

<h2>Спортивный клуб</h2>

<p>

@Html.ActionLink("Создать", "Create")

@using (@Html.BeginForm("Index", "Спортивный\_клуб"))

{

@Html.TextBox("search", "", new { style = "width:100%" })

<br />

<input type="submit" class="btn btn-primary" value="Найти по названию" />

}

</p>

<table class="table">

<tr>

<th>

@Html.DisplayNameFor(model => model.Название)

</th>

<th>

@Html.DisplayNameFor(model => model.Адрес)

</th>

<th>

@Html.DisplayNameFor(model => model.Телефон)

</th>

<th>

@Html.DisplayNameFor(model => model.Сауна)

</th>

<th>

@Html.DisplayNameFor(model => model.Тренажерный\_зал)

</th>

<th>

@Html.DisplayNameFor(model => model.Бассейн)

</th>

<th>

@Html.DisplayNameFor(model => model.Бар)

</th>

<th>

@Html.DisplayNameFor(model => model.Массажный\_кабинет)

</th>

<th>

@Html.DisplayNameFor(model => model.Дата\_основания)

</th>

<th></th>

</tr>

@foreach (var item in Model) {

<tr>

<td>

@Html.DisplayFor(modelItem => item.Название)

</td>

<td>

@Html.DisplayFor(modelItem => item.Адрес)

</td>

<td>

@Html.DisplayFor(modelItem => item.Телефон)

</td>

<td>

@Html.DisplayFor(modelItem => item.Сауна)

</td>

<td>

@Html.DisplayFor(modelItem => item.Тренажерный\_зал)

</td>

<td>

@Html.DisplayFor(modelItem => item.Бассейн)

</td>

<td>

@Html.DisplayFor(modelItem => item.Бар)

</td>

<td>

@Html.DisplayFor(modelItem => item.Массажный\_кабинет)

</td>

<td>

@Html.DisplayFor(modelItem => item.Дата\_основания)

</td>

<td>

@Html.ActionLink("Редактировать", "Edit", new { id=item.ID\_Спорт\_клуба }) |

@Html.ActionLink("Подробнее", "Details", new { id=item.ID\_Спорт\_клуба }) |

@Html.ActionLink("Удалить", "Delete", new { id=item.ID\_Спорт\_клуба })

</td>

</tr>

}

</table>

Спортивный клуб Create

@model WebApplication1.Models.Спортивный\_клуб

@{

ViewBag.Title = "Create";

}

<h2>Создание</h2>

@using (Html.BeginForm())

{

@Html.AntiForgeryToken()

<div class="form-horizontal">

<h4>Спортивный\_клуб</h4>

<hr />

@Html.ValidationSummary(true, "", new { @class = "text-danger" })

<div class="form-group">

@Html.LabelFor(model => model.Название, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.Название, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.Название, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.Адрес, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.Адрес, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.Адрес, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.Телефон, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.Телефон, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.Телефон, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.Сауна, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

<div class="checkbox">

@Html.EditorFor(model => model.Сауна)

@Html.ValidationMessageFor(model => model.Сауна, "", new { @class = "text-danger" })

</div>

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.Тренажерный\_зал, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

<div class="checkbox">

@Html.EditorFor(model => model.Тренажерный\_зал)

@Html.ValidationMessageFor(model => model.Тренажерный\_зал, "", new { @class = "text-danger" })

</div>

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.Бассейн, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

<div class="checkbox">

@Html.EditorFor(model => model.Бассейн)

@Html.ValidationMessageFor(model => model.Бассейн, "", new { @class = "text-danger" })

</div>

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.Бар, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

<div class="checkbox">

@Html.EditorFor(model => model.Бар)

@Html.ValidationMessageFor(model => model.Бар, "", new { @class = "text-danger" })

</div>

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.Массажный\_кабинет, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

<div class="checkbox">

@Html.EditorFor(model => model.Массажный\_кабинет)

@Html.ValidationMessageFor(model => model.Массажный\_кабинет, "", new { @class = "text-danger" })

