**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ**

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ**

**УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ**

**УНИВЕРСИТЕТ ТЕХНОЛОГИЙ И УПРАВЛЕНИЯ ИМЕНИ К.Г. РАЗУМОВСКОГО**

**(ПЕРВЫЙ КАЗАЧИЙ УНИВЕРСИТЕТ)»**

**(ФГБОУ ВО «МГУТУ ИМ. К.Г. РАЗУМОВСКОГО (ПКУ)»)**

**УНИВЕРСИТЕТСКИЙ КОЛЛЕДЖ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ**

**КУРСОВОЙ ПРОЕКТ**

по междисциплинарному курсу: МДК.02.02. Технология разработки и защиты баз данных.

на тему: Разработка информационной системы для

любителя кино..

студента группы ПКС-305

специальности 09.02.03 Программирование в компьютерных системах

Тешаева Рустама Мирзодавлатовича

Студент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Р.М. Тешаев

Руководитель курсового проекта \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Е.А. Ларионова

Председатель ПЦК специальности

09.02.03 Программирование в

компьютерных системах \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ А.И. Глускер

Дата защиты «\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_2020 г.

Оценка: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Заведующий отделением №3 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Е.Л. Гусева

**СОДЕРЖАНИЕ**

[СОДЕРЖАНИЕ 2](#_Toc39413617)

[ВВЕДЕНИЕ 3](#_Toc39413618)

[1 ОБЩАЯ ЧАСТЬ 5](#_Toc39413619)

[1.1 Назначение и цели создания системы 5](#_Toc39413620)

[1.2 Обзор и анализ предметной области 5](#_Toc39413621)

[1.3 Жизненный цикл базы данных 6](#_Toc39413622)

[1.4 Выбор и характеристика СУБД 7](#_Toc39413623)

[1.5 Выбор и характеристика среды разработки приложения 7](#_Toc39413624)

[2 СПЕЦИАЛЬНАЯ ЧАСТЬ 8](#_Toc39413625)

[2.1 Постановка задачи 8](#_Toc39413626)

[2.2 Архитектура информационной системы 8](#_Toc39413627)

[2.3 Логическая модель базы данных 9](#_Toc39413628)

[2.4 Нормализация таблиц 10](#_Toc39413629)

[2.5 Описание таблиц 11](#_Toc39413630)

[2.6 Другие объекты базы данных 13](#_Toc39413631)

[2.7 Разработка приложения 13](#_Toc39413632)

[2.7.4 Тестирование приложения 15](#_Toc39413633)

[2.8 Инструкция пользователю 16](#_Toc39413634)

[ЗАКЛЮЧЕНИЕ 20](#_Toc39413635)

[СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ 21](#_Toc39413636)

[ПРИЛОЖЕНИЕ А SQL Server скрипты 22](#_Toc39413637)

[ПРИЛОЖЕНИЕ Б Код приложения 29](#_Toc39413638)

# ВВЕДЕНИЕ

В настоящее время тема кино интересна, как никогда. Все больше различных книг-бестселлеров, различных видеоигр и автобиографий подаются экранизациям. Людям интересна кинотематика, поэтому они ходят в кинотеатры, смотрят дома кино, также у большинства людей есть как раз таки специальные программы которые посвящены кино тематике.

Объектом исследования является программа информационной системы любителя кино.

Основная цель курсовой работы – это приобретение и закрепление таких навыков, как:

1. закрепление знаний SQL Server при работе с ним в различных СУБД,
2. изучение работы с СУБД SQL Server через Entity Framework.

Основные задачи проекта:

1. анализ предметной области,
2. проектирование базы данных в ERAssistant,
3. реализация базы данных в СУБД SQL Server,
4. разработка представлений и хранимых процедур,
5. разработка моделей,
6. тестирование приложения.

Используемые методы при разработке приложения:

1. анализ,
2. моделирование,
3. реализация программы с помощью объектно-ориентированных принципов.

Структура курсового проекта:

* титульный лист,
* содержание,
* введение,
* общая часть,
* специальная часть,
* заключение,
* список используемой литературы,
* приложения.

Содержание включает в себя названия разделов и номера страниц этих разделов.

Введение описывает актуальность и значимость темы. Обозначает цели и задачи, поставленные перед написанием работы. С этого раздела начинается курсовой проект, так как в последствии на основании него можно будет сделать выводы о том, что удалось выполнить, а что не удалось.

В общую часть входят анализ предметной области, описывающий предметную область, для которой разрабатывается приложение и раздел спецификации, описывающий формирование структуры программного продукта, раздел программы.

В специальную часть входят раздел реализации программного изделия на языке программирования, раздел тестирования программного продукта, выявляющего степени соответствия готового программного продукта со спецификацией.

Заключение подводит итог по проделанной работе. Раздел демонстрирует выводы о выполнении целей и задач, указанных в введении, и трудностях, возникших при выполнении работы.

В списке используемых источников приводится перечень используемых источников, которые использовались при разработке приложения.

1. ОБЩАЯ ЧАСТЬ

## Назначение и цели создания системы

Назначение системы – создание приложения для автоматизации работы парикмахерской.

Цель – реализация учебного плана и формирование профессиональных компетенций по модулю ПМ.02. Разработка и администрирование баз данных таких как:

* разрабатывать объекты базы данных,
* реализовать базу данных в конкретной системе управления базами данных,
* решать вопросы администрирования базы данных,
* реализовывать методы и технологии защиты информации в базе данных.

## Обзор и анализ предметной области

База данных создается для информационного обслуживания любителей кино. БД должна содержать данные о фильмах, о касте, отзывы о фильмах.

В соответствии с предметной областью система строится с учётом следующих особенностей:

у каждого фильма должно быть описание

описание каста фильма

отзыв о фильме

Выделим базовые сущности этой предметной области:

Название фильма, описание каста фильма, отзыв о фильме.

## Жизненный цикл базы данных

Жизненный цикл базы данных - это процедура создания концептуальной схемы базы данных, определения данных, включаемых в базу данных создание программ обновления и обработки данных.

Жизненный цикл базы данных проходит несколько этапов:

* анализ. На данном этапе происходит анализ информации, необходимой для проектирования базы данных.
* проектирование. На этом этапе отображается словесное и естественное описание предметной области в схеме внутренней модели базы данных.
* реализация. На данном этапе концептуальная модель базы данных преобразуется в проект реализации базы данных, создается словарь базы данных, база данных заполняется данными, создаются прикладные программы.
* тестирование. На данном этапе база данных проверяется на корректность выполнения функций. Важно правильно подобрать данные для успешного тестирования.
* эксплуатация. На данном этапе база данных сопровождается наблюдением и поддержкой нормального функционирования.

## Выбор и характеристика СУБД

В качестве СУБД для хранения всей информации был выбран SQL Server.

Особенности:

* независимость от конкретных СУБД;
* переносимость с одной вычислительной системы на другую;
* наличие стандартов;
* реляционная основа;
* высокоуровневая структура;

Недостатки:

* допущение строк-дубликатов в таблицах и результатах выборок, что в рамках реляционной модели данных невозможно и недопустимо;
* поддержка неопределённых значений NULL, создающую фактически многозначную логику;

## Выбор и характеристика среды разработки приложения

В качестве среды разработки был выбран Visual Studio, по причине ряда преимуществ:

* встроенный тестировщик и отладчик,
* работа с Git репозиторием,
* привычное использование.

Для разработки приложения был выбран язык программирования С# из-за ряда преимуществ:

* безопасный код,
* опыт работы с ним.

1. **СПЕЦИАЛЬНАЯ ЧАСТЬ**

## Постановка задачи

Спроектировать средствами SQL Server базу данных и создать средствами С#, позволяющее автоматизировать работу парикмахерской. В базе должна храниться информация о клиентах, мастерах, услугах и истории приемов. При регистрации, пользователь должен ввести свои инициалы и номер телефона, выбрать нужную услугу и мастера, время и дату.

## Архитектура информационной системы

Трехзвенная архитектура – архитектурная модель программного комплекса, предполагающая наличие в нём трёх компонентов: клиента, сервера приложений и сервера баз данных. Рисунок (1).

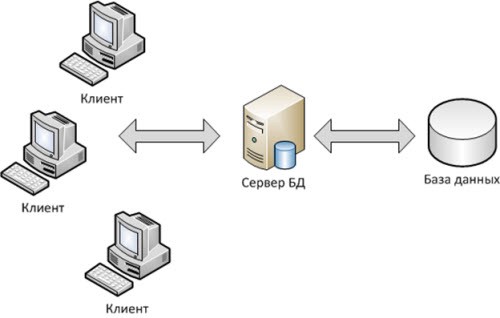


Рисунок 1 – Трехзвенная

Клиентский слой — интерфейс пользователя. Это может быть приложение, подключенное к серверу. Главное, чтобы с его помощью пользователь мог отправлять запросы на сервер и обрабатывать его ответы.

Слой логики — сервер, на котором происходит обработка запросов/ответов. Часто его еще называют серверным слоем. Также здесь происходят все логические операции: математические расчеты, операции с данными, обращения к другим сервисам или хранилищам данных.

Слой данных — сервер баз данных: к нему обращается сервер. В этом слое сохраняется вся необходимая информация, которой пользуется приложение при работе.

