**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ**

федеральное государственное автономное образовательное

учреждение высшего образования

**НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ**

**ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

Инженерная школа информационных технологий и робототехники

Отделение информационных технологий

Направление «Информатика и вычислительная техника»

**Курсовой проект по дисциплине**

**«Современные платформы программирования»**

Разработка многостраничного web-сайта по шаблону MVC

Выполнил:

Студент группы 8ВМ82 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ А.К. Бокижонов

Проверил:

Доцент ОИТ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ В.Н. Попов

СОДЕРЖАНИЕ

Оглавление

Введение……………………………………………... [6](#_Toc4686681)

[Изменение представления и макета страницы 6](#_Toc4686682)

[Добавление классов модели 8](#_Toc4686683)

[Создание строки подключения и работа с SQL Server LocalDB 8](#_Toc4686684)

[Доступ к данным модели из контроллера 10](#_Toc4686685)

[Создание фильма 12](#_Toc4686686)

**Введение**

Курсовой проект посвящен разработке наиболее функционального интернет-магазина, удовлетворяющего желаниям большинства пользователей. Многие функции реализованы на основе изучения аналогов, а так же учитывая спрос и тенденции пользователей сети. Разработан дружественный интерфейс пользователя, удовлетворяющий требованиям проекта. Так же разработана администраторская часть проекта, в который владелец магазина имеет право управления всем магазином и содержимым базы данных. Рассматриваемый проект является конкурентоспособным и имеет право на внедрение и дальнейшую работу.

**Постановка задачи**

* 1. **Описание предметной области**

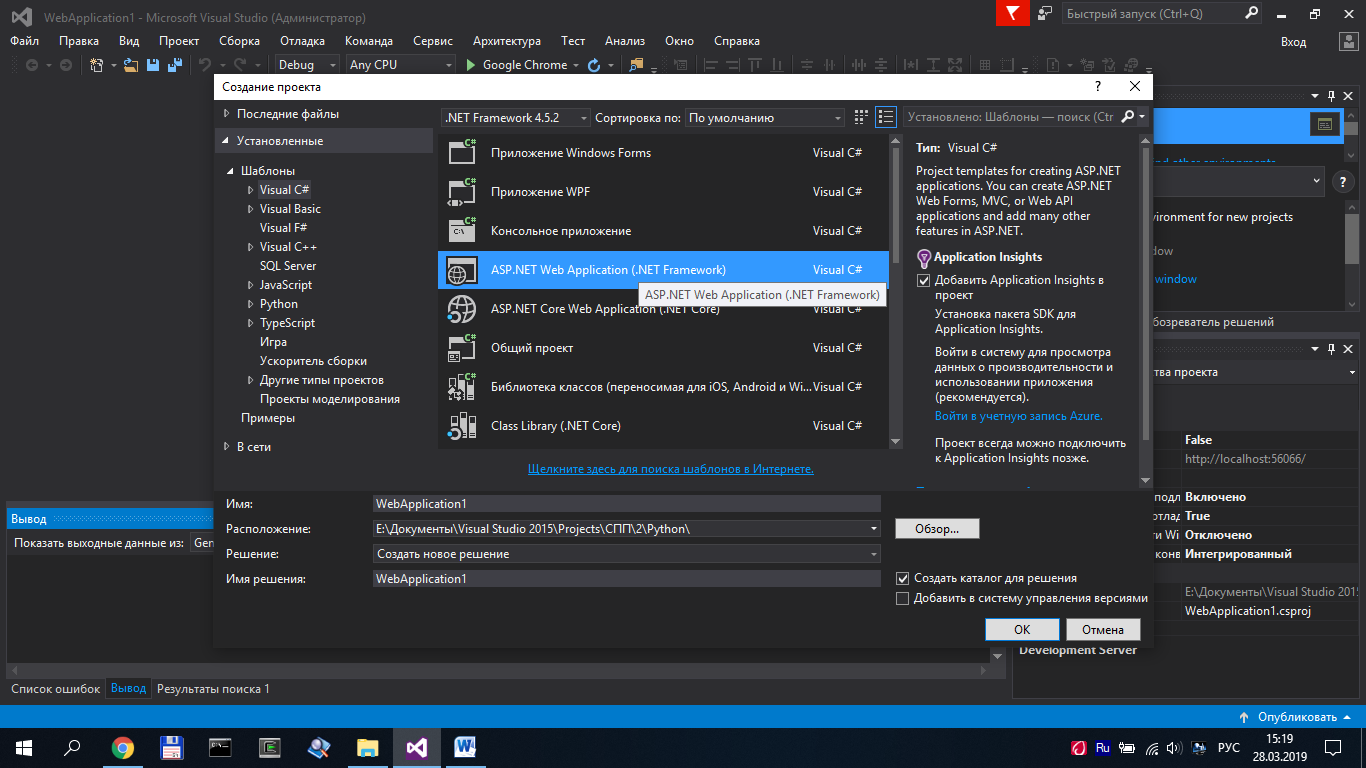
Разработать веб-приложение или веб сайт отвечающий следующим требованиям:

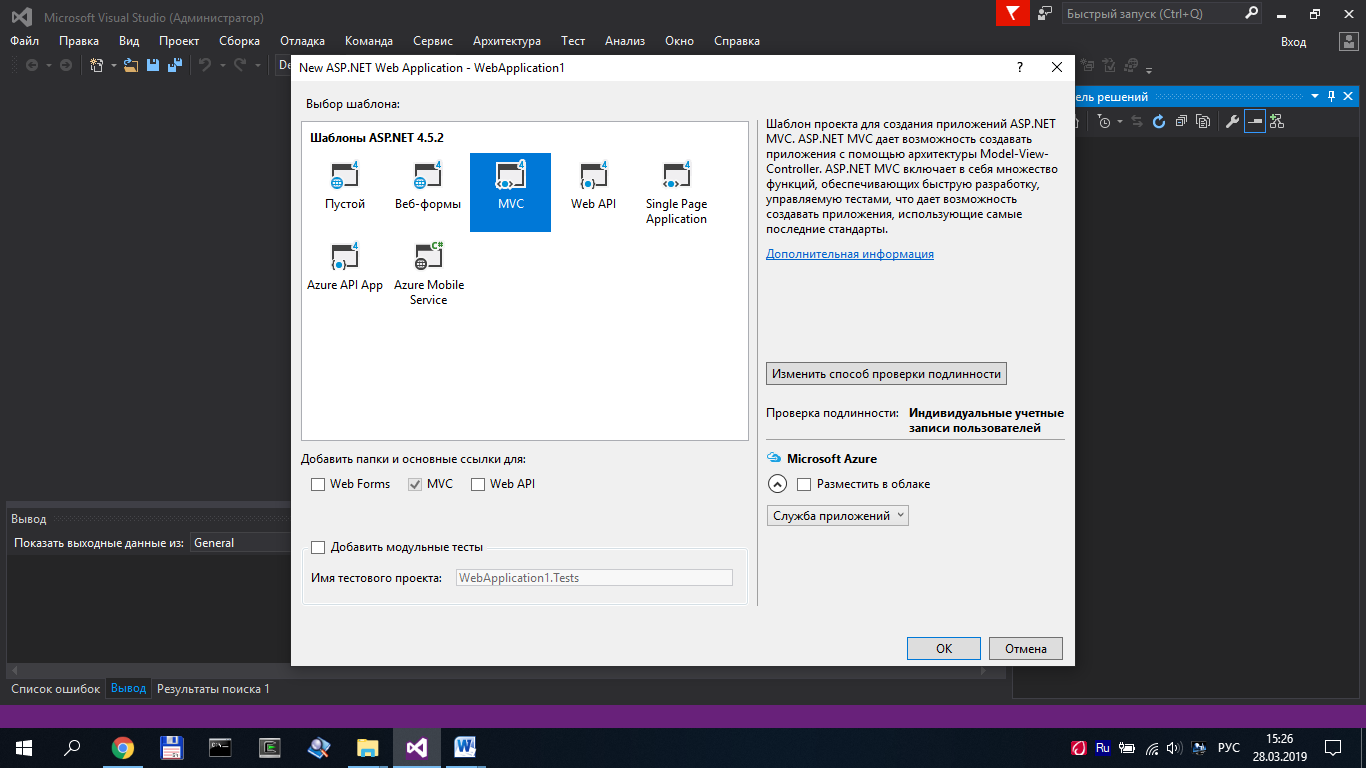
* Многостраничный веб-сайт с использованием шаблона MVC
* Хранение всей информации в СУБД
* Авторизация пользователей и администратора
* Интерфейсы доступа и для пользователя, и для администратора.
* Протоколирование основных действий приложения

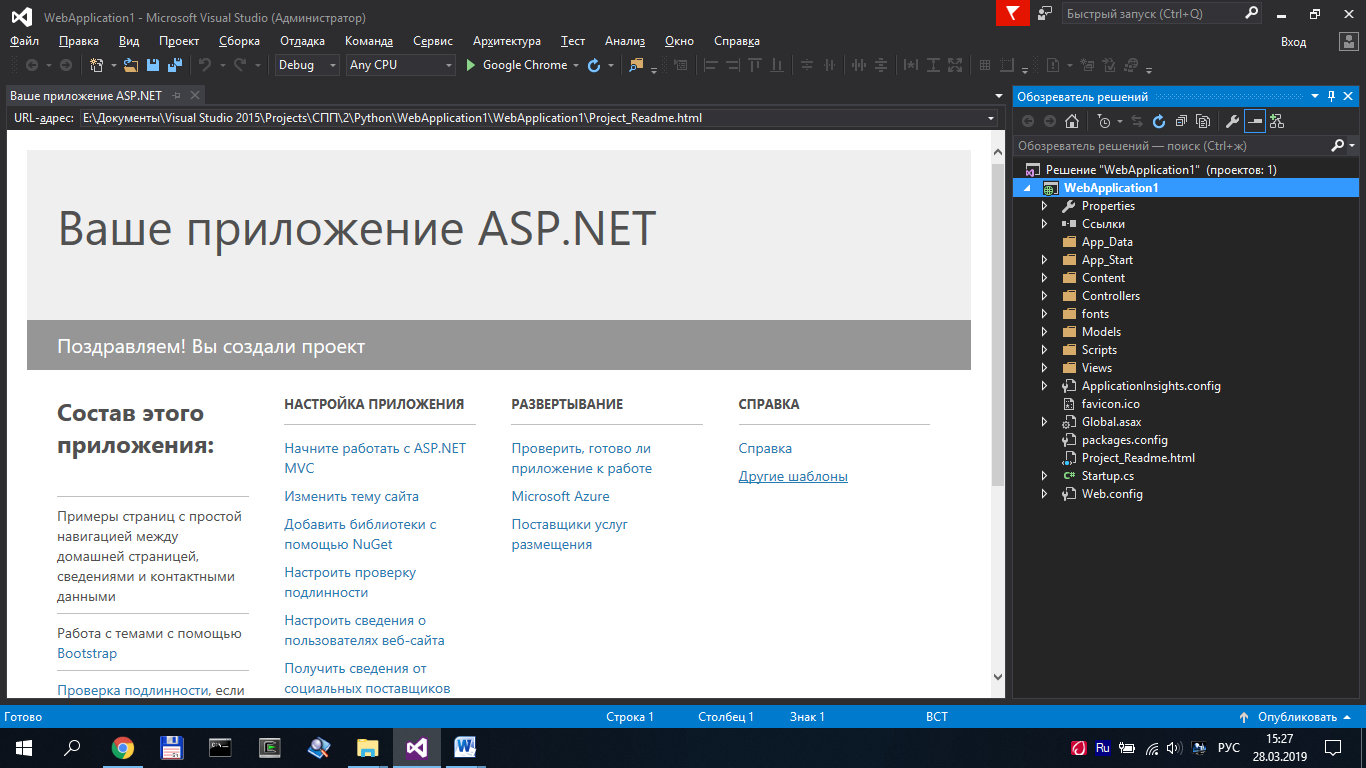
Новостной сайт на С# (новости вносят зарегистрированные пользователи, новости с тегами, поиск по тегам, каталог тегов)