</div>

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.Дата\_основания, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.Дата\_основания, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.Дата\_основания, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

<div class="col-md-offset-2 col-md-10">

<input type="submit" value="Создать" class="btn btn-default" />

</div>

</div>

</div>

}

<div>

@Html.ActionLink("Вернуться к таблице", "Index")

</div>

Спортивный клуб Delete

@model WebApplication1.Models.Спортивный\_клуб

@{

ViewBag.Title = "Delete";

}

<h2>Удаление</h2>

<h3>Вы уверены, что хотите удалить?</h3>

<div>

<h4>Спортивный\_клуб</h4>

<hr />

<dl class="dl-horizontal">

<dt>

@Html.DisplayNameFor(model => model.Название)

</dt>

<dd>

@Html.DisplayFor(model => model.Название)

</dd>

<dt>

@Html.DisplayNameFor(model => model.Адрес)

</dt>

<dd>

@Html.DisplayFor(model => model.Адрес)

</dd>

<dt>

@Html.DisplayNameFor(model => model.Телефон)

</dt>

<dd>

@Html.DisplayFor(model => model.Телефон)

</dd>

<dt>

@Html.DisplayNameFor(model => model.Сауна)

</dt>

<dd>

@Html.DisplayFor(model => model.Сауна)

</dd>

<dt>

@Html.DisplayNameFor(model => model.Тренажерный\_зал)

</dt>

<dd>

@Html.DisplayFor(model => model.Тренажерный\_зал)

</dd>

<dt>

@Html.DisplayNameFor(model => model.Бассейн)

</dt>

<dd>

@Html.DisplayFor(model => model.Бассейн)

</dd>

<dt>

@Html.DisplayNameFor(model => model.Бар)

</dt>

<dd>

@Html.DisplayFor(model => model.Бар)

</dd>

<dt>

@Html.DisplayNameFor(model => model.Массажный\_кабинет)

</dt>

<dd>

@Html.DisplayFor(model => model.Массажный\_кабинет)

</dd>

<dt>

@Html.DisplayNameFor(model => model.Дата\_основания)

</dt>

<dd>

@Html.DisplayFor(model => model.Дата\_основания)

</dd>

</dl>

@using (Html.BeginForm()) {

@Html.AntiForgeryToken()

<div class="form-actions no-color">

<input type="submit" value="Удалить" class="btn btn-default" /> |

@Html.ActionLink("Вернуться к таблице", "Index")

</div>

}

</div>

Спортивный клуб Details

@model WebApplication1.Models.Спортивный\_клуб

@{

ViewBag.Title = "Details";

}

<h2>Подробно</h2>

<div>

<h4>Спортивный\_клуб</h4>

<hr />

<dl class="dl-horizontal">

<dt>

@Html.DisplayNameFor(model => model.Название)

</dt>

<dd>

@Html.DisplayFor(model => model.Название)

</dd>

<dt>

@Html.DisplayNameFor(model => model.Адрес)

</dt>

<dd>

@Html.DisplayFor(model => model.Адрес)

</dd>

<dt>

@Html.DisplayNameFor(model => model.Телефон)

</dt>

<dd>

@Html.DisplayFor(model => model.Телефон)

</dd>

<dt>

@Html.DisplayNameFor(model => model.Сауна)

</dt>

<dd>

@Html.DisplayFor(model => model.Сауна)

</dd>

<dt>

@Html.DisplayNameFor(model => model.Тренажерный\_зал)

</dt>

<dd>

@Html.DisplayFor(model => model.Тренажерный\_зал)

</dd>

<dt>

@Html.DisplayNameFor(model => model.Бассейн)

</dt>

<dd>

@Html.DisplayFor(model => model.Бассейн)

</dd>

<dt>

@Html.DisplayNameFor(model => model.Бар)

</dt>

<dd>

@Html.DisplayFor(model => model.Бар)

</dd>

<dt>

@Html.DisplayNameFor(model => model.Массажный\_кабинет)