В простейшей конфигурации физически сервер приложений может быть совмещён с сервером базы данных на одном компьютере, к которому по сети подключается один или несколько терминалов.

Преимущества:

* возможность построить защиту от SQL-инъекций,
* возможность модифицировать данные перед отправкой клиенту,
* возможность расширить наше приложение на несколько серверов,
* высокая надежность,
* низкие требования к скорости канала между терминалом и сервером приложений,
* низкие требования к производительности и техническим характеристикам терминалов, как следствие снижение их стоимости.

Недостатки:

* более высокая сложность создания приложений,
* сложнее в разворачивании и администрировании,
* высокие требования к производительности серверов приложений и сервера базы данных, и высокая стоимость серверного оборудования,
* высокие требования к скорости канала между сервером базы данных и серверами приложений.

## Логическая модель базы данных

Логическая модель базы данных (Рисунок 2).

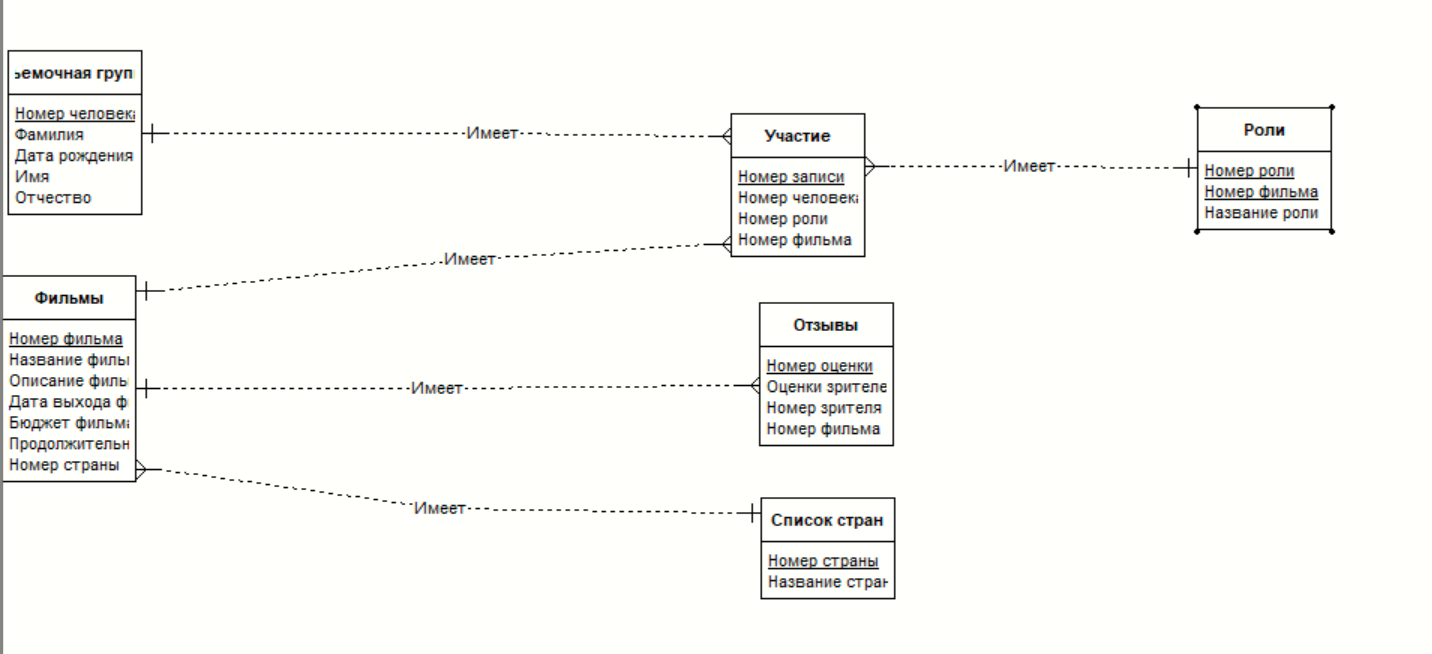


Рисунок 2 – Логическая модель базы данных

## Нормализация таблиц

Первая нормальная форма – отношение называется нормализованным, если все его атрибуты просты, далее – неделимы.

Вторая нормальная форма – отношение находится во второй нормальной форме, если оно соответствует требованиям первой нормальной формы и каждый не ключевой атрибут функционально зависит от первичного ключа.

Третья нормальная форма - отношение находится в третьей нормальной форме, если оно соответствует требованиям второй нормальной формы и каждый не ключевой атрибут не транзитивно зависит от первичного ключа.

Все таблицы приведены к третьей нормальной форме. Нормализация таблиц (Таблица 1).

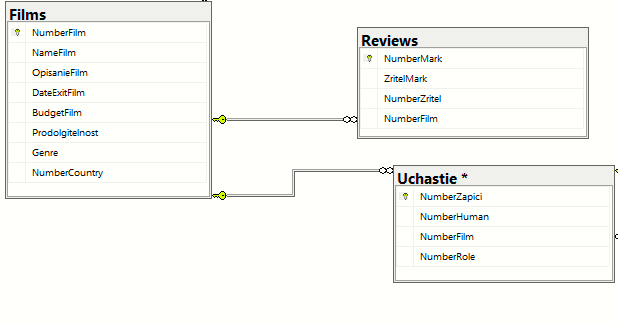


Рисунок 3 – пример третьей нормальной формы

## Описание таблиц

Описание таблиц (Таблица 1, 2, 3)

Таблица 1 – Films

|  |  |  |  |
| --- | --- | --- | --- |
| **Название столбца** | **Описание** | **Тип** | **Примечание** |
| NumberFilm | Номер фильма | int | РК(Первичный ключ) |
| NameFilm | Название фильма | varchar(20) |  |
| OpisanieFilm | Описание фильма | varchar(500) |  |
| DateExitFilm | Дата выхода фильма | char(20) |  |
| BudgetFilm | Бюджет фильма | char(20) |  |
| Prodolgitelnost | Продолжительность | char(20) |  |
| Genre | Жанр | varchar(20) |  |
| NumberCounrty | Номер страны | int |  |

Таблица 2 – Crew

|  |  |  |  |
| --- | --- | --- | --- |
| **Название столбца** | **Описание** | **Тип** | **Примечание** |
| NumberHuman | Номер человека | int | РК(Первичный ключ) |
| lastname | Фамилия | varchar(20) |  |
| Name | Имя | varchar(20) |  |
| Patronomyc | Отчество | varchar(20) |  |
| DateBithday | Дата рождения | char(20) |  |

Таблица 3 – list of countries

|  |  |  |  |
| --- | --- | --- | --- |
| **Название столбца** | **Описание** | **Тип** | **Примечание** |
| NumberCountry | Номер страны | int | РК(Первичный ключ) |
| NameCountry | Название страны | int |  |

Таблица 4 – Reviews

|  |  |  |  |
| --- | --- | --- | --- |
| **Название столбца** | **Описание** | **Тип** | **Примечание** |
| NumberMark | Номер оценки | int | РК(Первичный ключ) |
| ZritelMark | Оценка зрителя | char(20) |  |
| NumberZritel | Номер зрителя | char(20) |  |
| NumberFilm | Номер фильма | int |  |

Таблица 5 – Roles

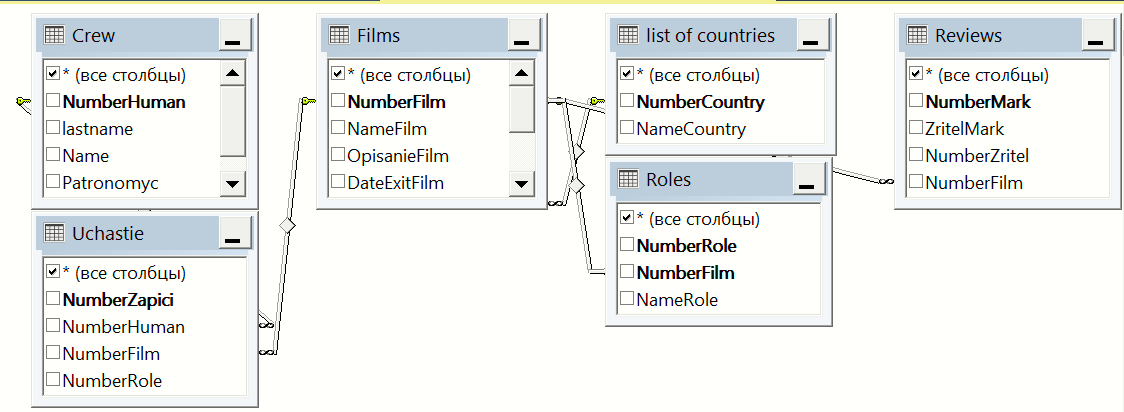
|  |  |  |  |
| --- | --- | --- | --- |
| **Название столбца** | **Описание** | **Тип** | **Примечание** |
| NumberRole | Номер роли | char(20) | РК(Первичный ключ) |
| NunberFilm | Номер фильма | int | РК(Первичный ключ) |
| NameRole | Название роли | varchar(80) |  |