**Начало** **работы**

Visual Studio является интегрированной среды разработки или интегрированной среды разработки. Создаем новый проект на начальной страницы, можно использовать в строке меню и выберите файл > новый проект.







Нажмите клавишу **F5** , чтобы начать отладку. При нажатии клавиши **F5**, Visual Studio запускает[IIS Express](https://docs.microsoft.com/ru-ru/iis/extensions/introduction-to-iis-express/iis-express-overview) и запускает веб-приложения. Затем Visual Studio запустит браузер и откроется домашняя страница приложения.

# 

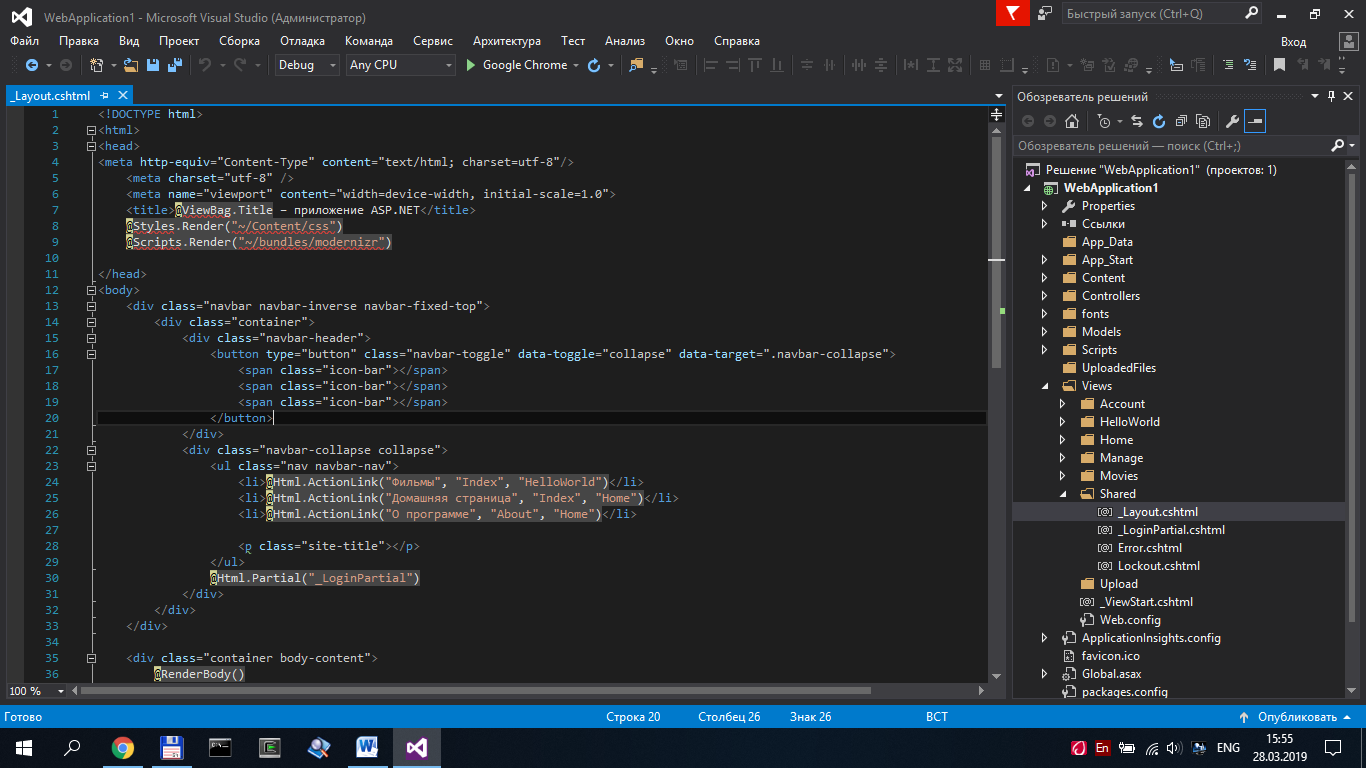
# 

MVC означает *model-view-controller*. MVC — это шаблон для разработки приложений, которые хорошо продуманной архитектурой, пригодного для тестирования и простые в обслуживании.Приложения на основе MVC содержат:

* **M** odels: Классы, представляющие данные приложения и используют логику проверки для реализации бизнес-правил к этим данным.
* **V** редставления: Файлы шаблонов, которые приложение использует для динамического создания HTML-ответов.
* **C** ontrollers: Классы, которые обрабатывают входящие запросы браузера, извлекают данные модели, а затем укажите шаблоны представлений, которые возвращают ответ в браузере.

## Изменение представления и макета страницы

Во-первых, необходимо изменить "имя\_приложения" ссылку в верхней части страницы. Этот текст является общим для каждой страницы. Фактически она реализуется только в одном месте в проекте, несмотря на то, что он отображается на каждой странице в приложении. Перейдите к /Views/Shared папку в **обозревателе решений** и откройте \_Layout.cshtml файла. Этот файл называется страницы макета и находится в общей папке, использовать другие страницы.



