МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

**ФГАОУ ВПО "КАЗАНСКИЙ (ПРИВОЛЖСКИЙ) ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ"**

**ИНСТИТУТ ВЫЧИСЛИТЕЛЬНОЙ МАТЕМАТИКИ И ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ**

КАФЕДРА ПРИКЛАДНОЙ ИНФОРМАТИКИ

Допустить к защите

Зав.кафедрой \_\_\_\_\_\_\_\_\_\_\_\_

«\_\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_ 2015 г.

**ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА**

**Создание cистемы электронного документооборота «Сводный контролирующий реестр исполняемых документов» на основе веб-технологий**

Автор дипломной работы \_\_\_\_\_\_\_\_\_\_\_ <ФИО>

Специальность <специальность>

Группа <группа>

Научный руководитель \_\_\_\_\_\_\_\_\_\_\_\_\_ *<ФИО>*

«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_ 2015 г.

Казань 2015

СОДЕРЖАНИЕ

[ВВЕДЕНИЕ](#_Toc326499425) 4

[Глава 1. СРЕДСТВА И МЕТОДЫ СОЗДАНИЯ СИСТЕМ ЭЛЕКТРОННОГО ДОКУМЕНТООБОРОТА](#_Toc326499431) 6

[1.1. Рынок систем электронного документооборота в России](#_Toc326499432) 6

[1.2. Обзор средств создания систем электронного документооборота](#_Toc326499432) 7

[1.2.1. bpm^online 7.6](#_Toc326499432) 7

[1.2.2. Такском-Файлер](#_Toc326499432) 8

[1.2.3. Электронные офисные системы](#_Toc326499432) 9

[1.2.4. Барс Груп](#_Toc326499432) 11

[Глава 2. СОЗДАНИЕ СИСТЕМЫ ЭЛЕКТРОННОГО ДОКУМЕНТООБОРОТА "СВОДНЫЙ КОНТРОЛИРУЮЩИЙ РЕЕСТР ИСПОЛНЯЕМЫХ ДОКУМЕНТОВ" 1](#_Toc326499431)9

[2.1. Обоснование выбора программной среды](#_Toc326499432) 19

[2.2. Возможности ASP.NET/MVC](#_Toc326499432) 20

[2.3. Руководство пользователя программой “СКРИД”](#_Toc326499432) 21

[2.3.1. Запуск программы](#_Toc326499432) 21

[2.3.2. Интерфейс. Общий вид окон](#_Toc326499432) 25

[2.3.3. Реестр документов](#_Toc326499432) 36

[2.3.4. Реестр пользователей](#_Toc326499432) 36

[2.3.4. Реестр назначенных документов](#_Toc326499432) 36

[2.4. Основные этапы создания документов](#_Toc326499432) 34

[2.4.1. Создание новой записи о файле](#_Toc326499432) 34

[2.4.2. Добавление, перестановка, удаление свойств загруженного документа](#_Toc326499432) 34

[2.4.3. Открытие документов](#_Toc326499432) 40

[Глава 3. АНАЛИЗ ПРОГРАММНОГО КОДА](#_Toc326499437) 41

[3.1. Примеры функций из контроллера DocumentsController.cs](#_Toc326499438) 41

[3.2. Примеры функция из представления Create.cshtml.](#_Toc326499438) 42

[ЗАКЛЮЧЕНИЕ](#_Toc326499446) 63

[СПИСОК ЛИТЕРАТУРЫ](#_Toc326499447) 66

[ПРИЛОЖЕНИЯ](#_Toc326499448) 70

**ВВЕДЕНИЕ**

Система автоматизации документооборота, система электронного документооборота (СЭДО) — автоматизированная многопользовательская система, сопровождающая процесс управления работой иерархической организации с целью обеспечения выполнения этой организацией своих функций. При этом предполагается, что процесс управления опирается на человеко-читаемые документы, содержащие инструкции для сотрудников организации, необходимые к исполнению.

Основные понятия электронного документооборота:

Документооборот — движение документов в организации с момента их создания или получения до завершения исполнения или отправления (ГОСТ Р 51141-98 (отменен, взамен него ГОСТ Р 7.0.8-2013 с 01.03.2014 г.)); комплекс работ с документами: приём, регистрация, рассылка, контроль исполнения, формирование дел, хранение и повторное использование документации, справочная работа.

Электронный документооборот (ЭДО) — единый механизм по работе с документами, представленными в электронном виде, с реализацией концепции «безбумажного делопроизводства».

Машиночитаемый документ — документ, пригодный для автоматического считывания содержащейся в нём информации, записанный на магнитных, оптических и других носителях информации.

Электронный документ (ЭД) — документ, созданный с помощью средств компьютерной обработки информации, который может быть подписан электронной подписью (ЭП) и сохранён на машинном носителе в виде файла соответствующего формата.

Электронная подпись (ЭП) — аналог собственноручной подписи, являющийся средством защиты информации, обеспечивающим возможность контроля целостности и подтверждения подлинности электронных документов.

Основные принципы электронного документооборота

Однократная регистрация документа, позволяющая однозначно идентифицировать документ.

Возможность параллельного выполнения операций, позволяющая сократить время движения документов и повышения оперативности их исполнения

Непрерывность движения документа, позволяющая идентифицировать ответственного за исполнение документа (задачи) в каждый момент времени жизни документа (процесса).

Единая (или согласованная распределённая) база документной информации, позволяющая исключить возможность дублирования документов.

Эффективно организованная система поиска документа, позволяющая находить документ, обладая минимальной информацией о нём.

Развитая система отчётности по различным статусам и атрибутам документов, позволяющая контролировать движение документов по процессам документооборота и принимать управленческие решения, основываясь на данных из отчётов.

Классификация систем электронного документооборота

1. Универсальные «коробочные» СЭДО:

* стандартный набор функций;
* невозможность полного соответствия потребностям конкретной организации;
* низкие временные затраты на приобретение и установку;
* относительно низкая стоимость;
* необходимость приобретения лицензии на каждое внедряемое рабочее место.

1. Индивидуально разрабатываемые СЭДО:

* максимально персонифицированная система;
* большие временные затраты;
* высокая стоимость разработки;
* сопутствующие расходы: затраты на обучение сотрудников, покупку нового оборудования и программного обеспечения.

1. Комбинированные СЭДО:

* базовая платформа, к которой разрабатываются необходимые дополнительные модули;
* полное соответствие нуждам предприятия;
* небольшие временные затраты на разработку и внедрение;
* стоимость включает: цену базовой платформы и стоимость индивидуальной доработки, зависящей от сложности заказа;
* передача заказчику прав на продукт;
* простота освоения и использования;
* полная локализация;
* удобный интерфейс;
* взаимодействие с существующими офисными приложениями.

Целью данной дипломной работы является разработка системы электронного документооборота «Сводный контролирующий реестр исполняемых документов».