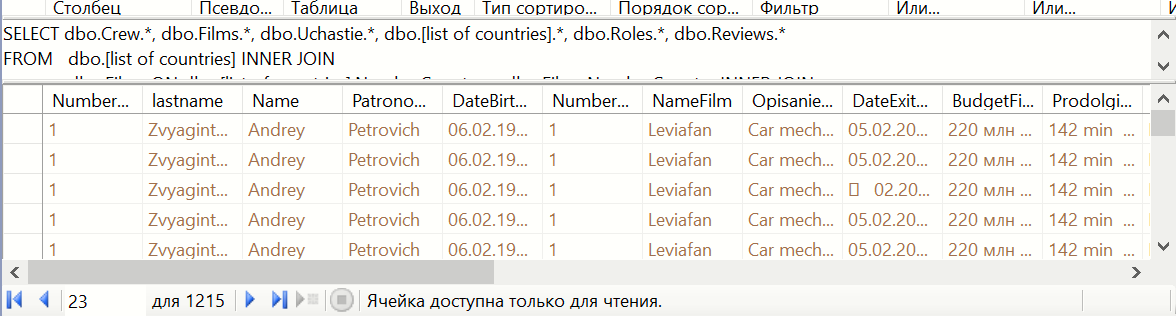
Таблица 6 – Uchastie

|  |  |  |  |
| --- | --- | --- | --- |
| **Название столбца** | **Описание** | **Тип** | **Примечание** |
| NumberZapici | Номер записи | int | РК(Первичный ключ) |
| NunberHuman | Номер человека | int |  |
| NumberFilm | Название фильма | int |  |
| NumberRole | Номер роли | char(20) |  |

## Другие объекты базы данных

Другие объекты базы данных (Рисунок 4)





## Разработка приложения

* + 1. **Диаграмма вариантов использования**

Диаграмма вариантов использования (Рисунок 3).

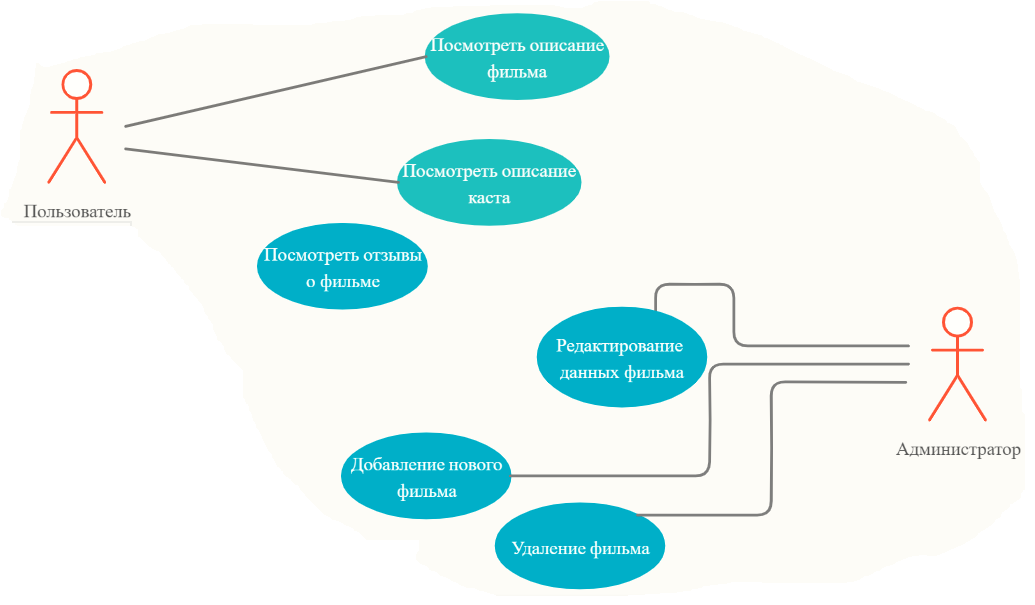


Рисунок 3 – Диаграмма вариантов использования

* + 1. **Архитектура приложения**

Технология ASP.NET использует многоуровневую архитектуру. Приложение написано на языке программирования C#, используя паттерн MVC. **MVC** — **это** **паттерн** проектирования веб-приложений, который включает в себя несколько более мелких шаблонов.

На рисунке (Рисунок 4) показаны уровни, на которых построена работа приложения ASP.NET (.NET Framework) с использованием паттерна MVC:

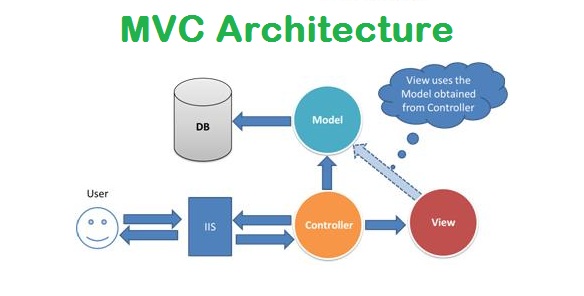


Рисунок 4 – Архитектура ASP.NET (.NET Framework) MVC .

Ниже описаны ключевые компоненты архитектуры ASP.NET (.NET Framework) MVC:

Модели, содержащие или представляющие данные, с которыми работают пользователи. Они могут быть простыми моделями представлений, которые только представляют данные, передаваемые между представлениями и контроллерами, или же они могут быть моделями предметной области, которые содержат бизнес-данные, а также операции, преобразования и правила для манипулирования этими данными.

Представления, применяемые для визуализации некоторой части модели в виде пользовательского интерфейса.

Контроллеры, которые обрабатывают поступающие запросы, выполняют операции с моделью и выбирают представления для визуализации пользователю.

MVC, расшифровка аббревиатуры на русском языке, модели, представления и контроллеры.

* + 1. **Решение главной задачи проекта**

Главная задача проекта – предусмотреть чтобы пользователь мог просматривать информацию о фильмах, а администратор редактировать, добавлять и удалять фильмы.

Удаление фильма

// GET: Home/Delete

public ActionResult Delete(int id)

{

var filmsToDelete = (from m in \_db.Films

where m.NumberFilm == id

select m).First();

return View(filmsToDelete);

}

// POST: Home/Delete

[HttpPost]

public ActionResult Delete(int id, FormCollection collection)

{

var filmsToDelete = (from m in \_db.Films

where m.NumberFilm == id

select m).First();

try

{

\_db.Films.Remove(filmsToDelete);

\_db.SaveChanges();

// TODO: Add delete logic here

return RedirectToAction("Index");

}

catch

{

return View(filmsToDelete);

}

Редактирование фильма

// GET: Home/Edit/

public ActionResult Edit(int id)

{

var filmsToEdit = (from m in \_db.Films

where m.NumberFilm == id

select m).First();

return View(filmsToEdit);

}

// POST: Home/Edit/

[HttpPost]

public ActionResult Edit(Films filmsToEdit)

{

var originalFilms = (from m in \_db.Films

where m.NumberFilm == filmsToEdit.NumberFilm

select m).First();

if (!ModelState.IsValid)

return View(originalFilms);

\_db.Entry(originalFilms).CurrentValues.SetValues(filmsToEdit);

\_db.SaveChanges();

return RedirectToAction("Index");

}

Добавление фильма

// GET: Home/Create

public ActionResult Create()

{

return View();

}

// POST: Home/Create

[HttpPost]

public ActionResult Create([Bind(Exclude = "ID")] Films filmsToCreate)

{

if (!ModelState.IsValid)

return View();

\_db.Films.Add(filmsToCreate);

\_db.SaveChanges();

return RedirectToAction("Index");

}

* + 1. Тестирование прил**ожения**

Тестирование приложения (Таблица 9).

Таблица 13 – Методы проверки требований к приложению

|  |  |  |
| --- | --- | --- |
| **№** | **Действие** | **Результат** |
| 1 | Запуск приложения | Приложение подключается к базе данных |
| 2 | Пользователь | Видит содержимое без необходимости авторизации |
| 3 | Администратор | В данном приложении, пользователь и есть администратор, он добавляет, редактирует и удаляет все данные |
| 4 | Удаление фильма | При нажатии на ссылку, появится веб-страница, с соответствующим вопросом, пользователь может удалить фильм или же вернутся к списку нажав на ссылку |
| 6 | Добавление | При нажатии на ссылку, появится веб-страница. Здесь идет заполнение пустых полей и создание фильма, принцип тот же, что и у удаления, пользователь может, как создать фильм, так и вернуться назад к списку фильмов |
| 7 | Редактирование | При нажатии на ссылку, появится веб-страница. Здесь можно изменять любые данные и как уже было сказано раньше, принцип тот же, если никаких действий не нужно предпринимать, вы можете просто вернуться назад к спику фильмов |

* + 1. **Защита информационной системы**

Для защиты информационной системы будут использоваться следующие методы:

* ограничение прав доступа к объектам базы данных,
* хеширование паролей в таблицах базы данных.

## Инструкция пользователю

* + 1. **Общие сведения об информационной системе**

Приложение предназначено для личного использования, для просмотра, создания, редактирования и удаления фильмов.

* + 1. **Требования к техническим средствам**

Рекомендуемые системные требования к компьютеру:

* процессор Intel Core i3 и остальные процессоры,
* оперативная память не менее 80мб,
* HDD или SSD-диск объемом не менее 500мб,
* клавиатуру,
* монитор с разрешением не менее 1366x766 пикселей.
  + 1. **Требования к программным средствам**

Требования к программным средствам клиента:

* копия операционной системы Windows 10
* поддержка .NET Framework.
  + 1. **Интерфейс приложения**

Главная страница (Рисунок 5).