<!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 – приложение ASP.NET</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>

</div>

<div class="navbar-collapse collapse">

<ul class="nav navbar-nav">

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

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

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

<p class="site-title"></p>

</ul>

@Html.Partial("\_LoginPartial")

</div>

</div>

</div>

<div class="container body-content">

@RenderBody()

<hr />

<footer>

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

</footer>

</div>

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

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

@RenderSection("scripts", required: false)

</body>

</html>

## Добавление классов модели

В **обозревателе решений**, щелкните правой кнопкой мыши *моделей* папку, выберите**добавить**, а затем выберите **класс**.

Введите класс имя "фильма".

Добавьте следующие пять свойства Movie класса:

using System;

namespace MvcMovie.Models

{

public class Movie

{

public int ID { get; set; }

public string Title { get; set; }

public DateTime ReleaseDate { get; set; }

public string Genre { get; set; }

public decimal Price { get; set; }

}

}

MovieDBContext Класс представляет контекст базы данных movie Entity Framework, который обрабатывает получение, хранения и обновления Movie экземпляров в базе данных класса. MovieDBContext Является производным от DbContext базовый класс, предоставляемый платформой Entity Framework.

Чтобы иметь возможность ссылаться на DbContext и DbSet, необходимо добавить следующие using инструкция в верхней части файла:

**using System.Data.Entity;**

## Создание строки подключения и работа с SQL Server LocalDB

MovieDBContext Созданный класс обрабатывает задачи подключения к базе данных и сопоставления Movie объектов для записи базы данных. Один вопрос, на который вы можете спросить, однако, как указать базу данных, которая подключается к. Вы фактически нет необходимости указывать базу данных использовать, Entity Framework по умолчанию будет использовать [LocalDB](https://docs.microsoft.com/sql/database-engine/configure-windows/sql-server-2016-express-localdb). В этом разделе мы будем явно добавить строку подключения вWeb.config файл приложения. [LocalDB](https://docs.microsoft.com/sql/database-engine/configure-windows/sql-server-2016-express-localdb) — это облегченная версия SQL Server Express Database Engine, запускаемая по запросу и работает в пользовательском режиме. LocalDB выполняется в специального режима выполнения SQL Server Express, которая позволяет работать с базами данных как .mdf файлов.Как правило, хранятся файлы базы данных LocalDB в приложения\_данных папку веб-проекта.

Добавим следующую строку подключения для <**connectionStrings**> элемент в Web.config файл.

<?xml version="1.0" encoding="utf-8"?>

<configuration>

<configSections>

<!-- For more information on Entity Framework configuration, visit http://go.microsoft.com/fwlink/?LinkID=237468 -->

<section name="entityFramework" type="System.Data.Entity.Internal.ConfigFile.EntityFrameworkSection, EntityFramework, Version=6.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089" requirePermission="false" />

</configSections>

<connectionStrings>

**<add name="MovieDBContext"**

**connectionString="Data Source=(LocalDb)\MSSQLLocalDB;Initial Catalog=aspnet-MvcMovie;Integrated Security=SSPI;AttachDBFilename=|DataDirectory|\Movies.mdf" providerName="System.Data.SqlClient"/>**

<add name="DefaultConnection" connectionString="Data Source=(LocalDb)\MSSQLLocalDB;AttachDbFilename=|DataDirectory|\aspnet-WebApplication1-20190324092619.mdf;Initial Catalog=aspnet-WebApplication1-20190324092619;Integrated Security=True" providerName="System.Data.SqlClient" />

</connectionStrings>

Имя строки подключения должно соответствовать имя [DbContext](https://msdn.microsoft.com/library/system.data.entity.dbcontext(v=vs.103).aspx) класса.

C#

using System;

using System.Data.Entity;