Для достижения поставленной цели определены следующие основные задачи:

1. рассмотреть теоретические основы создания приложения(сайта) с использованием Visual Studio 2015 на языке C#(технологии ASP.NET+MVC)
2. определить инструментарий для решения задачи;
3. разработать систему электронного документооборота «Сводный контролирующий реестр исполняемых документов»

Объектом исследования является автоматизация процессов управления и контроля на каком-нибудь промышленном предприятии, использующем данную систему электронного документооборота, что лучшим образом скажется на уровне реализации человеческих ресурсов

Предметом исследования является разработка системы электронного документооборота «Сводный контролирующий реестр исполняемых документов»

Теоретическую и методологическую основу данной работы составляют работы отечественных и зарубежных авторов в области программирования, проектирования и разработки электронных учебных ресурсов.

Работа состоит из трех глав.

В первой главе рассматриваются средства и системы создания электронного документооборота, анализируется проблема исследования и актуальность применения данных СЭД на промышленном уровне.

Во второй главе описываются методы и способы создания приложения, а также разработано руководство пользователя для работы в данном сервисе.

В третьей главе раскрывается содержание разработанной СЭД и перечисляются возможности работы с ним. Описывается возможности как применения, так и расширения данного приложения.

# ГЛАВА 1. СРЕДСТВА И МЕТОДЫ СОЗДАНИЯ СИСТЕМ ЭЛЕКТРОННОГО ДОКУМЕНТООБОРОТА

# 1.1. Рынок систем электронного документооборота в России

# По данным агентства International Data Corporation, на которые ссылается издание PC Week[1], в 2012 году затраты компаний на внедрение СЭД выросли на 20 % по сравнению с 2011 годом, а объем продаж тиражных программных продуктов — на 13 %. Согласно обзору аналитического агентства TAdviser, представленном в материале Cnews, российский рынок СЭД (ЕСМ) и услуг по их внедрению в 2012 году составил в объеме 25,6 млрд рублей, а ежегодный рост рынка составляет от 20 до 35 процентов. На российском рынке представлены как отечественные системы, созданные внутри страны, так и решения на базе крупнейших СЭД (на западе применяется аббревиатура EDMS) и ЕСМ-систем (IBM Lotus — Notes, Microsoft SharePoint и др.).

# Лидером по количеству реализованных проектов внедрения СЭД является продукт «Дело» компании «Электронные офисные системы» (49 процентов проектов). К числу крупнейших проектов внедрения систем электронного документооборота в коммерческой сфере в 2012—2013 году в обзоре отнесены внедрения СЭД «Дело» в Центробанке (60 тысяч пользователей), внедрение EMC Documentum в Ростелекоме (56 тыс. сотрудников), проект по внедрению DocsVision в Сбербанке (50 тыс. пользователей). К крупнейшим интеграторам СЭД авторы обзора отнесли компании «Логика бизнеса», «КРОК инкорпорейтед», «ИнтерТраст» (CompanyMedia), ЭОС и все ее партнеры (СЭД Дело и EOS for Sharepoint), «Парма-Телеком» (на базе SAP /R3) и «Хоулмонт» (на базе СЭД ТЕЗИС).

# По данным аналитического отчета петербургской компании DSS-Consulting (на который ссылается издание IT-Bestsellers[3]) к числу наиболее внедряемых продуктов отечественного рынка СЭД относятся следующие: по числу проектов внедрений в 2012 году — СЭД Дело (25,3 % проектов), Directum (23,6 %), DocsVision (13,3 %), 1С: Документооборот (12,7 %), CompanyMedia (5,9 %); по числу автоматизированных рабочих мест — СЭД Дело (40,1 %), CompanyMedia (17,5 %), DocsVision (17,1 %), Directum (8 %). Еще один критерий — это количество новых внедрений в 2012 году, по данной позиции лидерами являются пять продуктов — СЭД Дело (42,1 %), Directum (28,2 %), DocsVision (9,3 %), Логика ЕСМ (БОСС-Референт) (6,1 %) и ТЕЗИС (4,3 %).

# Что касается государственной сферы, по данным отчета Министерства коммуникаций и связи[4], основанного на данных опроса 79 ведомств, в 2013 году СЭД были внедрены в 74 федеральных органах исполнительной власти, из них 4 органа (ФССП, ФНС, Министерства культуры и Министерства транспорта) используют несколько разных систем, а 17 ведомств подключены к межведомственным системам электронного документооборота (23 % всех органов), а три из них так же подключены к элементам электронного правительства. Не были внедрены системы электронного документооборота в ФСТЭК, ФСБ, СВР, Ростуризме и МВД.

# 1.2. Обзор средств создания систем электронного документооборота

# 1.2.1. bpm^online 7.6

Платформа bpm’online, созданная для управления бизнес-процессами продаж, маркетинга и сервиса, лидер по количеству баллов в категории сильнейших разработчиков мира в отчете Forrester Wave. В 2014 году издание CRM Magazine признало продукт bpm’online sales мировым лидером на рынке программного обеспечения для управления продажами. А в 2015 году bpm’online включена в рейтинг лучших CRM-систем для управления клиентским сервисом — Магический Квадрант Gartner.

6000 средних и крупных предприятий используют платформу bpm’online для автоматизации своих бизнес-процессов.

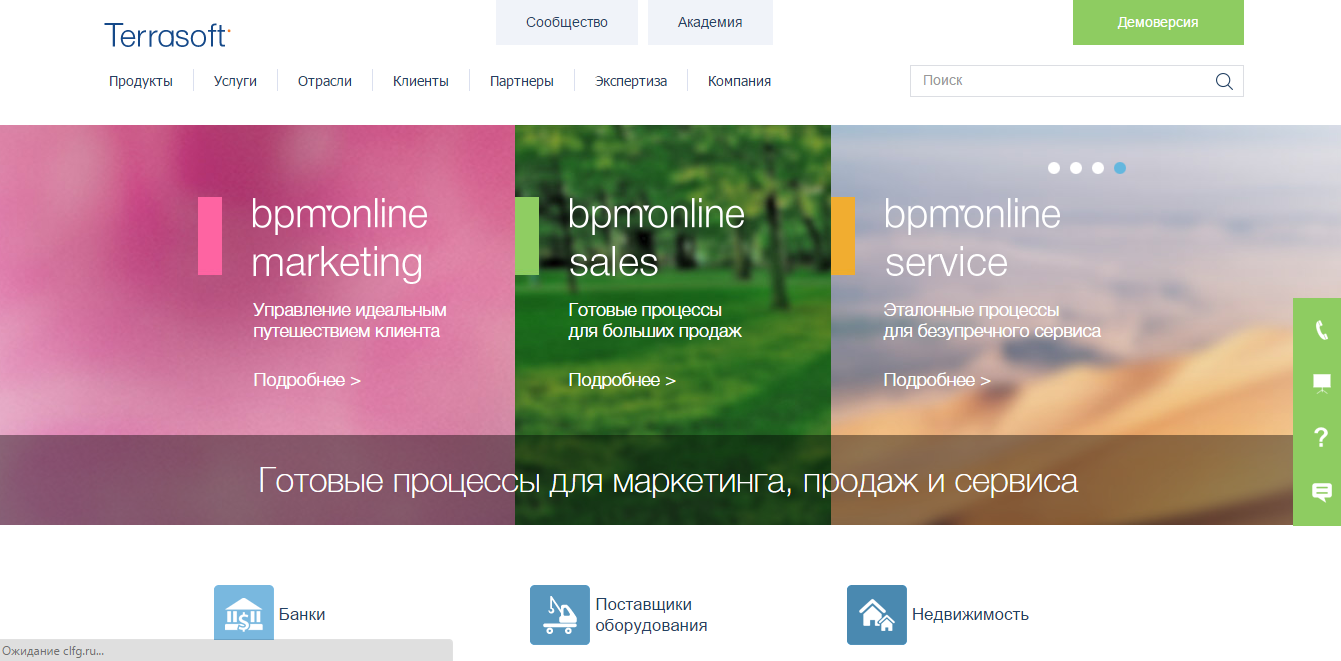


Рис. 1.1 – Сайт компании «Terrasoft»