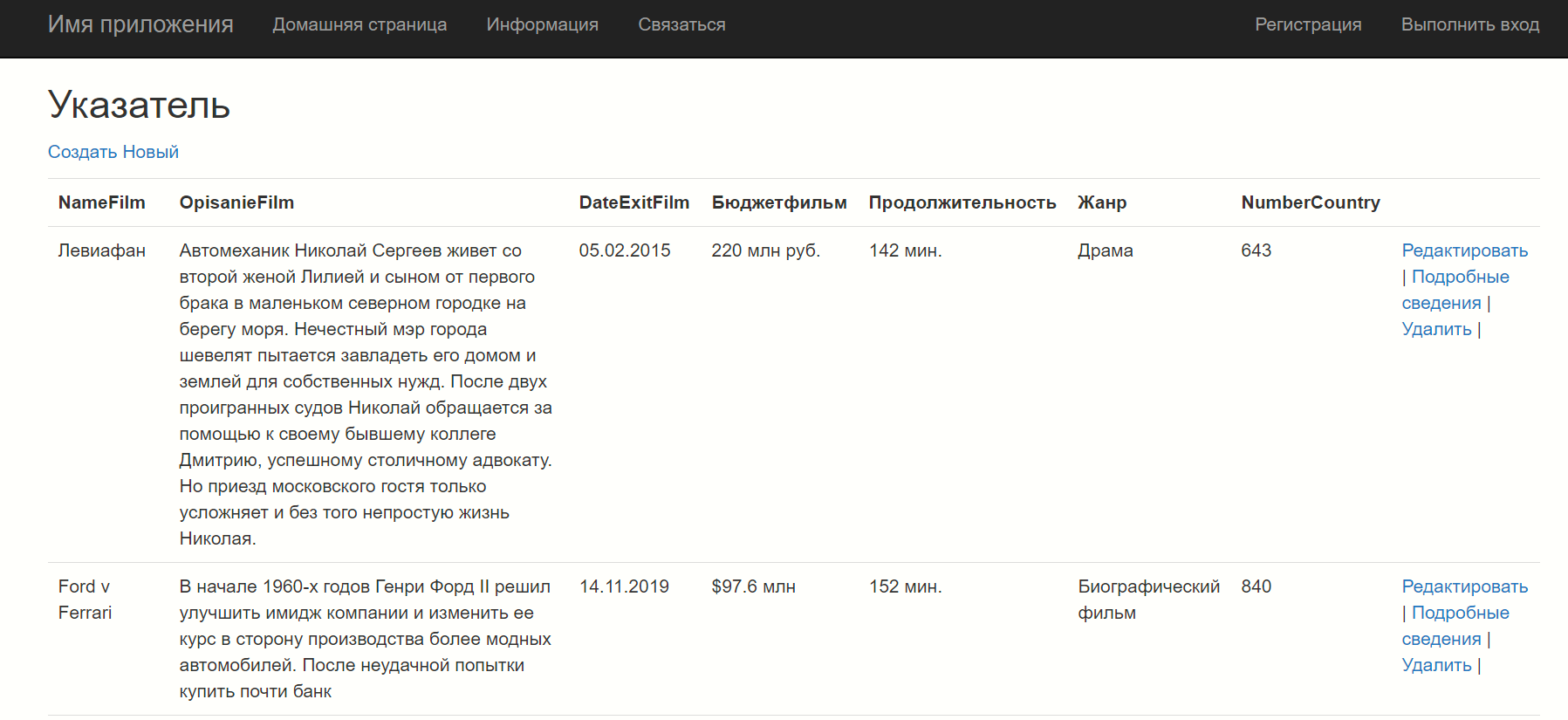
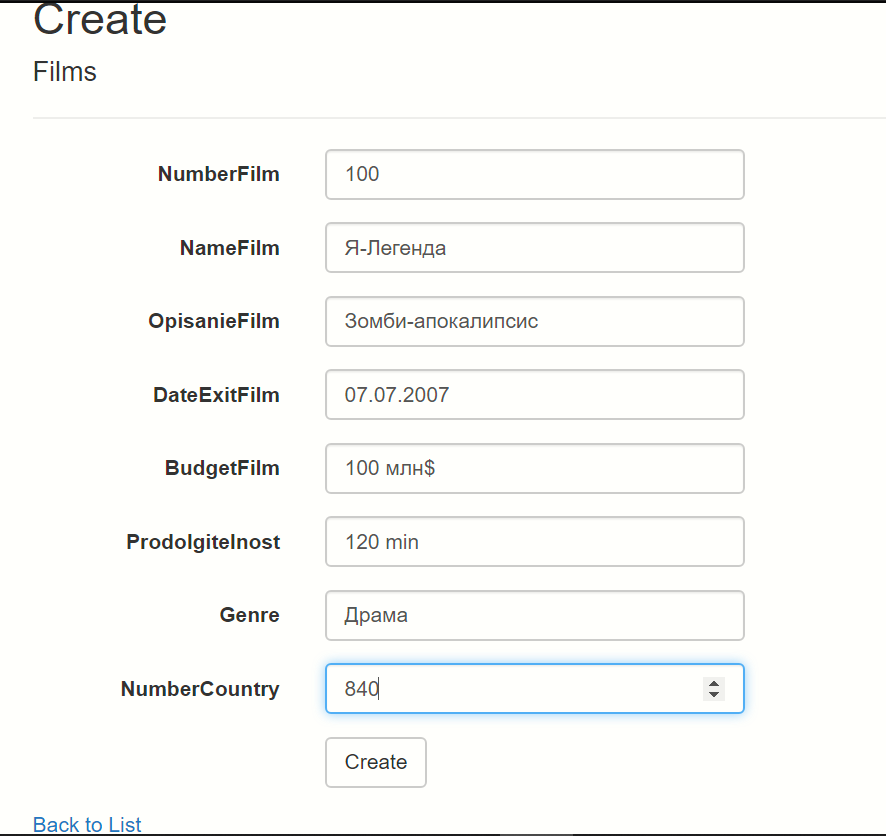


Рисунок 5 – Главный страница

Добавление фильма (Рисунок 6).



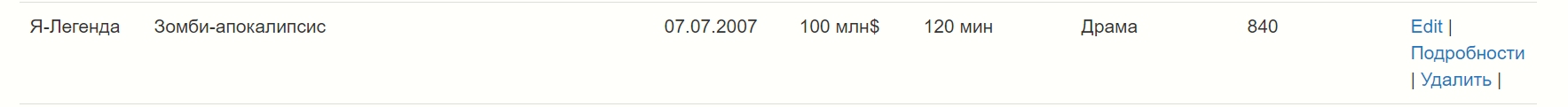
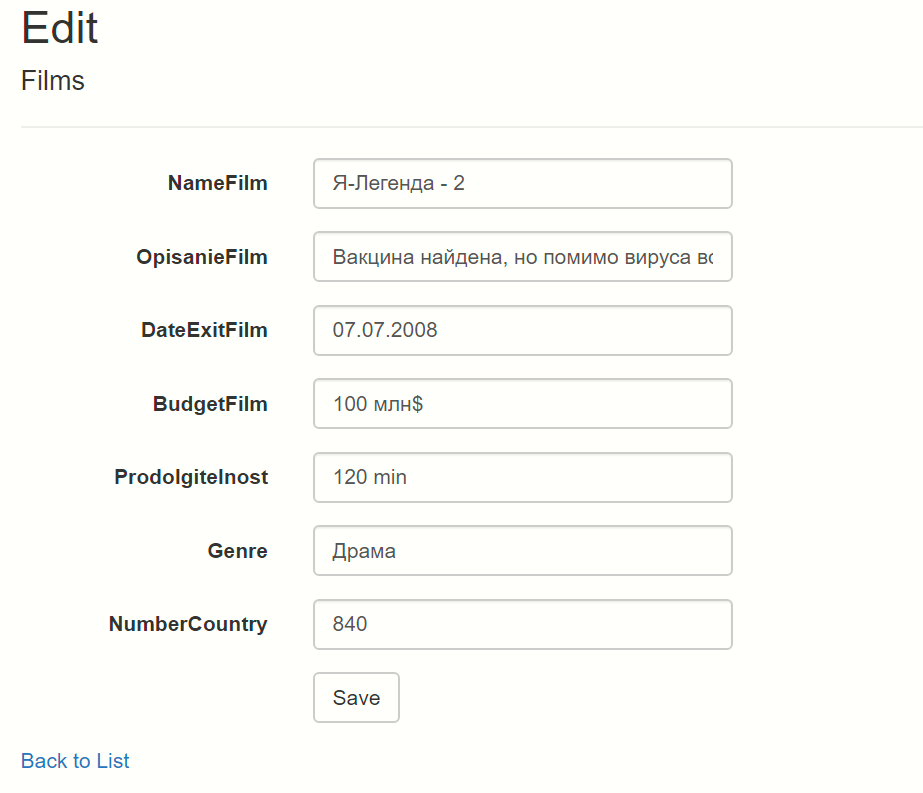


Рисунок 6 – Добавление фильма

Редактирование фильма (Рисунок 7).