namespace MvcMovie.Models

{

    public class Movie

    {

        public int ID { get; set; }

        public string Title { get; set; }

        public DateTime ReleaseDate { get; set; }

        public string Genre { get; set; }

        public decimal Price { get; set; }

    }

    public class MovieDBContext : DbContext

    {

        public DbSet<Movie> Movies { get; set; }

    }

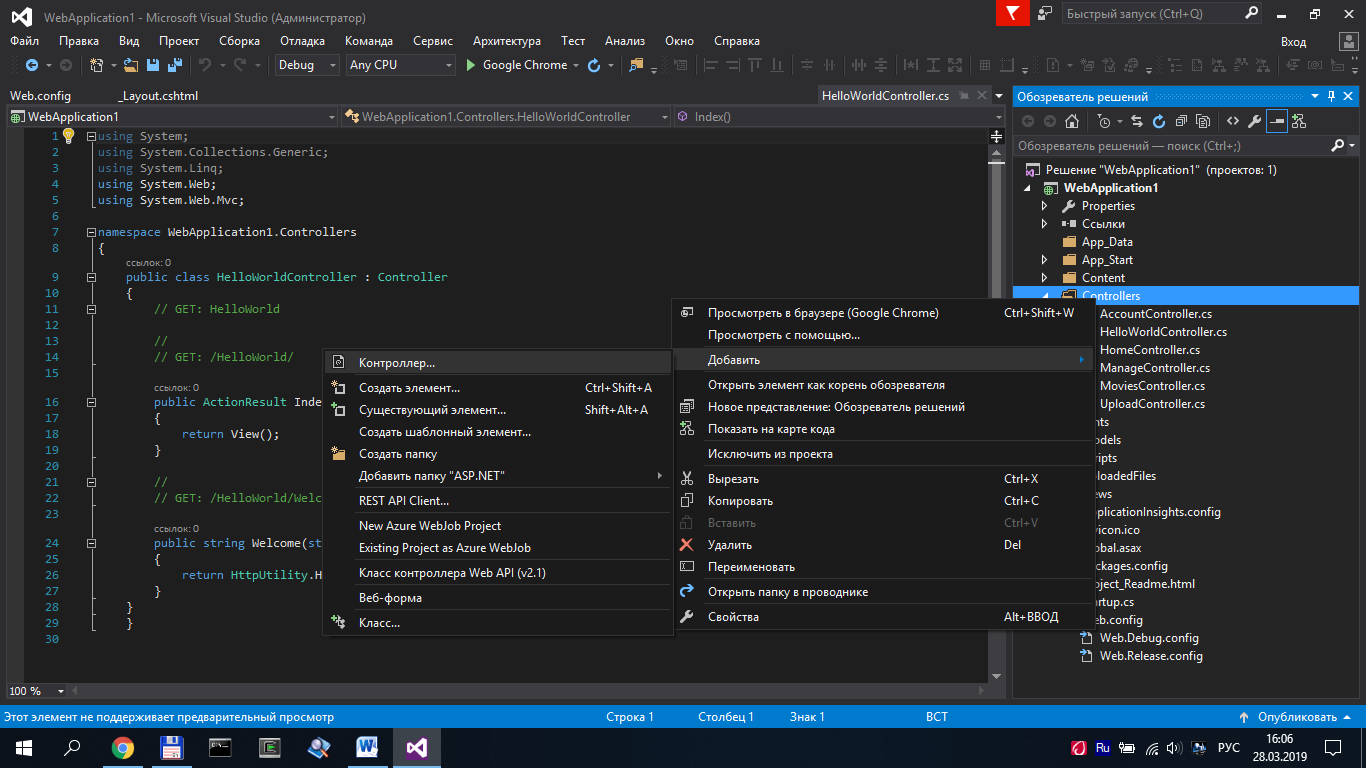
}

# Доступ к данным модели из контроллера

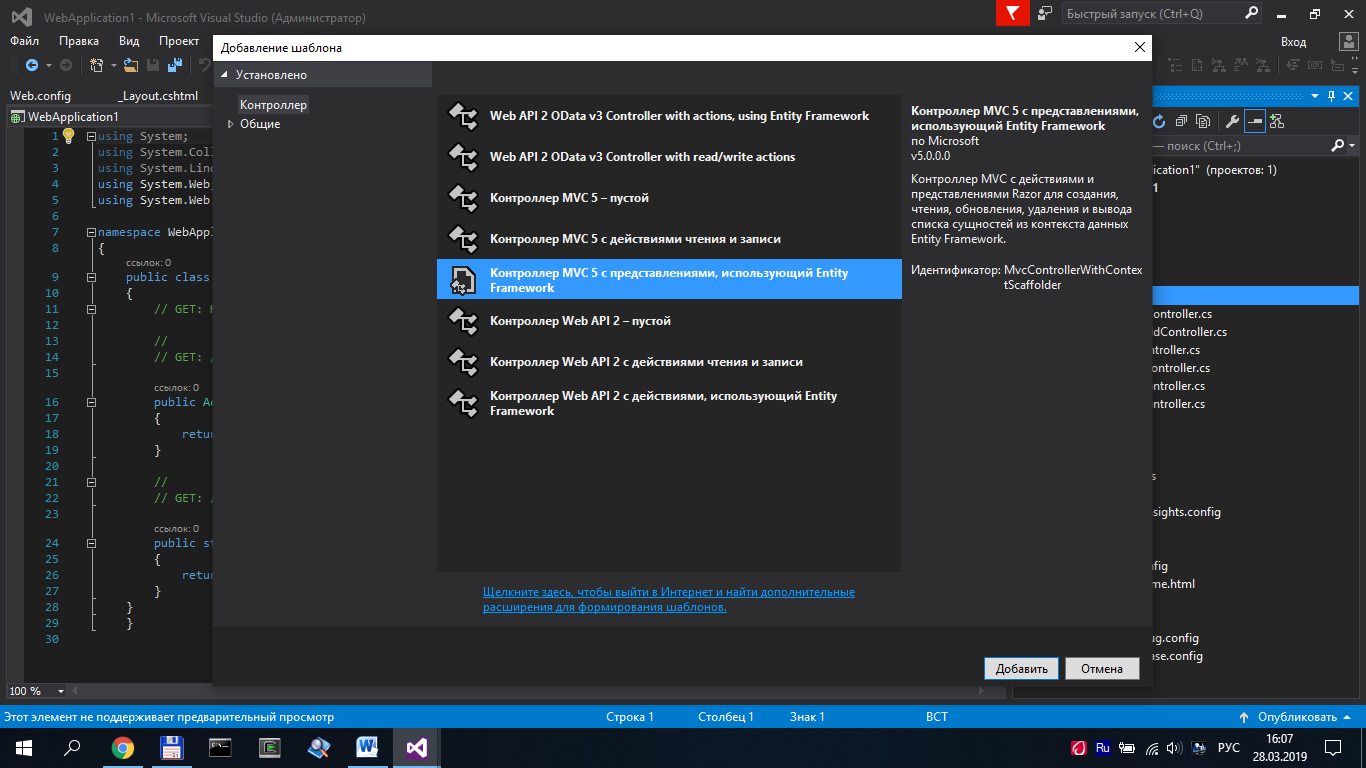
В этом разделе мы создадите новый MoviesController класса и написать код, который извлекает данные фильма и отображает его в браузере с помощью шаблона представления.

**Постройте приложение** перед переходом к следующему шагу. Если вы не создадите приложение, вы получите ошибку при добавлении контроллера.

В обозревателе решений щелкните правой кнопкой мыши контроллеров папку и нажмите кнопку **добавить**, затем **контроллера**.