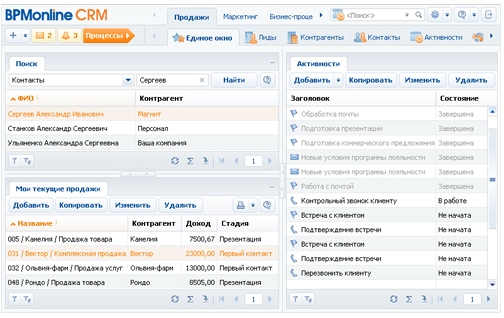


Рис. 1.2 – Запущенная программа bpm^online 7.6

**1.2.2. Такском-Файлер**

Компания «Такском» с 2000 года специализируется на разработке, внедрении и сопровождении систем электронного документооборота, обеспечивающих обмен по каналам связи конфиденциальной информацией, защищенной электронной цифровой подписью и СКЗИ. Подобные системы применяются для представления отчетности в контролирующие органы и фонды, а также для электронного документооборота между организациями и ведомствами.

За годы успешного существования на рынке IT-технологий компания «Такском»:

* совместно с Минфином России и ФНС России по поручению Правительства РФ провела ряд пилотных проектов, направленных на разработку и внедрение в России систем защищенного электронного документооборота (ЭДО), на основе которых вывела на рынок комплекс уникальных программных продуктов (отчетность через Интернет, электронные счета-фактуры и т.п.);
* разработала и внедрила в промышленную эксплуатацию систему сдачи отчетности «Такском-Спринтер»;
* создала крупнейший Центр сертификации ключей ЭЦП, который оказывает услуги в получении и замене ЭЦП для систем ЭДО и для участников электронных торгов;
* открыла корпоративный Учебный центр для абонентов и партнеров компании, сформировала многочисленный профессиональный коллектив;
* стала лауреатом Национальной бизнес-премии «Компания года 2009» в номинации «IT-компания»;
* получила звание «Лучшая компания российского рынка программных продуктов в области бухгалтерского учета, аудита и налогообложения».

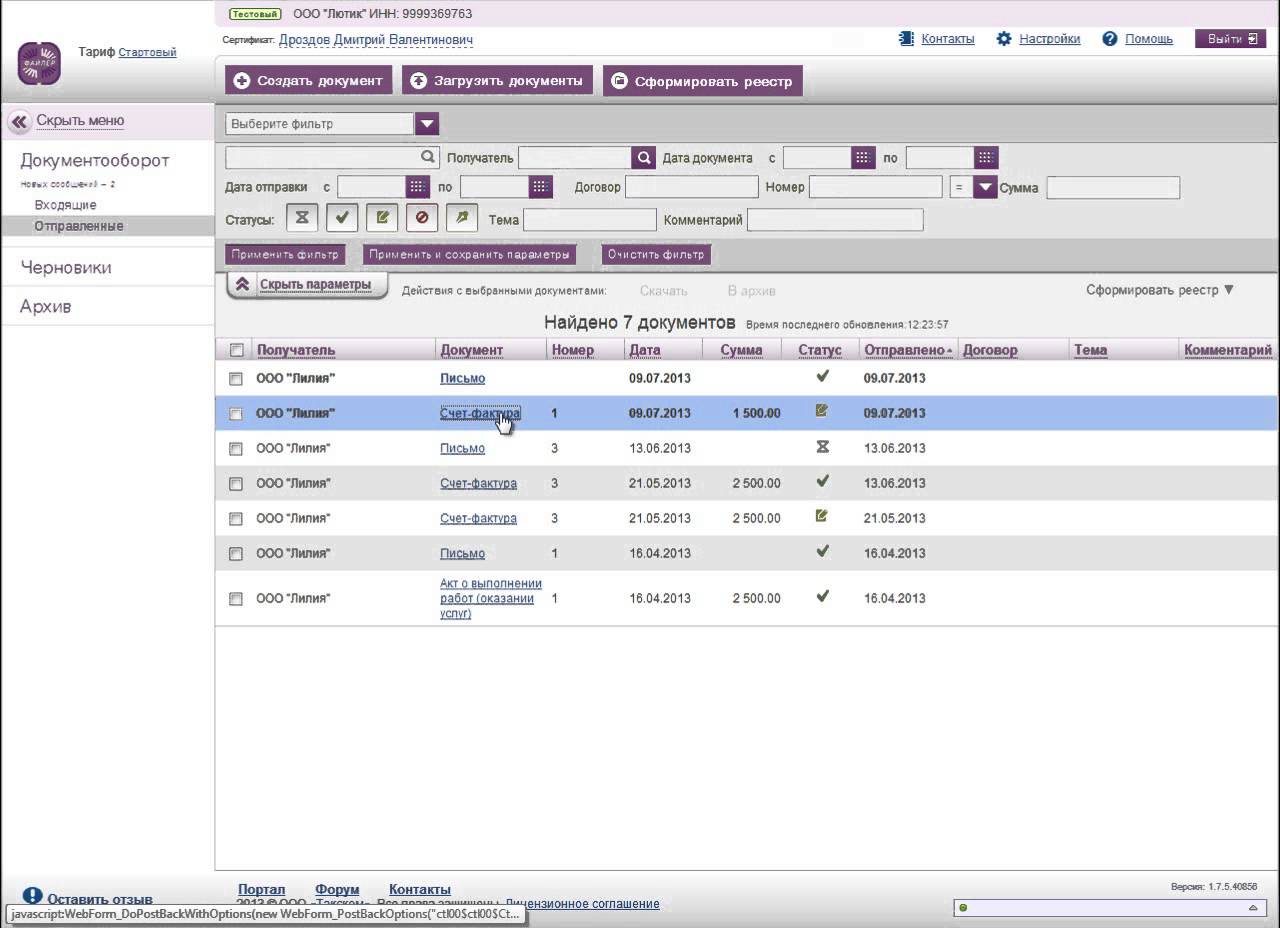


Рис. 1.3 – Программа «Такском-файлер»