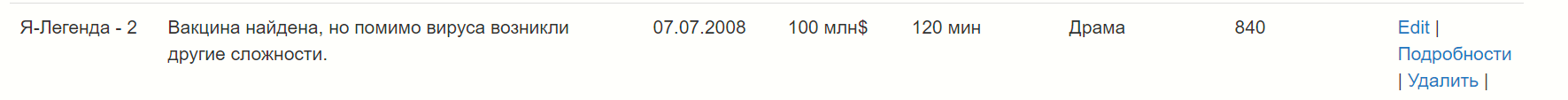
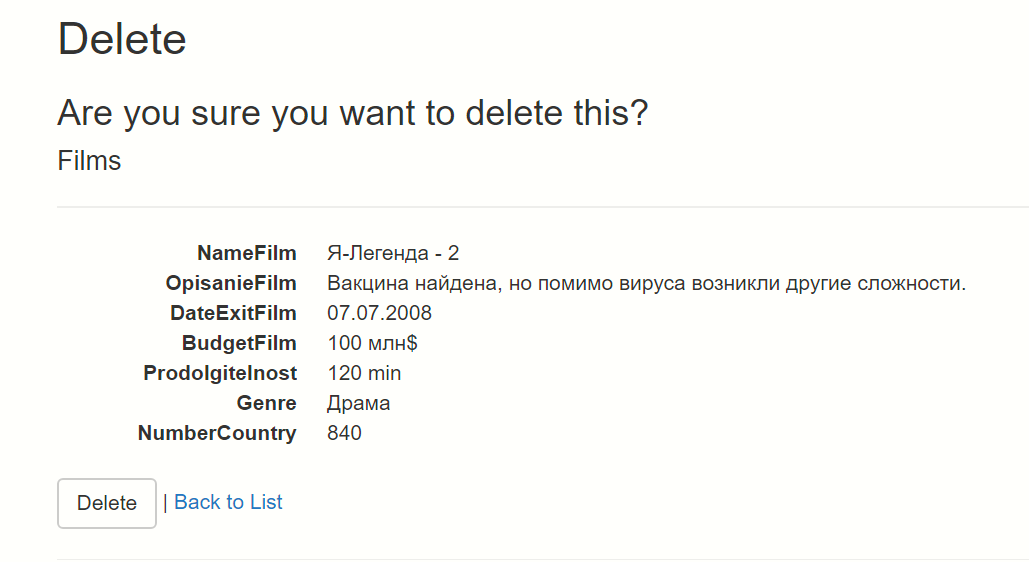


Рисунок 7 – Редактирование фильма

Удаление фильма (Рисунок 8).



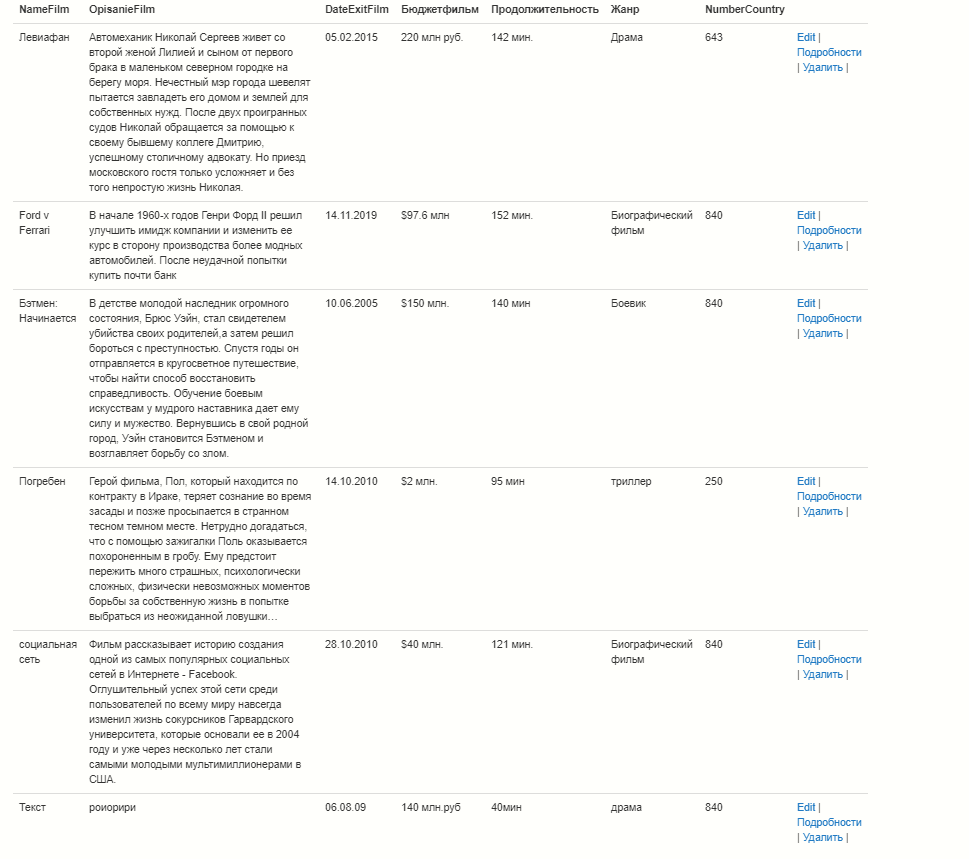


Рисунок 8 – Удаление фильма

# ЗАКЛЮЧЕНИЕ

Результатом работы курсового проекта стало приложение для хранения информации о кино. Во время работы над курсовым проектом было сделано:

* проектирование базы данных,
* создание физической модели в СУБД SQL Server,
* закрепление навыков в работе с языком программирования C#,
* изучена технология ASP.NET (.NET Framework) с использованием паттерна MVC,
* изучена технология Entity Framework
* работа с SQL Server,
* оформления документации, удовлетворяющей стандартам ГОСТ.

# СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ

1. «Форум программистов «Cyberforum.ru»» [Электронный ресурс]. – 2020. – URL: https://[www.cyberforum.ru](http://www.cyberforum.ru/) (дата обращения 05.04.2020),
2. Руководство по SQL Server [Электронный ресурс]. – 2020. – URL: https://docs.microsoft.com/ru-ru/sql/sql-server/ (дата обращения 12.04.2020),
3. Базы данных и системы управления базами данных: Учебное пособие / Лазицкас Е.А., Загумённикова И.Н., Гилевский П.Г. – Мн.:РИПО. – 2016. – URL: http://znanium.com/ bookread2.php?book=946561,
4. ГОСТ 7.32 – 2017 СИБИД. Отчёт о научно-исследовательской работе. Структура и правила оформления,
5. Единая система конструкторской документации. Общие требования к текстовым документам [текст]: ГОСТ 2.105-95. – Взамен ГОСТ 2.105-79 и ГОСТ 2.906-71; введ. 01.07.1996. – М., 2007. – 28 с. – (Межгосударственный стандарт),
6. Положение по организации выполнения и защиты курсовой работы [текст]. – утв. Ректором университета д.э.н., профессором В.Н. Ивановой 29.08.2018; введено в действие 01.09.2018.

# ПРИЛОЖЕНИЕ А SQL скрипты

**Скрипт на создание и заполнение базы данных**

GO

USE Кино

GO

IF EXISTS (SELECT \* FROM SYSOBJECTS WHERE NAME='Uchastie' AND TYPE='U')

DROP TABLE Uchastie

IF EXISTS (SELECT \* FROM SYSOBJECTS WHERE NAME='reviews' AND TYPE='U')

DROP TABLE reviews

IF EXISTS (SELECT \* FROM SYSOBJECTS WHERE NAME='Films' AND TYPE='U')

DROP TABLE Films

IF EXISTS (SELECT \* FROM SYSOBJECTS WHERE NAME='Roles' AND TYPE='U')

DROP TABLE Roles

IF EXISTS (SELECT \* FROM SYSOBJECTS WHERE NAME='Crew' AND TYPE='U')

DROP TABLE Crew

IF EXISTS (SELECT \* FROM SYSOBJECTS WHERE NAME='list of countries' AND TYPE='U')

DROP TABLE [list of countries]

GO

CREATE TABLE [list of countries]

(NumberCountry INT PRIMARY KEY CHECK(NumberCountry>0),

[NameCountry] VARCHAR(20)

)

GO

CREATE TABLE Crew

(NumberHuman INT PRIMARY KEY CHECK(NumberHuman>0),

[lastname] VARCHAR(20),

[Name] VARCHAR(20),

[Patronomyc] VARCHAR(20),

[DateBirthday] CHAR(20)

)

GO

CREATE TABLE Roles

([NumberRole] CHAR(20),

[NumberFilm] CHAR(20),

[NameRole] VARCHAR(80)

CONSTRAINT PK\_Roles Primary KEY ([numberRole], [numberFilm])

)

GO

CREATE TABLE Films

( NumberFilm INT PRIMARY KEY CHECK(NumberFilm>0),

[NameFilm] VARCHAR(20),

[OpisanieFilm] VARCHAR(500),

[DateExitFilm] CHAR(20),

[BudgetFilm] CHAR(20),

[Prodolgitelnost] CHAR(20),

[Genre] VARCHAR(20),

[NumberCountry] INT

)

GO

CREATE TABLE Reviews

(NumberMark INT PRIMARY KEY CHECK(NumberMark>0),

[ZritelMark] CHAR(20),

[NumberZritel] CHAR(20),

[NumberFilm] INT

)