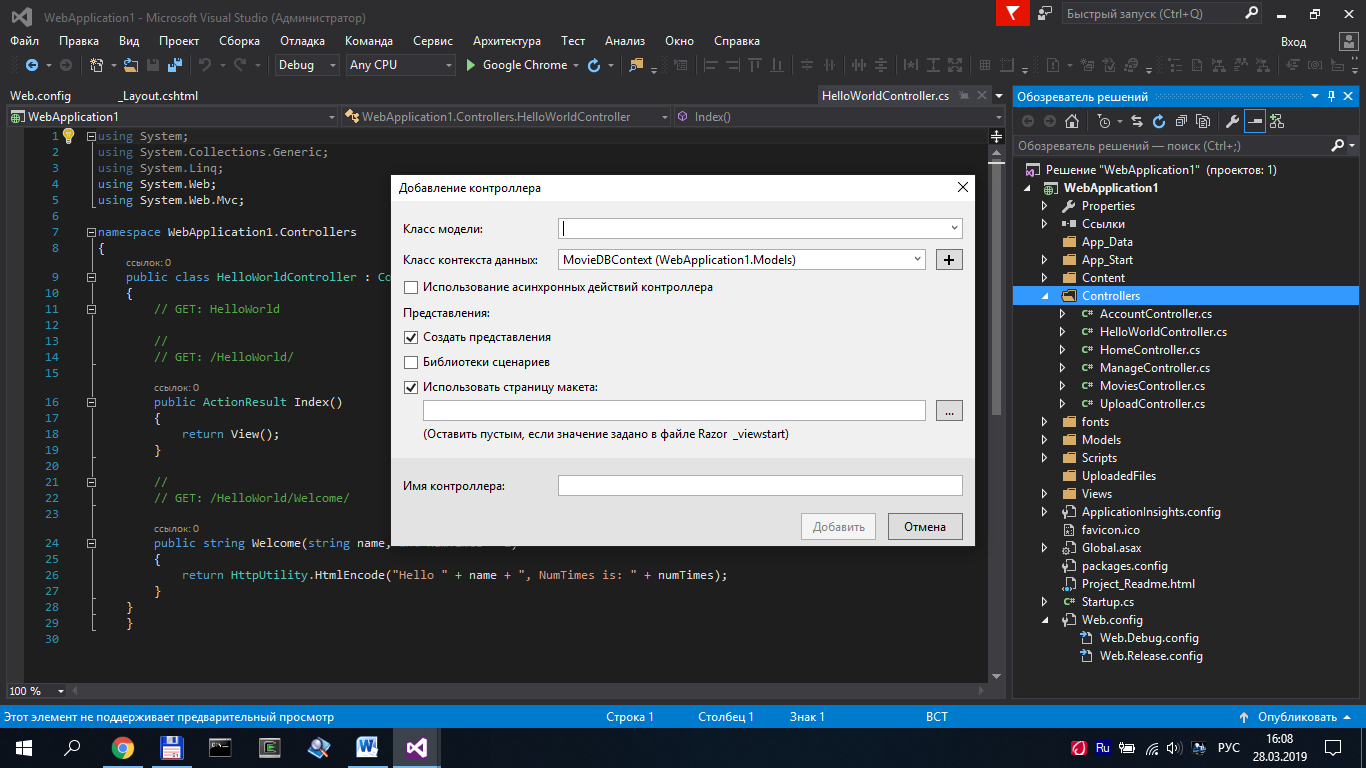


**Добавление шаблона** диалоговом окне щелкните **контроллер MVC 5 с представлениями, использующий Entity Framework**, а затем нажмите кнопку **добавить**.