* + 1. **Барс Груп**

Работает на рынке информационных технологий с 1992 г. Компания известна по работе в сфере информатизации российского государственного сектора. Была одной из первых российских IT-компаний, которые стали создавать государственные информационные системы на основе облачных технологий. Большой опыт реализации проектов и высокое качество программных продуктов позволили «БАРС Груп» совершить выход на сегмент «бизнес для бизнеса» и международный рынок (СНГ, Ближний Восток, Юго-Восточная Азия). Имеет государственную аккредитацию в сфере информационных технологий, сертификат ИСО 9001-2008 о менеджменте качества, лицензию ФСТЭК по технической защите конфиденциальной информации. По не уточненным данным до осени 2012 года в группу компаний входили: компания «БАРС Груп», «Компания БАРС», «Центр методологии и автоматизации бюджетного учета», Центр информационных технологий «ПРОКСИМА» (автоматизация автономных учреждений)

Компания включает в себя следующие направления деятельности: «БАРС.Государственное управление», «БАРС.Муниципальное управление», «БАРС.Бюджет», «БАРС.Здравоохранение», «БАРС.Образование», «БАРС.ЖКХ», «БАРС.Социальная защита», «БАРС.Стройкомплекс», «БАРС.Сельское хозяйство», «БАРС.Портал»

В числе крупнейших проектов «БАРС Груп» создание системы бюджетного планирования в Министерстве финансов РФ и ряда других функциональных блоков «Электронного бюджета». Системами и сервисами «БАРС Груп» пользуются такие крупные заказчики как МЧС России, Минкомсвязь России, ГК «Фонд содействия реформированию ЖКХ», Департамент строительства г. Москвы. Многие проекты обладают высокой социальной значимостью. К таким проектам можно отнести создание региональных фрагментов Единой государственной информационной системы в сфере здравоохранения (Калининградская, Костромская, Новосибирская, Орловская, Саратовская и Томская области), создание сервисов для школ, организацию электронной очередности в дошкольные учебные заведения, обеспечение открытости ЖКХ и др.

**«Облака»**

«БАРС Груп» является одним из пионеров «облачных» технологий в российском государственном секторе. Решения «БАРС Груп» – это web-приложения, которые функционируют в частных и открытых «облаках», а также имеют возможность интеграции в Национальную платформу распределённой обработки данных (НПРОД).

**Гибкость**

Все приложения компании «БАРС Груп» обладают инструментарием настройки под бизнес-процессы клиента. Функционал может модифицироваться без изменения исходного кода системы путём подключения плагинов. При этом нет необходимости привлекать специалистов компании-разработчика: все доработки осуществимы силами заказчика или компании-партнёра.

**Простота настройки**

Конфигурация приложений осуществляется в интуитивно понятном интерфейсе и не требует от специалиста навыков разработки.

**Масштабируемость**

Приложения «БАРС Груп» обеспечивают масштабируемость серверной части – как серверов приложений, так и серверов базы данных – различными способами балансировки нагрузки.

**Кроссплатформенность**

Рабочие места и серверы, на которых устанавливаются приложения «БАРС Груп», способны функционировать на основе любой распространённой операционной системой (Windows, Linux, MacOS). Также пользователи не ограничиваются в выборе СУБД и браузера в качестве сервера базы данных.

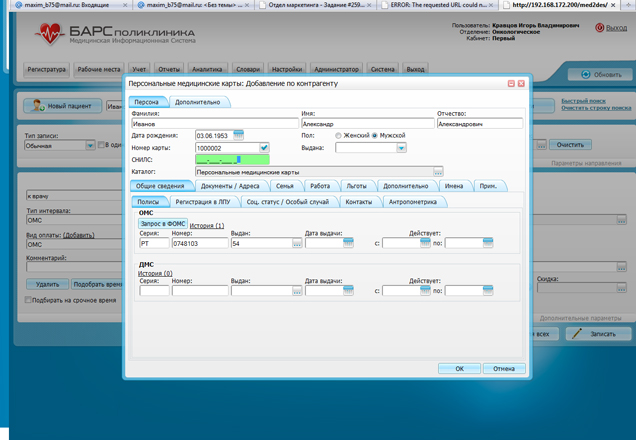


Рис. 1.4 – Пример одного из приложений системы «БАРС»: «БАРСполиклиника»

**ГЛАВА 2. СОЗДАНИЕ СИСТЕМЫ ЭЛЕКТРОННОГО ДОКУМЕНТООБОРОТА "СВОДНЫЙ КОНТРОЛИРУЮЩИЙ РЕЕСТР ИСПОЛНЯЕМЫХ ДОКУМЕНТОВ"**

# 2.1. Обоснование выбора программной среды

Microsoft Visual Studio — линейка продуктов компании Microsoft, включающих интегрированную среду разработки программного обеспечения и ряд других инструментальных средств. Данные продукты позволяют разрабатывать как консольные приложения, так и приложения с графическим интерфейсом, в том числе с поддержкой технологии Windows Forms, а также веб-сайты, веб-приложения, веб-службы как в родном, так и в управляемом кодах для всех платформ, поддерживаемых Windows, Windows Mobile, Windows CE, .NET Framework, Xbox, Windows Phone .NET Compact Framework и Silverlight.

Visual Studio включает в себя редактор исходного кода с поддержкой технологии IntelliSense и возможностью простейшего рефакторинга кода. Встроенный отладчик может работать как отладчик уровня исходного кода, так и как отладчик машинного уровня. Остальные встраиваемые инструменты включают в себя редактор форм для упрощения создания графического интерфейса приложения, веб-редактор, дизайнер классов и дизайнер схемы базы данных. Visual Studio позволяет создавать и подключать сторонние дополнения (плагины) для расширения функциональности практически на каждом уровне, включая добавление поддержки систем контроля версий исходного кода (как, например, Subversion и Visual SourceSafe), добавление новых наборов инструментов (например, для редактирования и визуального проектирования кода на предметно-ориентированных языках программирования) или инструментов для прочих аспектов процесса разработки программного обеспечения (например, клиент Team Explorer для работы с Team Foundation Server).Ниже представлено окно работы программы (Рис. 2.1).