GO

CREATE TABLE Uchastie

(NumberZapici int PRIMARY KEY CHECK(NumberZapici>0),

[NumberHuman] INT,

[NumberFilm] INT,

[NumberRole] CHAR(20)

)

GO

/\*Страны\*/

INSERT INTO [dbo].[list of countries] VALUES('643', 'Russia')

INSERT INTO [dbo].[list of countries] VALUES('840', 'USA')

INSERT INTO [dbo].[list of countries] VALUES('250', 'France')

INSERT INTO [dbo].[list of countries] VALUES('352', 'Japan')

INSERT INTO [dbo].[list of countries] VALUES('826', 'UK')

/\* Съемочная группа фильм Левиафан \*/

INSERT INTO [dbo].[Crew] VALUES('1', 'Zvyagintsev', 'Andrey', 'Petrovich', '06.02.1964')

INSERT INTO [dbo].[Crew] VALUES('2', 'Negin', 'Oleg', 'Petrovich', '02.07.1970')

INSERT INTO [dbo].[Crew] VALUES('3', 'Rodnyansky', 'Alexander', 'Efimovich', '02.07.1961')

INSERT INTO [dbo].[Crew] VALUES('4', 'Serebryakov', 'Aleksei ', 'Valerievich', '03.07.1964')

INSERT INTO [dbo].[Crew] VALUES('5', 'Lyadova','Elena', 'Igorevna', '25.12.1980')

INSERT INTO [dbo].[Crew] VALUES('6', 'Vdovichenkov', 'Vladimir', 'Vladimirovich', '13.08.1971')

INSERT INTO [dbo].[Crew] VALUES('7', 'Madyanov', 'Roman', 'Sergeevich', '22.07.1962')

INSERT INTO [dbo].[Crew] VALUES('8', 'Krichman', 'Mikhail ', 'Vladimirovich', '17.06.1967')

INSERT INTO [dbo].[Crew] VALUES('9', 'Glass', 'Philip', 'None', '31.02.1937')

/\* Съемочная группа фильм Ford против Ferrari\*/

INSERT INTO [dbo].[Crew] VALUES('10', 'Mangold', 'James', 'NULL', '16.12.1963')

INSERT INTO [dbo].[Crew] VALUES('11', 'Butterworth', 'Jez', 'NULL', '16.03.1969')

INSERT INTO [dbo].[Crew] VALUES('12', 'Chernin', 'Peter', 'NULL', '29.05.1951')

INSERT INTO [dbo].[Crew] VALUES('13', 'Bale', 'Christian', 'Charles Philip', '30.01.1974')

INSERT INTO [dbo].[Crew] VALUES('14', 'Damon', 'Matthew', 'Paige', '08.10.1970')

INSERT INTO [dbo].[Crew] VALUES('15', 'Balfe', 'Caitriona', 'NULL', '04.10.1979')

INSERT INTO [dbo].[Crew] VALUES('16', 'Letts', 'Tracy', 'NULL', '04.07.1965')

INSERT INTO [dbo].[Crew] VALUES('17', 'Papamichael', 'Fidon', 'NULL', '04.02.1962')

INSERT INTO [dbo].[Crew] VALUES('18', 'Beltrami', 'Marco', 'NULL', '07.10.1966')

/\* Роли в Левиафане\*/

INSERT INTO [dbo].[Roles] VALUES('1', '1', 'Andrey Zvagintsev/director/Leviafan')

INSERT INTO [dbo].[Roles] VALUES('2', '1', 'Oleg Negin/scriptwriter/Leviafan')

INSERT INTO [dbo].[Roles] VALUES('3', '1', 'Alexander Rodnyansky/producer/Leviafan')

INSERT INTO [dbo].[Roles] VALUES('4', '1', 'Aleksei Serebryakov/Nicolay/actor/Leviafan')

INSERT INTO [dbo].[Roles] VALUES('5', '1', 'Elena Lyadova/Lylia/actor/Leviafan')

INSERT INTO [dbo].[Roles] VALUES('6', '1', 'Vladimir Vdovichenkov/Dmitry/actor/Leviafan')

INSERT INTO [dbo].[Roles] VALUES('7', '1', 'Roman Madyanov/Vadim/actor/Leviafan')

INSERT INTO [dbo].[Roles] VALUES('8', '1', 'Mikhail Krichman/operator/Leviafan')

INSERT INTO [dbo].[Roles] VALUES('9', '1', 'Philip Glass/composer/Leviafan')

/\* Роли в Ford против Ferrari\*/

INSERT INTO [dbo].[Roles] VALUES('10', '2', 'James Mangold/director/Ford v Ferrari')

INSERT INTO [dbo].[Roles] VALUES('11', '2', 'Jez Butterworth/scriptwriter/Ford v Ferrari')

INSERT INTO [dbo].[Roles] VALUES('12', '2', 'Peter Chernin/producer/Ford v Ferrari')

INSERT INTO [dbo].[Roles] VALUES('13', '2', 'Christiian Bale/Ken Miles/actor/Ford v Ferrari')

INSERT INTO [dbo].[Roles] VALUES('14', '2', 'Matt Damon/Carroll Shelby/actor/Ford v Ferrari')

INSERT INTO [dbo].[Roles] VALUES('15', '2', 'Caitriona Balfe/Molly Miles/actor/Ford v Ferrari')

INSERT INTO [dbo].[Roles] VALUES('16', '2', 'Tracy Letts/Henry Ford II/actor/Ford v Ferrari')

INSERT INTO [dbo].[Roles] VALUES('17', '2', 'Fidon Papamichael/operator/Ford v Ferrari')

INSERT INTO [dbo].[Roles] VALUES('18', '2', 'Marco Beltrami/compsoer/Ford v Ferrari')

/\* Фильм Левиафан\*/

INSERT INTO [dbo].[Films] VALUES('1', 'Leviafan', 'Car mechanic Nikolai Sergeyev lives with his second wife Lily and his son from his first marriage in a small northern town on the seashore. The dishonest mayor of the city of Shevelyat is trying to seize his house and land for his own needs. After two lost courts, Nikolai turns for help to his former colleague Dmitry, a successful metropolitan lawyer. But the arrival of the Moscow guest only complicates the already difficult life of Nikolai.', '05.02.2015', '220 млн руб.', '142 min', 'Drama', '643')

/\* Фильм Ford против Ferrari\*/

INSERT INTO [dbo].[Films] VALUES('2', 'Ford v Ferrari', 'In the early 1960s, Henry Ford II decided to improve the companys image and change its course towards the production of more fashionable cars. After a failed attempt to buy almost bank', '14.11.2019', '$97.6 млн', '152 min', 'Biography film', '840')

/\* Отзывы о фильме Левиафан\*/

INSERT INTO [dbo].[Reviews] VALUES('1', '10 out of 10', '1', '1')

INSERT INTO [dbo].[Reviews] VALUES('2', '7 out of 10', '2', '1')

INSERT INTO [dbo].[Reviews] VALUES('3', '6 out of 10', '3', '1')

/\* Отзывы о фильме Ford против Ferrari\*/

INSERT INTO [dbo].[Reviews] VALUES('4', '8 out of 10', '4', '2')

INSERT INTO [dbo].[Reviews] VALUES('5', '9 out of 10', '5', '2')

INSERT INTO [dbo].[Reviews] VALUES('6', '7 out of 10', '6', '2')

/\*Участие в фильме Левиафан\*/

INSERT INTO [dbo].[Uchastie] VALUES('1', '1', '1', '1')

INSERT INTO [dbo].[Uchastie] VALUES('2', '2', '1', '2')

INSERT INTO [dbo].[Uchastie] VALUES('3', '3', '1', '3')

INSERT INTO [dbo].[Uchastie] VALUES('4', '4', '1', '4')

INSERT INTO [dbo].[Uchastie] VALUES('5', '5', '1', '5')

INSERT INTO [dbo].[Uchastie] VALUES('6', '6', '1', '6')

INSERT INTO [dbo].[Uchastie] VALUES('7', '7', '1', '7')

INSERT INTO [dbo].[Uchastie] VALUES('8', '8', '1', '8')

INSERT INTO [dbo].[Uchastie] VALUES('9', '9', '1', '9')

/\*Участие в фильме Ford v Ferrari\*/

INSERT INTO [dbo].[Uchastie] VALUES('10', '10', '2', '10')

INSERT INTO [dbo].[Uchastie] VALUES('11', '11', '2', '11')

INSERT INTO [dbo].[Uchastie] VALUES('12', '12', '2', '12')

INSERT INTO [dbo].[Uchastie] VALUES('13', '13', '2', '13')

INSERT INTO [dbo].[Uchastie] VALUES('14', '14', '2', '14')

INSERT INTO [dbo].[Uchastie] VALUES('15', '15', '2', '15')

INSERT INTO [dbo].[Uchastie] VALUES('16', '16', '2', '16')

INSERT INTO [dbo].[Uchastie] VALUES('17', '17', '2', '17')

INSERT INTO [dbo].[Uchastie] VALUES('18', '18', '2', '18')

/\* Съемочная группа фильма Бэтмен:начало \*/

INSERT INTO [dbo].[Crew] VALUES('19', 'Nolan', 'Christopher', 'Jonathan James', '30.07.1970')