</dt>

<dd>

@Html.DisplayFor(model => model.Массажный\_кабинет)

</dd>

<dt>

@Html.DisplayNameFor(model => model.Дата\_основания)

</dt>

<dd>

@Html.DisplayFor(model => model.Дата\_основания)

</dd>

</dl>

</div>

<p>

@Html.ActionLink("Редактировать", "Edit", new { id = Model.ID\_Спорт\_клуба }) |

@Html.ActionLink("Вернуться к таблице", "Index")

</p>

Спортивный клуб Edit

@model WebApplication1.Models.Спортивный\_клуб

@{

ViewBag.Title = "Edit";

}

<h2>Редактирование</h2>

@using (Html.BeginForm())

{

@Html.AntiForgeryToken()

<div class="form-horizontal">

<h4>Спортивный\_клуб</h4>

<hr />

@Html.ValidationSummary(true, "", new { @class = "text-danger" })

@Html.HiddenFor(model => model.ID\_Спорт\_клуба)

<div class="form-group">

@Html.LabelFor(model => model.Название, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.Название, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.Название, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.Адрес, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.Адрес, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.Адрес, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.Телефон, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.Телефон, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.Телефон, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.Сауна, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

<div class="checkbox">

@Html.EditorFor(model => model.Сауна)

@Html.ValidationMessageFor(model => model.Сауна, "", new { @class = "text-danger" })

</div>

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.Тренажерный\_зал, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

<div class="checkbox">

@Html.EditorFor(model => model.Тренажерный\_зал)

@Html.ValidationMessageFor(model => model.Тренажерный\_зал, "", new { @class = "text-danger" })

</div>

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.Бассейн, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

<div class="checkbox">

@Html.EditorFor(model => model.Бассейн)

@Html.ValidationMessageFor(model => model.Бассейн, "", new { @class = "text-danger" })

</div>

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.Бар, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

<div class="checkbox">

@Html.EditorFor(model => model.Бар)

@Html.ValidationMessageFor(model => model.Бар, "", new { @class = "text-danger" })

</div>

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.Массажный\_кабинет, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

<div class="checkbox">

@Html.EditorFor(model => model.Массажный\_кабинет)

@Html.ValidationMessageFor(model => model.Массажный\_кабинет, "", new { @class = "text-danger" })

</div>

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.Дата\_основания, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.Дата\_основания, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.Дата\_основания, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

<div class="col-md-offset-2 col-md-10">

<input type="submit" value="Сохранить" class="btn btn-default" />

</div>

</div>

</div>

}

<div>

@Html.ActionLink("Вернуться к таблице", "Index")

</div>

Спортивный клуб Error

@model IEnumerable<WebApplication1.Models.Спортивный\_клуб>

@{

ViewBag.Title = "Error";

}

<h2>Ошибка! Удалите зависимые поля связанной таблицы</h2>

Спортсмен Index

@model IEnumerable<WebApplication1.Models.Спортсмен>

@{

ViewBag.Title = "Index";

}

<h2>Спортсмен</h2>

<p>

@Html.ActionLink("Создать", "Create")

@using (@Html.BeginForm("Index", "Спортсмен"))

{

@Html.TextBox("search", "", new { style = "width:100%" })

<br />

<input type="submit" class="btn btn-primary" value="Найти по фамилии" />

}

<br />

@using (@Html.BeginForm("Index", "Спортсмен"))

{

@Html.TextBox("filter", "", new { style = "width:100%" })

<br />

<input type="submit" class="btn btn-primary" value="Отфильтровать по гражданству" />

}

</p>

<table class="table">

<tr>

<th>

@Html.DisplayNameFor(model => model.Имя)

</th>

<th>

@Html.DisplayNameFor(model => model.Фамилия)

</th>

<th>

@Html.DisplayNameFor(model => model.Пол)

</th>

<th>

@Html.DisplayNameFor(model => model.Дата\_рождения)

</th>

<th>

@Html.DisplayNameFor(model => model.Группа)