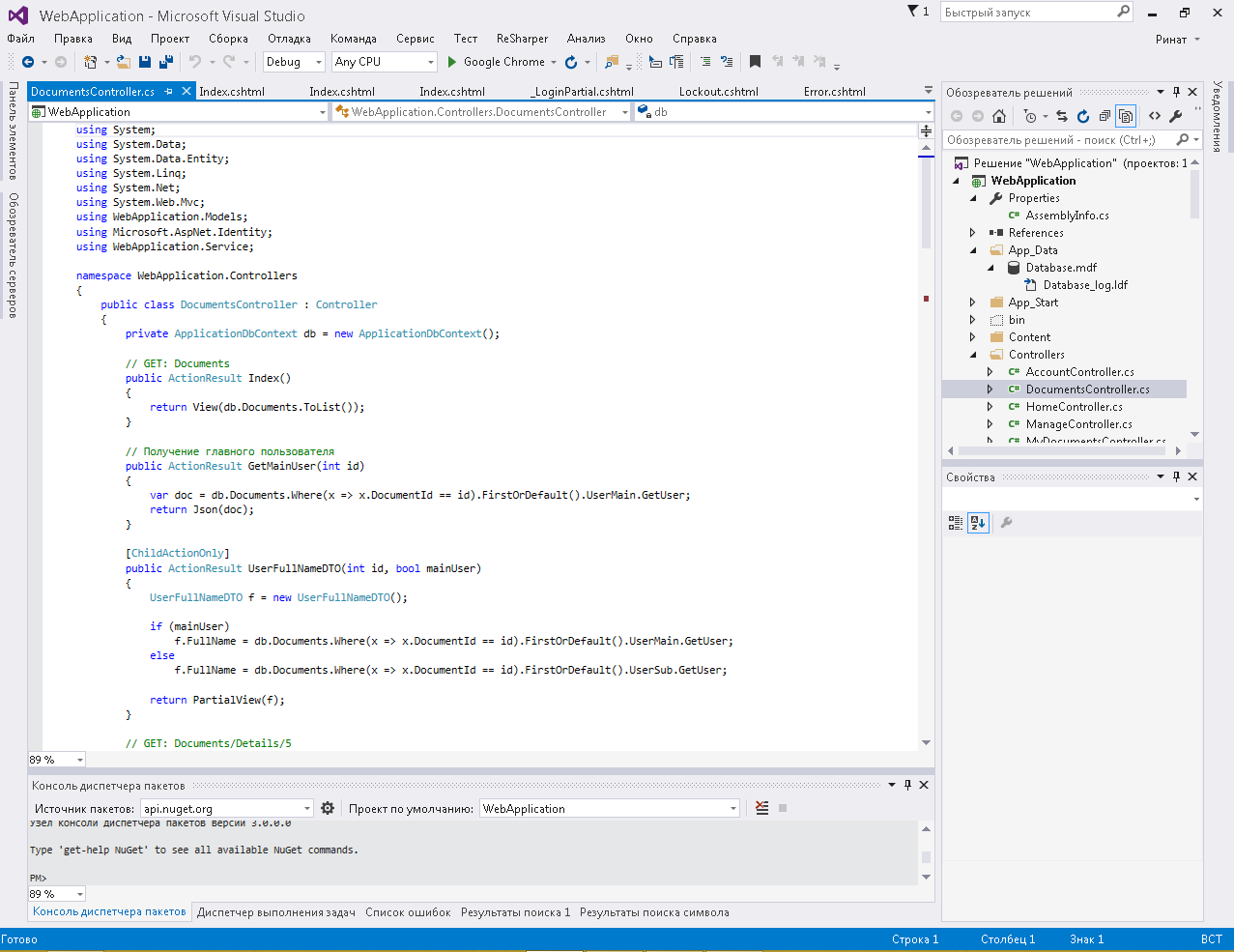


Рис. 2.1 – Работа в программе iBooks Author.

ASP.NET MVC Framework — фреймворк для создания веб-приложений, который реализует шаблон Model-view-controller. Данный фреймворк добавлен Microsoft в ASP.NET.

В апреле 2009 года исходный код ASP.NET MVC был опубликован под лицензией Microsoft Public License (MS-PL)[1]. 27 марта 2012 года лицензия была изменена на Apache License 2.0.

Платформа ASP.NET MVC базируется на взаимодействии трех компонентов: контроллера, модели и представления. Контроллер принимает запросы, обрабатывает пользовательский ввод, взаимодействует с моделью и представлением и возвращает пользователю результат обработки запроса.

Модель представляет слой, описывающий логику организации данных в приложении. Представление получает данные из контроллера и генерирует элементы пользовательского интерфейса для отображения информации.

**Движок представлений**

Для управления разметкой и вставками кода в представлении используется движок представлений. До версии MVC 5 использовались два движка:

Web Forms и Razor. Начиная с MVC 5 единственным движком, встроенным по умолчанию, является Razor. Движок WebForms использует файлы .aspx, а Razor - файлы .cshtml и .vbhtml для хранения кода представлений. Основой синтаксиса Razor является знак @, после которого осуществляется переход к коду на языках C#/VB.NET. Также возможно и использование сторонних движков. Файлы представлений не являются стандарными статическими страницами с кодом html, а в процессе генерации контроллером ответа с использованием представлений компилируются в классы, из которых затем генерируется страница html.

**Маршрутизация**

При обработке запросов фреймворк ASP.NET MVC опирается на систему маршрутизации, которая сопоставляет все входящие запросы с определенными в системе маршрутами, которые указывают какой контроллер и метод должен обработать данный запрос. Встроенный маршрут по умолчанию предполагает трехзвенную структуру: контроллер/действие/параметр.

ASP.NET MVC и ASP.NET Web Forms являются двумя родственными технологиями, в основании которых лежит одна платформа ASP.NET. И все же ASP.NET MVC имеет ряд преимуществ перед ASP.NET Web Forms[24]:

* разделение ответственности (отдельная разработка разных компонентов - контроллера, моделей, представлений)
* улучшенная тестируемость
* соответствие протоколу HTTP (ASP.NET MVC не поддерживает объект ViewState)
* повышенная гибкость и настраиваемость под собственные нужды

**2.2. Возможности ASP.NET/MVC**

Следует внимательно продумать вопрос о создании веб-приложения на основе платформы ASP.NET MVC или на основе модели веб-форм ASP.NET. Платформа MVC не заменяет собой модель веб-форм. Обе модели можно использовать для веб-приложений. (при наличии существующих приложений на основе веб-форм они будут продолжать работу в нормальном режиме).

Перед использованием платформы MVC или модели веб-форм для определенного веб-сайта следует взвесить все преимущества каждого из подходов.

Платформа ASP.NET MVC имеет следующие преимущества.

Она облегчает управление сложными структурами путем разделения приложения на модель, представление и контроллер.

Она не использует состояние просмотра и серверные формы. Это делает платформу MVC идеальной для разработчиков, которым необходим полный контроль над поведением приложения.

Она использует схему основного контроллера, при которой запросы веб-приложения обрабатываются через один контроллер. Это позволяет создавать приложения, поддерживающие расширенную инфраструктуру маршрутизации. Дополнительные сведения см. в разделе Основной контроллер.

Она обеспечивает расширенную поддержку разработки на основе тестирования.

Она хорошо подходит для веб-приложений, поддерживаемых крупными коллективами разработчиков, а также веб-разработчикам, которым необходим высокий уровень контроля над поведением приложения.

Платформа на основе веб-форм имеет следующие преимущества.

Она поддерживает модель событий, которая сохраняет состояние при передаче через HTTP, что облегчает разработку бизнес веб-приложений. Приложение на основе веб-форм предоставляет множество событий, поддерживаемых различными серверными элементами управления.

Она использует шаблон контроллера страницы, добавляющий функции к отдельным страницам. Дополнительные сведения см. в разделе Контроллер страницы.

Она использует состояние просмотра для серверных форм, что может облегчить управление информацией о состоянии.

Она подходит для небольших коллективов веб-разработчиков, которым необходимо использовать большое количество компонентов для быстрого развертывания приложений.