INSERT INTO [dbo].[Crew] VALUES('20', 'Nolan', 'Christopher', 'Jonathan James', '02.07.1970')

INSERT INTO [dbo].[Crew] VALUES('21', 'Tomas', 'Emma', 'NULL', '09.12.1971')

INSERT INTO [dbo].[Crew] VALUES('22', 'Bale', 'Christian', 'Charles Philip', '30.01.1974')

INSERT INTO [dbo].[Crew] VALUES('23', 'Holmes','Kate', 'Noelle', '18.12.1978')

INSERT INTO [dbo].[Crew] VALUES('24', 'Micklewhite', 'Maurice', 'Joseph', '14.03.1933')

INSERT INTO [dbo].[Crew] VALUES('25', 'Neeson', 'William', 'John', '07.06.1952')

INSERT INTO [dbo].[Crew] VALUES('26', 'Pfister', 'Walter', 'NULL', '08.07.1961')

INSERT INTO [dbo].[Crew] VALUES('27', 'Zimmer', 'Hanz', 'Florian', '12.09.1957')

/\*Роли в Бэтмен:Начало\*/

INSERT INTO [dbo].[Roles] VALUES('19', '3', 'Christopher Nolan/director/Batman:Begins')

INSERT INTO [dbo].[Roles] VALUES('20', '3', 'Christopher Nolan/scriptwriter/Batman:Begins')

INSERT INTO [dbo].[Roles] VALUES('21', '3', 'Emma Tomas/producer/Batman:Begins')

INSERT INTO [dbo].[Roles] VALUES('22', '3', 'Christiian Bale/Bruce Wayne,Batman/actor/Batman:Begins')

INSERT INTO [dbo].[Roles] VALUES('23', '3', 'Katie Holmes/Rachel Dawes/actor/Batman:Begins')

INSERT INTO [dbo].[Roles] VALUES('24', '3', 'Michael Caine/Alphred Pennyworth/actor/Batman:Begins')

INSERT INTO [dbo].[Roles] VALUES('25', '3', 'Liam Neeson/Ra’s al Ghul/actor/Batman:Begins')

INSERT INTO [dbo].[Roles] VALUES('26', '3', 'Wally Pfister/operator/Batman:Begins')

INSERT INTO [dbo].[Roles] VALUES('27', '3', 'Hanz Zimmer/compsoer/Batman:Begins')

/\*Фильм Бэтмен:Начало\*/

INSERT INTO [dbo].[Films] VALUES('3', 'Batman:Begins', 'As a child, the young heir to a huge fortune, Bruce Wayne, witnessed the murder of his parents, and then he decided to fight crime. Years later, he goes on a journey around the world to find a way to restore justice. Learning martial arts from a wise mentor gives him strength and courage. Returning to his hometown, Wayne becomes Batman and leads the fight against evil.', '10.06.2005', '$150 млн.', '140 min', 'Action movie', '840')

/\* Отзывы о фильме Бэтмен:Начало\*/

INSERT INTO [dbo].[Reviews] VALUES('7', '10 out of 10', '7', '3')

INSERT INTO [dbo].[Reviews] VALUES('8', '10 out of 10', '8', '3')

INSERT INTO [dbo].[Reviews] VALUES('9', '10 out of 10', '9', '3')

/\*Участие в фильме Бэтмен:Начало\*/

INSERT INTO [dbo].[Uchastie] VALUES('19', '19', '3', '19')

INSERT INTO [dbo].[Uchastie] VALUES('20', '20', '3', '20')

INSERT INTO [dbo].[Uchastie] VALUES('21', '21', '3', '21')

INSERT INTO [dbo].[Uchastie] VALUES('22', '22', '3', '22')

INSERT INTO [dbo].[Uchastie] VALUES('23', '23', '3', '23')

INSERT INTO [dbo].[Uchastie] VALUES('24', '24', '3', '24')

INSERT INTO [dbo].[Uchastie] VALUES('25', '25', '3', '25')

INSERT INTO [dbo].[Uchastie] VALUES('26', '26', '3', '26')

INSERT INTO [dbo].[Uchastie] VALUES('27', '27', '3', '27')

/\* Съемочная группа фильма Погребенный заживо\*/

INSERT INTO [dbo].[Crew] VALUES('28', 'Cortez', 'Rodrigo', 'Giraldez', '31.05.1973')

INSERT INTO [dbo].[Crew] VALUES('29', 'Sparling', 'Christopher', 'NULL', '21.03.1977')

INSERT INTO [dbo].[Crew] VALUES('30', 'Safran', 'Peter', 'NULL', '22.11.1965')

INSERT INTO [dbo].[Crew] VALUES('31', 'Reynolds', 'Ryan', 'Rodney', '23.10.1976')

INSERT INTO [dbo].[Crew] VALUES('32', 'Tobolowsky','Stephen', 'Harold', '30.05.1951')

INSERT INTO [dbo].[Crew] VALUES('33', 'Mathis', 'Samantha', 'NULL', '12.05.1970')

INSERT INTO [dbo].[Crew] VALUES('34', 'Perez', 'Jose', 'Luis Garcia', '01.09.1972')

INSERT INTO [dbo].[Crew] VALUES('35', 'Grau', 'Eduard', 'NULL', '06.07.1981')

INSERT INTO [dbo].[Crew] VALUES('36', 'Reyes', 'Victor', 'NULL', '25.02.1962')

/\*Роли в Погребенный заживо\*/

INSERT INTO [dbo].[Roles] VALUES('28', '4', 'Rodrigo Cortez/director/Buried')

INSERT INTO [dbo].[Roles] VALUES('29', '4', 'Chris Sparling/scriptwriter/Buried')

INSERT INTO [dbo].[Roles] VALUES('30', '4', 'Peter Safran/producer/Buried')

INSERT INTO [dbo].[Roles] VALUES('31', '4', 'Ryan Reynolds/Paul/actor/Buried')

INSERT INTO [dbo].[Roles] VALUES('32', '4', 'Stephen Tobolowsky/Alan/actor/Buried')

INSERT INTO [dbo].[Roles] VALUES('33', '4', 'Samantha Mathis/Linda/actor/Buried')

INSERT INTO [dbo].[Roles] VALUES('34', '4', 'Jose Perez/Jabir/actor/Buried')

INSERT INTO [dbo].[Roles] VALUES('35', '4', 'Eduard Grau/operator/Buried')

INSERT INTO [dbo].[Roles] VALUES('36', '4', 'Victor Reyes/composer/Buried')

/\*Фильм Погребенный заживо\*/

INSERT INTO [dbo].[Films] VALUES('4', 'Buried', 'The hero of the film, Paul, who is under contract in Iraq, loses consciousness during an ambush and later wakes up in a strange close dark place. It is not difficult to guess that using a lighter, Paul finds himself buried in a coffin. He will have to go through many terrible, psychologically difficult, physically impossible moments of fighting for his own life in an attempt to get out of an unexpected trap…', '14.10.2010', '$2 млн.', '95 min', 'thriller', '250')

/\* Отзывы о фильме Погребенный заживо\*/

INSERT INTO [dbo].[Reviews] VALUES('10', '1 out of 10', '10', '4')

INSERT INTO [dbo].[Reviews] VALUES('11', '5 out of 10', '11', '4')

INSERT INTO [dbo].[Reviews] VALUES('12', '8 out of 10', '12', '4')

/\*Участие в фильме Погребенный заживо\*/

INSERT INTO [dbo].[Uchastie] VALUES('28', '28', '4', '28')

INSERT INTO [dbo].[Uchastie] VALUES('29', '29', '4', '29')

INSERT INTO [dbo].[Uchastie] VALUES('30', '30', '4', '30')

INSERT INTO [dbo].[Uchastie] VALUES('31', '31', '4', '31')

INSERT INTO [dbo].[Uchastie] VALUES('32', '32', '4', '32')

INSERT INTO [dbo].[Uchastie] VALUES('33', '33', '4', '33')

INSERT INTO [dbo].[Uchastie] VALUES('34', '34', '4', '34')

INSERT INTO [dbo].[Uchastie] VALUES('35', '35', '4', '35')

INSERT INTO [dbo].[Uchastie] VALUES('36', '36', '4', '36')

/\* Съемочная группа фильма Социальная сеть\*/

INSERT INTO [dbo].[Crew] VALUES('37', 'Fincher', 'David', 'Andrew Leo', '28.08.1962')

INSERT INTO [dbo].[Crew] VALUES('38', 'Sorokin', 'Aaron', 'Benjamin', '09.06.1961')

INSERT INTO [dbo].[Crew] VALUES('39', 'Spacey', 'Kevin', 'Fowler', '26.07.1959')

INSERT INTO [dbo].[Crew] VALUES('40', 'Eisenberg', 'Jesse', 'Adam', '05.10.1983')

INSERT INTO [dbo].[Crew] VALUES('41', 'Garfield','Andrew', 'Russel', '20.08.1983')

INSERT INTO [dbo].[Crew] VALUES('42', 'Timberlake', 'Justin', 'Randall', '31.01.1981')

INSERT INTO [dbo].[Crew] VALUES('43', 'Hammer', 'Armand', 'Douglas', '28.08.1986')