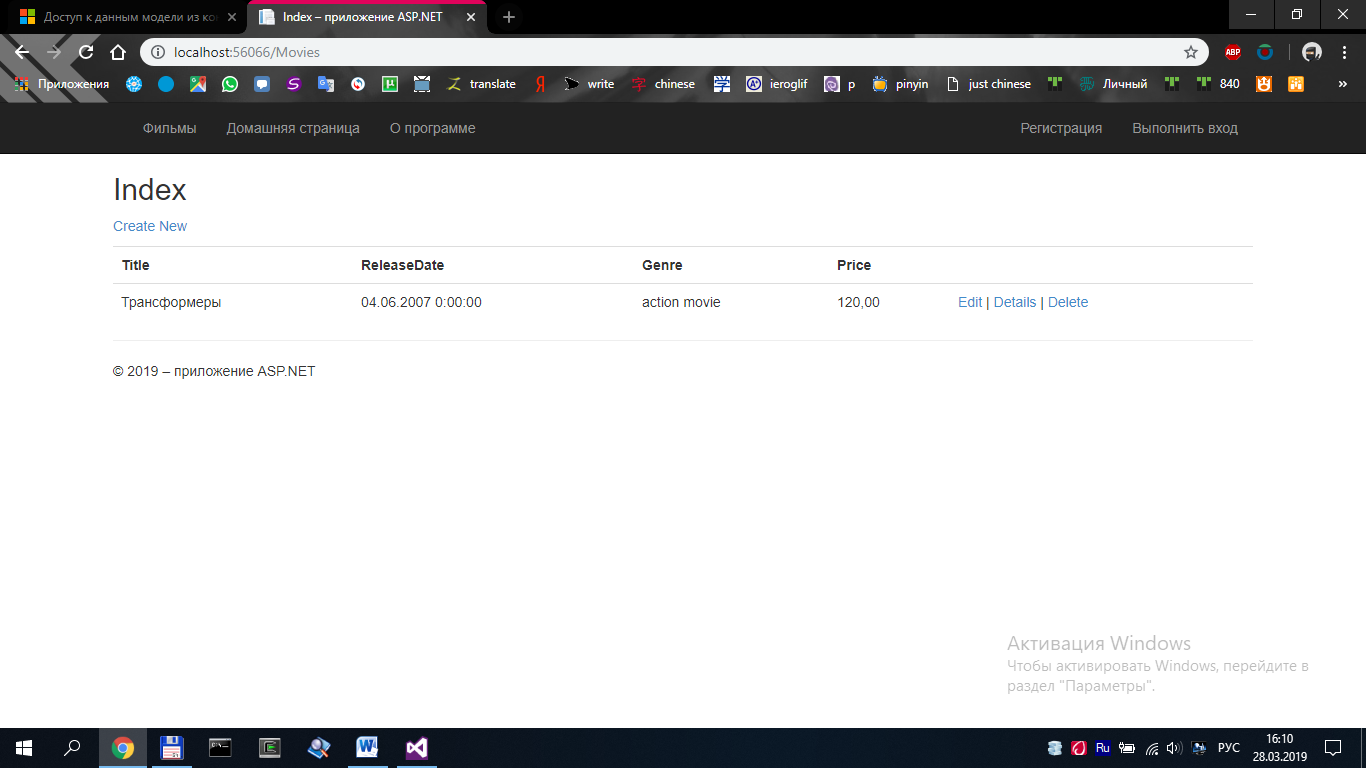


* Выберите **Movie (MvcMovie.Models)** класс модели.
* Выберите **MovieDBContext (MvcMovie.Models)** для класса контекста данных.
* Имя контроллера введите **MoviesController**.

На следующем рисунке показано завершенное диалоговое окно.

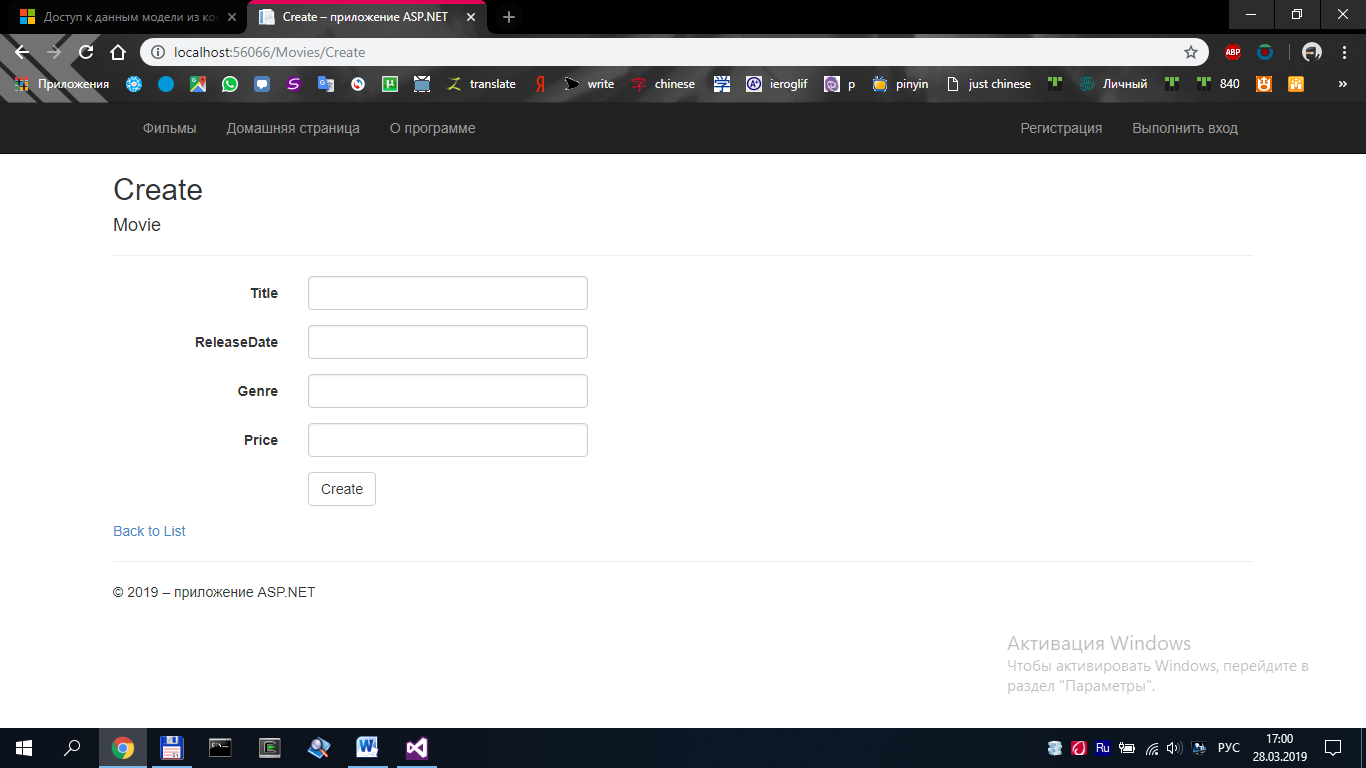


Запустите приложение и щелкнуть **MVC Movie** ссылку (или перейдите к Movies контроллер, добавляя /Movies на URL-адрес в адресной строке браузера). Так как приложение полагается на маршрутизацию по умолчанию (определенный в приложения\_Start\RouteConfig.cs файл), запрос браузера http://localhost:xxxxx/Movies направляется по умолчанию Index метод действия Movies контроллера. Другими словами, запрос браузера http://localhost:xxxxx/Movies так же, как запрос браузера http://localhost:xxxxx/Movies/Index.Результатом является пустой список фильмов, так как мы их добавили.



### Создание фильма

Щелкните ссылку **Create New** (Создать). Введите некоторые сведения о фильм, а затем нажмите кнопку **создать** кнопки.



Откройте Controllers\MoviesController.cs файла и изучите созданный Index метод. Часть контроллера movie с Index метод приведен ниже.

public class MoviesController : Controller

{

private MovieDBContext db = new MovieDBContext();

// GET: /Movies/

public ActionResult Index()

{

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

}

Запрос на Movies контроллер возвращает все записи в Movies таблицу и затем передает результаты в Index представления. В следующей строке из MoviesController класс создает экземпляр контекста базы данных фильмов, как описано выше. Контекст базы данных movie позволяет запрашивать, изменять и удалять элементы.