В общем случае эта платформа менее сложна для разработки приложений, так как все компоненты (класс Page, элементы управления и т. д.) тесно интегрированы и требуют меньшего объема кода, чем в модели MVC.

Платформа ASP.NET MVC предоставляет следующие возможности.

Разделение задач приложения (логика ввода, бизнес-логика и логика пользовательского интерфейса), широкое возможности тестирования и разработки на основе тестирования. Все основные контракты платформы MVC основаны на интерфейсе и подлежат тестированию с помощью макетов объекта, которые имитируют поведение реальных объектов приложения. Приложение можно подвергать модульному тестированию без запуска контроллеров в процессе ASP.NET, что ускоряет тестирование и делает его более гибким. Для тестирования возможно использование любой платформы модульного тестирования, совместимой с .NET Framework.

Расширяемая и дополняемая платформа. Компоненты платформы ASP.NET MVC можно легко заменить или настроить. Разработчик может подключать собственный механизм представлений, политику маршрутизации URL-адресов, сериализацию параметров методов действий и другие компоненты. Платформа ASP.NET MVC также поддерживает использование моделей контейнера внедрения зависимости (DI) и инверсии элемента управления (IOC). Модель внедрения зависимости позволяет внедрять объекты в класс, а не ожидать создания объекта самим классом. Модель инверсии элемента управления указывает на то, что если один объект требует другой объект, то первые объекты должны получить второй объект из внешнего источника (например, из файла конфигурации). Это облегчает тестирование.

Расширенная поддержка маршрутизации ASP.NET. Этот мощный компонент сопоставления URL-адресов позволяет создавать приложения с понятными URL-адресами, которые можно использовать в поиске. URL-адреса не должны содержать расширения имен файлов и предназначены для поддержки шаблонов именования URL-адресов, обеспечивающих адресацию, оптимизированную для поисковых систем (SEO) и для передачи репрезентативного состояния (REST).

Поддержка использования разметки в существующих файлах страниц ASP.NET (ASPX), элементов управления (ASCX) и главных страниц (MASTER) как шаблонов представлений. Вместе с платформой ASP.NET MVC можно использовать существующие функции ASP.NET, например вложенные главные страницы, встроенные выражения (<%= %>), декларативные серверные элементы управления, шаблоны, привязку данных, локализацию и т. д.

Поддержка существующих функций ASP.NET. ASP.NET MVC позволяет использовать такие функции, как проверка подлинности с помощью форм и Windows, проверка подлинности по URL-адресу, членство и роли, кэширование вывода и данных, управление состоянием сеанса и профиля, наблюдение за работоспособностью, система конфигурации и архитектура поставщика.

**2.3. Руководство пользователя системой «СКРИД»**

«Сводный контролирующий реестр исполняемых документов» (в дальнейшем «СКРИД» был разработан мною как пример автоматизации процессов управления и контроля на каком-нибудь промышленном предприятии. Основная цель программы(в виде удаленного сервиса) – это контроль зарегистрированных пользователей и отправка им документации на исполнение.

Каждый пользователь может загружать документ, назначать ему исполнителя(установщик автоматически идентифицируется системой) и назначает дату исполнения. Он может как создавать документы, так и контролировать общий реестр загруженных документов, контролировать реестр зарегистрированных пользователей. Он так же видит реестр документов, назначенных лично на него. Каждый документ имеет три статуса: «Не выполнен», «На выполнении», «Выполнен». Тем самым, происходит контроль исполнения документа. Система не идеальна, но она всегда может быть доработана.

Для того, чтобы пользоваться СКРИД, пользователь должен вначале пройти простейшую систему регистрации: ввести свою данные и отправить их программе.

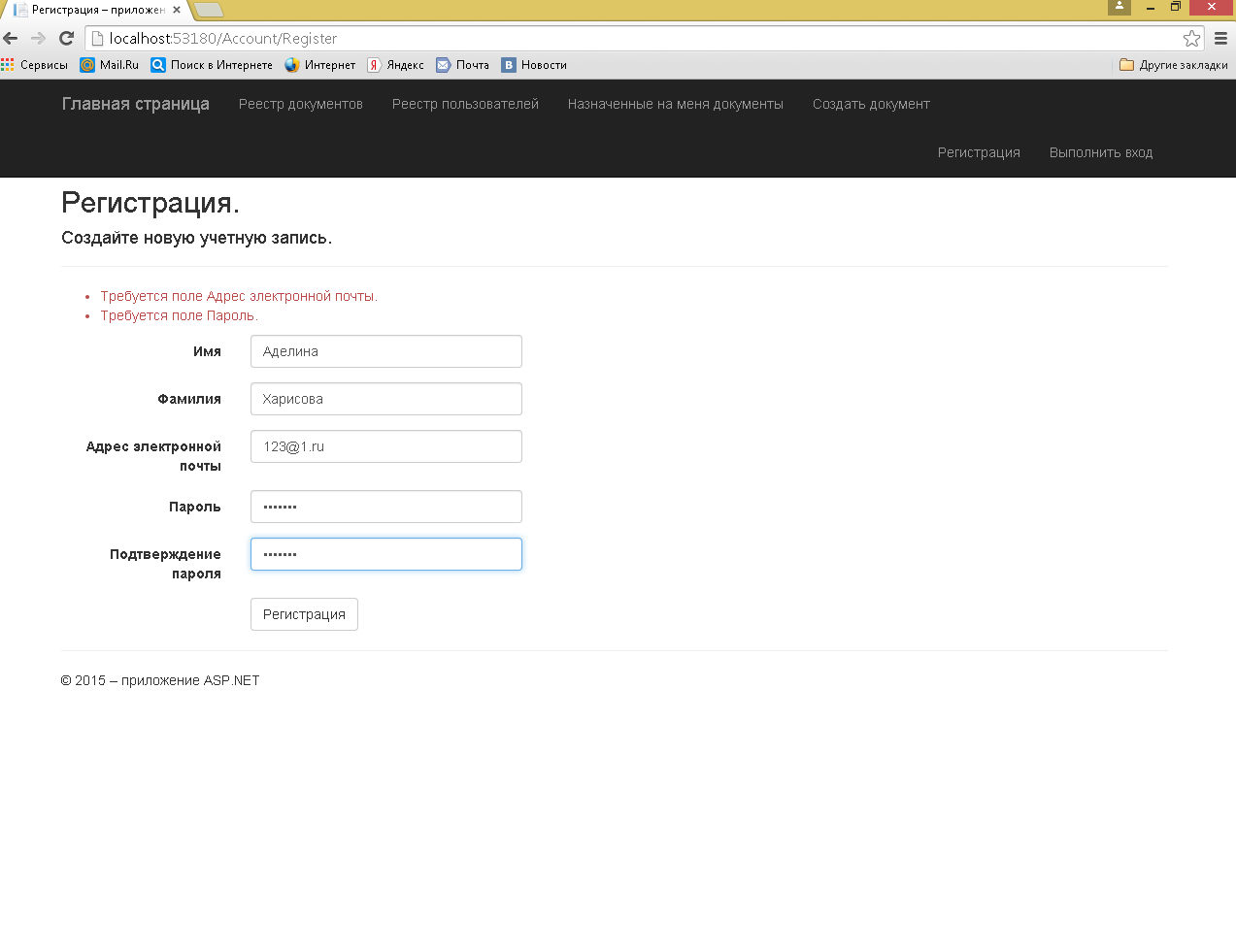


Рис. 2.2. – Регистрация пользователя

И после ввода данных данные о пользователе попадут в таблицу AspNetUsers:

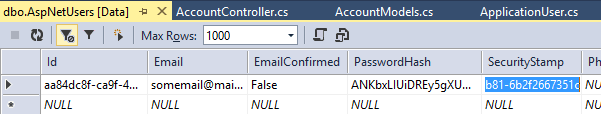


Рис. 2.2.1 – Запись о пользователе

### 2.3.1. Запуск программы

Для запуска программы либо разместить её на каком-нибудь хостинге и открыть как обычный сайт(но это дело платное, довольно дорогое для студента), либо поступить как разработчик: зайти в программу VisualStudio и открыть проект, нажать «Запустить в браузере». Студия скомпилирует код и запустит проект на локальном хостинге локального компьютера как рабочий сайт(Рис. 2.3).

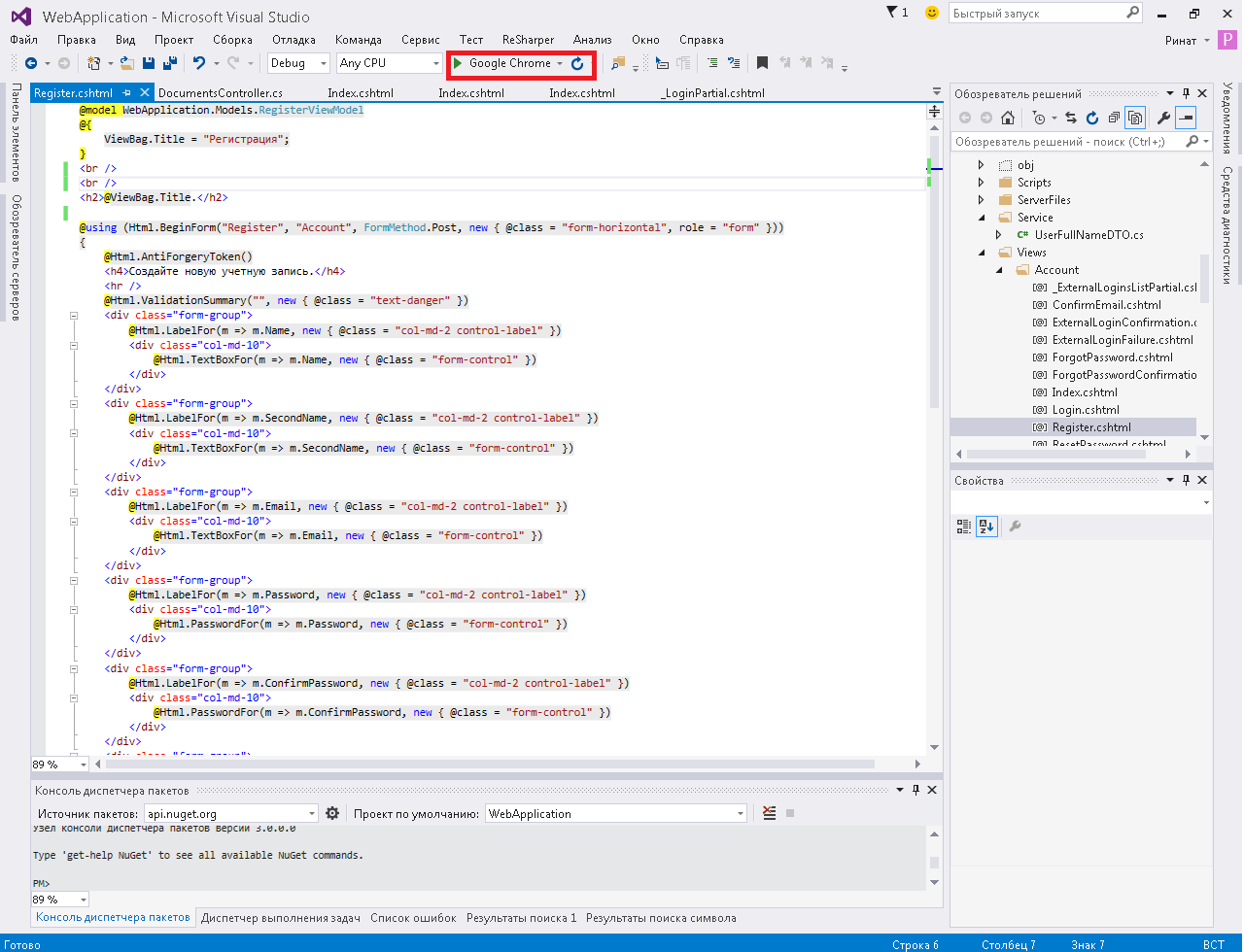


Рис. 2.3 – Запуск как локальном хостинге

После успешно пройденной регистрации открывается главное окно приложения и пользователь может полностью пользоваться системой.