</th>

<th>

@Html.DisplayNameFor(model => model.Телефон)

</th>

<th>

@Html.DisplayNameFor(model => model.Гражданство)

</th>

<th>

@Html.DisplayNameFor(model => model.Адрес\_проживания)

</th>

<th></th>

</tr>

@foreach (var item in Model) {

<tr>

<td>

@Html.DisplayFor(modelItem => item.Имя)

</td>

<td>

@Html.DisplayFor(modelItem => item.Фамилия)

</td>

<td>

@Html.DisplayFor(modelItem => item.Пол)

</td>

<td>

@Html.DisplayFor(modelItem => item.Дата\_рождения)

</td>

<td>

@Html.DisplayFor(modelItem => item.Группа)

</td>

<td>

@Html.DisplayFor(modelItem => item.Телефон)

</td>

<td>

@Html.DisplayFor(modelItem => item.Гражданство)

</td>

<td>

@Html.DisplayFor(modelItem => item.Адрес\_проживания)

</td>

<td>

@Html.ActionLink("Редактировать", "Edit", new { id = item.ID\_Спортсмена }) |

@Html.ActionLink("Подробнее", "Details", new { id = item.ID\_Спортсмена }) |

@Html.ActionLink("Удалить", "Delete", new { id = item.ID\_Спортсмена })

</td>

</tr>

}

</table>

Спортсмен Create

@model WebApplication1.Models.Спортсмен

@{

ViewBag.Title = "Create";

}

<h2>Создание</h2>

@using (Html.BeginForm())

{

@Html.AntiForgeryToken()

<div class="form-horizontal">

<h4>Спортсмен</h4>

<hr />

@Html.ValidationSummary(true, "", new { @class = "text-danger" })

<div class="form-group">

@Html.LabelFor(model => model.Имя, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.Имя, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.Имя, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.Фамилия, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.Фамилия, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.Фамилия, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.Пол, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

<div class="checkbox">

@Html.EditorFor(model => model.Пол)

@Html.ValidationMessageFor(model => model.Пол, "", new { @class = "text-danger" })

</div>

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.Дата\_рождения, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.Дата\_рождения, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.Дата\_рождения, "", new { @class = "text-danger" })

</div>

</div>

@\*<div class="form-group">

@Html.LabelFor(model => model.Группа, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.Группа, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.Группа, "", new { @class = "text-danger" })

</div>

</div>\*@

<div class="form-group">

@Html.LabelFor(model => model.Телефон, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.Телефон, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.Телефон, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.Гражданство, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.Гражданство, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.Гражданство, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.Адрес\_проживания, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.Адрес\_проживания, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.Адрес\_проживания, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

<div class="col-md-offset-2 col-md-10">

<input type="submit" value="Создать" class="btn btn-default" />

</div>

</div>

</div>

}

<div>

@Html.ActionLink("Вернуться к таблице", "Index")

</div>

Спортсмен Delete

@model WebApplication1.Models.Спортсмен

@{

ViewBag.Title = "Delete";

}

<h2>Удаление</h2>

<h3>Вы уверены, что хотите удалить?</h3>

<div>

<h4>Спортсмен</h4>

<hr />

<dl class="dl-horizontal">

<dt>

@Html.DisplayNameFor(model => model.Имя)

</dt>

<dd>

@Html.DisplayFor(model => model.Имя)

</dd>

<dt>

@Html.DisplayNameFor(model => model.Фамилия)

</dt>

<dd>

@Html.DisplayFor(model => model.Фамилия)

</dd>

<dt>

@Html.DisplayNameFor(model => model.Пол)

</dt>

<dd>

@Html.DisplayFor(model => model.Пол)

</dd>

<dt>

@Html.DisplayNameFor(model => model.Дата\_рождения)

</dt>

<dd>

@Html.DisplayFor(model => model.Дата\_рождения)

</dd>

<dt>

@Html.DisplayNameFor(model => model.Группа)

</dt>

<dd>

@Html.DisplayFor(model => model.Группа)