Ранее в этом учебнике вы видели, как контроллер может передать данные или объекты в шаблон представления с помощью ViewBag объекта. ViewBag Является динамическим объектом, который предоставляет удобный способ для передачи информации в представление с поздним связыванием.

MVC также предоставляет возможность передавать строго типизированные объекты шаблона представления. Такой подход строго типизированные позволяет лучше компиляции во время проверки кода и более широкие [IntelliSense](https://msdn.microsoft.com/library/hcw1s69b(v=vs.120).aspx) в редакторе Visual Studio. Этот подход используется механизм формирования шаблонов в Visual Studio (то есть передача строготипизированной модели) с MoviesController класс и представление шаблонов при создании методов и представлений.

В Controllers\MoviesController.cs изучите созданный файл Details метод. Details Метод приведен ниже.

public ActionResult Details(int? id)

{

if (id == null)

{

return new HttpStatusCodeResult(HttpStatusCode.BadRequest);

}

Movie movie = db.Movies.Find(id);

if (movie == null)

{

return HttpNotFound();

}

return View(movie);

}

Если Movie находится экземпляр Movie модель передается Details представления:

return View(movie);

Views\Movies\Details.cshtml

@model MvcMovie.Models.Movie

@{

ViewBag.Title = "Details";

}

<h2>Details</h2>

<div>

<h4>Movie</h4>

<hr />

<dl class="dl-horizontal">

<dt>

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

</dt>

@\*Markup omitted for clarity.\*@

</dl>

</div>

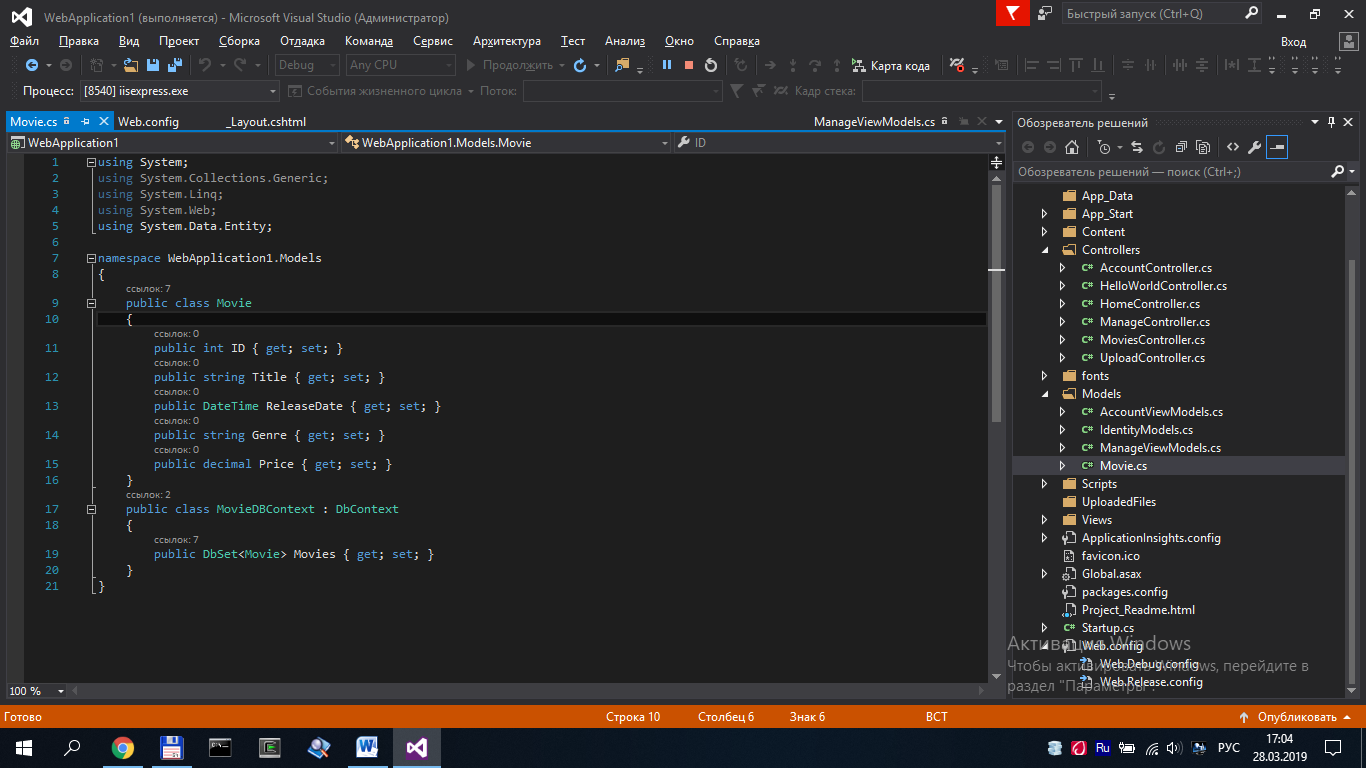
<p>

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

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

</p>

В этом разделе мы рассмотрим созданный Edit методы действий и представления для контроллера movie. Но сначала потребуется короткий интересную вносить поиска лучше даты выпуска. Откройте Models\Movie.cs файл и добавьте указанные ниже выделенные строки:



using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Data.Entity;

namespace WebApplication1.Models

{

public class Movie

{

public int ID { get; set; }

public string Title { get; set; }

public DateTime ReleaseDate { get; set; }

public string Genre { get; set; }

public decimal Price { get; set; }

}

public class MovieDBContext : DbContext

{

public DbSet<Movie> Movies { get; set; }

}

}