INSERT INTO [dbo].[Crew] VALUES('44', 'Cronenweth', 'Jeffrey', 'Scott', '14.01.1962')

INSERT INTO [dbo].[Crew] VALUES('45', 'Reznor', 'Michael', 'Trent', '25.02.1962')

/\*Роли в Социальная сеть\*/

INSERT INTO [dbo].[Roles] VALUES('37', '5', 'David Fincher/director/The Social Network')

INSERT INTO [dbo].[Roles] VALUES('38', '5', 'Aaron Sorokin/scriptwriter/The Social Network')

INSERT INTO [dbo].[Roles] VALUES('39', '5', 'Kevin Spacey/producer/The Social Network')

INSERT INTO [dbo].[Roles] VALUES('40', '5', 'Jesse Eisenberg/Mark Zuckerberg/actor/The Social Network')

INSERT INTO [dbo].[Roles] VALUES('41', '5', 'Andrew Garfield/Eduardo Saverin/actor/The Social Network')

INSERT INTO [dbo].[Roles] VALUES('42', '5', 'Justin Timberlake/Sean Parker/actor/The Social Network')

INSERT INTO [dbo].[Roles] VALUES('43', '5', 'Armie Hammer/Cameron Winklevoss, Tyler Winklevoss/actor/The Social Network')

INSERT INTO [dbo].[Roles] VALUES('44', '5', 'Jeffrey Croneweth/operator/The Social Network')

INSERT INTO [dbo].[Roles] VALUES('45', '5', 'Trent Reznor/composer/The Social Network')

/\*Фильм Социальная сеть\*/

INSERT INTO [dbo].[Films] VALUES('5', 'The Social Network', 'The film tells the story of the creation of one of the most popular social networks on the Internet - Facebook. The resounding success of this network among users around the world forever changed the lives of fellow students at Harvard University, who founded it in 2004 and within a few years became the youngest multi-millionaires in the United States.', '28.10.2010', '$40 млн.', '121 min', 'Biography film', '840')

/\* Отзывы о фильме Социальная сеть\*/

INSERT INTO [dbo].[Reviews] VALUES('13', '8 out of 10', '13', '5')

INSERT INTO [dbo].[Reviews] VALUES('14', '9 out of 10', '14', '5')

INSERT INTO [dbo].[Reviews] VALUES('15', '10 out of 10', '15', '5')

/\*Участие в фильме Социальная сеть\*/

INSERT INTO [dbo].[Uchastie] VALUES('37', '37', '5', '37')

INSERT INTO [dbo].[Uchastie] VALUES('38', '38', '5', '38')

INSERT INTO [dbo].[Uchastie] VALUES('39', '39', '5', '39')

INSERT INTO [dbo].[Uchastie] VALUES('40', '40', '5', '40')

INSERT INTO [dbo].[Uchastie] VALUES('41', '41', '5', '41')

INSERT INTO [dbo].[Uchastie] VALUES('42', '42', '5', '42')

INSERT INTO [dbo].[Uchastie] VALUES('43', '43', '5', '43')

INSERT INTO [dbo].[Uchastie] VALUES('44', '44', '5', '44')

INSERT INTO [dbo].[Uchastie] VALUES('45', '45', '5', '45')

GO

# ПРИЛОЖЕНИЕ Б Код приложения

using WebApplication.Models;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.Mvc;

using WebApplication.Models;

namespace WebApplication.Controllers

{

public class HomeController : Controller

{

private КиноEntities \_db = new КиноEntities();

// GET: Home

public ActionResult Index()

{

return View(\_db.Films.ToList());

}

// GET: Home/Details/5

public ActionResult Details(int id)

{

return View();

}

// GET: Home/Create

public ActionResult Create()

{

return View();

}

// POST: Home/Create

[HttpPost]

public ActionResult Create([Bind(Exclude = "ID")] Films filmsToCreate)

{

if (!ModelState.IsValid)

return View();

\_db.Films.Add(filmsToCreate);

\_db.SaveChanges();

return RedirectToAction("Index");

}

// GET: Home/Edit/

public ActionResult Edit(int id)

{

var filmsToEdit = (from m in \_db.Films

where m.NumberFilm == id

select m).First();

return View(filmsToEdit);

}

// POST: Home/Edit/

[HttpPost]

public ActionResult Edit(Films filmsToEdit)

{

var originalFilms = (from m in \_db.Films

where m.NumberFilm == filmsToEdit.NumberFilm

select m).First();

if (!ModelState.IsValid)

return View(originalFilms);

\_db.Entry(originalFilms).CurrentValues.SetValues(filmsToEdit);

\_db.SaveChanges();

return RedirectToAction("Index");

}

// GET: Home/Delete

public ActionResult Delete(int id)

{

var filmsToDelete = (from m in \_db.Films

where m.NumberFilm == id

select m).First();

return View(filmsToDelete);

}

// POST: Home/Delete

[HttpPost]

public ActionResult Delete(int id, FormCollection collection)

{

var filmsToDelete = (from m in \_db.Films

where m.NumberFilm == id

select m).First();

try

{

\_db.Films.Remove(filmsToDelete);

\_db.SaveChanges();

// TODO: Add delete logic here

return RedirectToAction("Index");

}

catch

{

return View(filmsToDelete);

}

}

}

}

//Index

@model IEnumerable<WebApplication.Films>

@{

ViewBag.Title = "Index";

}

<h2>Index</h2>

<p>

@Html.ActionLink("Create New", "Create")

</p>

<table class="table">

<tr>

<th>

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

</th>

<th>

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

</th>

<th>

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

</th>

<th>

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

</th>

<th>

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

</th>

<th>

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

</th>

<th>

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

</th>

<th></th>

</tr>

@foreach (var item in Model) {

<tr>

<td>

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

</td>

<td>

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

</td>

<td>

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

</td>

<td>

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

</td>

<td>

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

</td>

<td>

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

</td>

<td>

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

</td>

<td>

@Html.ActionLink("Edit", "Edit", new { id = item.NumberFilm }) |

@Html.ActionLink("Details", "Details", new { id = item.NumberFilm }) |

@Html.ActionLink("Delete", "Delete", new { id = item.NumberFilm }, null) |

</td>

</tr>

}

</table>

//Create

@model WebApplication.Films

@{

ViewBag.Title = "Create";

}

<h2>Create</h2>

@using (Html.BeginForm())

{

@Html.AntiForgeryToken()

<div class="form-horizontal">

<h4>Films</h4>

<hr />

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

<div class="form-group">

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

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

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

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

</div>

</div>

<div class="form-group">

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

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

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

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

</div>

</div>

<div class="form-group">

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

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

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

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

</div>

</div>

<div class="form-group">

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

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

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

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

</div>

</div>

<div class="form-group">

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

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

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

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

</div>

</div>

<div class="form-group">

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

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

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

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

</div>

</div>

<div class="form-group">

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

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

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

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

</div>

</div>

<div class="form-group">

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

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

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

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

</div>

</div>

<div class="form-group">

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

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

</div>

</div>

</div>

}

<div>

@Html.ActionLink("Back to List", "Index")

</div>

@section Scripts {

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

}

//Edit

@model WebApplication.Films

@{

ViewBag.Title = "Edit";

}

<h2>Edit</h2>

@using (Html.BeginForm())

{

@Html.AntiForgeryToken()

<div class="form-horizontal">

<h4>Films</h4>

<hr />

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

@Html.HiddenFor(model => model.NumberFilm)

<div class="form-group">

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

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

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

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

</div>

</div>

<div class="form-group">

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

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

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

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

</div>

</div>

<div class="form-group">

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

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

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

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

</div>

</div>

<div class="form-group">

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

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

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

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

</div>

</div>

<div class="form-group">

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

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

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

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

</div>

</div>

<div class="form-group">

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

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

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

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

</div>

</div>

<div class="form-group">

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

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

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

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

</div>

</div>

<div class="form-group">

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

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

</div>

</div>

</div>

}

<div>

@Html.ActionLink("Back to List", "Index")

</div>

@section Scripts {

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

}

//Delete

@model WebApplication.Films

@{

ViewBag.Title = "Delete";

}

<h2>Delete</h2>

<h3>Are you sure you want to delete this?</h3>

<div>

<h4>Films</h4>

<hr />

<dl class="dl-horizontal">

<dt>

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

</dt>

<dd>

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

</dd>

<dt>

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

</dt>

<dd>

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

</dd>

<dt>

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

</dt>

<dd>

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

</dd>

<dt>

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

</dt>

<dd>

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

</dd>

<dt>

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

</dt>

<dd>

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

</dd>

<dt>

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

</dt>

<dd>

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

</dd>

<dt>

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

</dt>

<dd>

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

</dd>

</dl>

@using (Html.BeginForm()) {

@Html.AntiForgeryToken()

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

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

@Html.ActionLink("Back to List", "Index")

</div>

}

</div>

//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("Имя приложения", "Index", "Home", new { area = "" }, new { @class = "navbar-brand" })

</div>

<div class="navbar-collapse collapse">

<ul class="nav navbar-nav">

<li>@Html.ActionLink("Домашняя страница", "Index", "Home")</li>

<li>@Html.ActionLink("Информация", "About", "Home")</li>

<li>@Html.ActionLink("Связаться", "Contact", "Home")</li>

</ul>

@Html.Partial("\_LoginPartial")

</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>