</dd>

<dt>

@Html.DisplayNameFor(model => model.Телефон)

</dt>

<dd>

@Html.DisplayFor(model => model.Телефон)

</dd>

<dt>

@Html.DisplayNameFor(model => model.Гражданство)

</dt>

<dd>

@Html.DisplayFor(model => model.Гражданство)

</dd>

<dt>

@Html.DisplayNameFor(model => model.Адрес\_проживания)

</dt>

<dd>

@Html.DisplayFor(model => model.Адрес\_проживания)

</dd>

</dl>

@using (Html.BeginForm()) {

@Html.AntiForgeryToken()

<div class="form-actions no-color">

<input type="submit" value="Удалить" class="btn btn-default" /> |

@Html.ActionLink("Вернуться к таблице", "Index")

</div>

}

</div>

Спортсмен Details

@model WebApplication1.Models.Спортсмен

@{

ViewBag.Title = "Details";

}

<h2>Подробно</h2>

<div>

<h4>Спортсмен</h4>

<hr />

<dl class="dl-horizontal">

<dt>

@Html.DisplayNameFor(model => model.Имя)

</dt>

<dd>

@Html.DisplayFor(model => model.Имя)

</dd>

<dt>

@Html.DisplayNameFor(model => model.Фамилия)

</dt>

<dd>

@Html.DisplayFor(model => model.Фамилия)

</dd>

<dt>

@Html.DisplayNameFor(model => model.Пол)

</dt>

<dd>

@Html.DisplayFor(model => model.Пол)

</dd>

<dt>

@Html.DisplayNameFor(model => model.Дата\_рождения)

</dt>

<dd>

@Html.DisplayFor(model => model.Дата\_рождения)

</dd>

<dt>

@Html.DisplayNameFor(model => model.Группа)

</dt>

<dd>

@Html.DisplayFor(model => model.Группа)

</dd>

<dt>

@Html.DisplayNameFor(model => model.Телефон)

</dt>

<dd>

@Html.DisplayFor(model => model.Телефон)

</dd>

<dt>

@Html.DisplayNameFor(model => model.Гражданство)

</dt>

<dd>

@Html.DisplayFor(model => model.Гражданство)

</dd>

<dt>

@Html.DisplayNameFor(model => model.Адрес\_проживания)

</dt>

<dd>

@Html.DisplayFor(model => model.Адрес\_проживания)

</dd>

</dl>

</div>

<p>

@Html.ActionLink("Редактировать", "Edit", new { id = Model.ID\_Спортсмена }) |

@Html.ActionLink("Вернуться к таблице", "Index")

</p>

Спортсмен Edit

@model WebApplication1.Models.Спортсмен

@{

ViewBag.Title = "Edit";

}

<h2>Редактирование</h2>

@using (Html.BeginForm())

{

@Html.AntiForgeryToken()

<div class="form-horizontal">

<h4>Спортсмен</h4>

<hr />

@Html.ValidationSummary(true, "", new { @class = "text-danger" })

@Html.HiddenFor(model => model.ID\_Спортсмена)

<div class="form-group">

@Html.LabelFor(model => model.Имя, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.Имя, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.Имя, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.Фамилия, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.Фамилия, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.Фамилия, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.Пол, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

<div class="checkbox">

@Html.EditorFor(model => model.Пол)

@Html.ValidationMessageFor(model => model.Пол, "", new { @class = "text-danger" })

</div>

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.Дата\_рождения, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.Дата\_рождения, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.Дата\_рождения, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.Группа, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.Группа, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.Группа, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.Телефон, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.Телефон, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.Телефон, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.Гражданство, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.Гражданство, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.Гражданство, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.Адрес\_проживания, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.Адрес\_проживания, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.Адрес\_проживания, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

<div class="col-md-offset-2 col-md-10">

<input type="submit" value="Сохранить" class="btn btn-default" />

</div>

</div>

</div>

}

<div>

@Html.ActionLink("Вернуться к таблице", "Index")

</div>