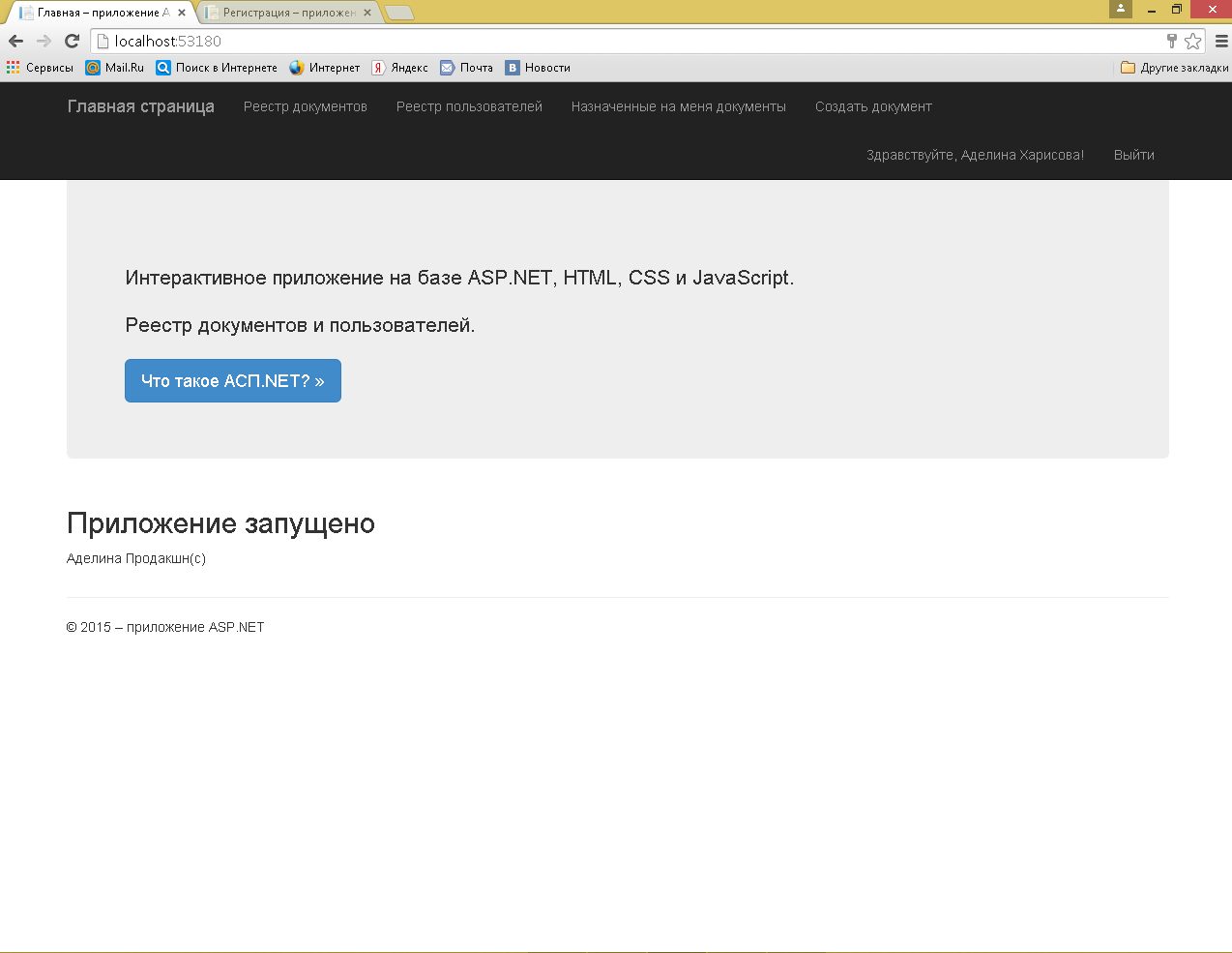


Рис. 2.4 – Рабочая среда приложения

### 2.3.2. Интерфейс. Общий вид окна

Основное окно «СКРИД» состоит из двух частей: основного меню сверху, которое как «слой» покрывает всё приложение и все его страницы, а так же – основного контента, расположенного посреди(это текст и поля ввода-управления, разнообразные таблицы).

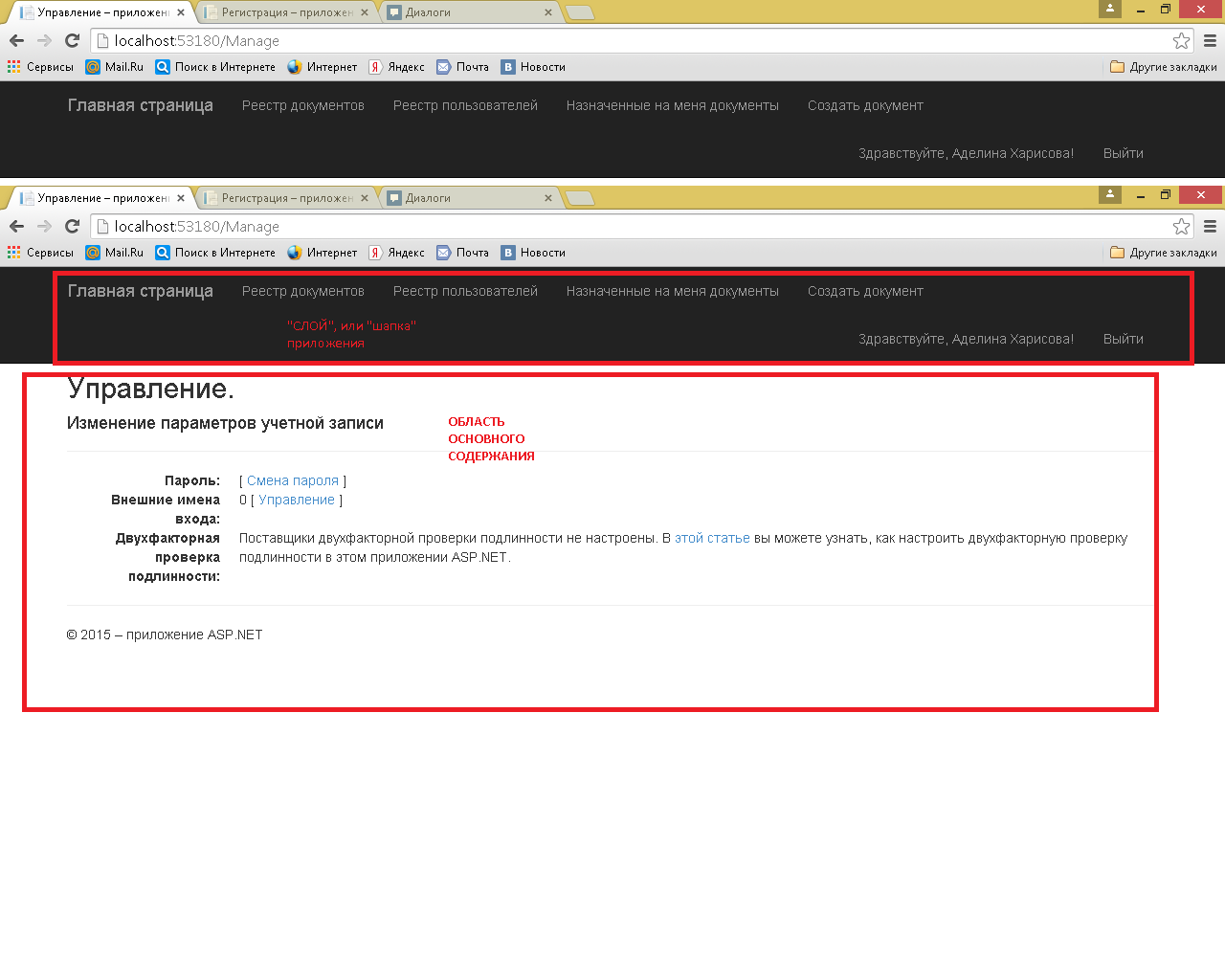


Рис. 2.5

#### 2.3.3. Реестр документов

Каждый добавленный документ отображается в общем реестр документов, который представлен в виде таблицы. Столбцы таблицы:

* Имя файла – содержит название файла
* Тип файла – типы могут быть самые разные(например doc, avi и т.п.)
* Дедлайн – срок, к которому задача должна быть исполнена
* Установщик – пользователь, который загрузил задание
* Исполнитель – пользователь, на которого назначено задание. Этот пользователь должен выполнить задание и отправить отчет установившему задание.
* Состояние – показывается текущее состояние файла(«Не принят на выполнение», «Выполняется», «Выполнен»)
* Ссылка на скачивание – позволяет скачать файл с сервера для просмотра на локальном компьютере
* Размер файла в байтах – показывает размер загруженного файла

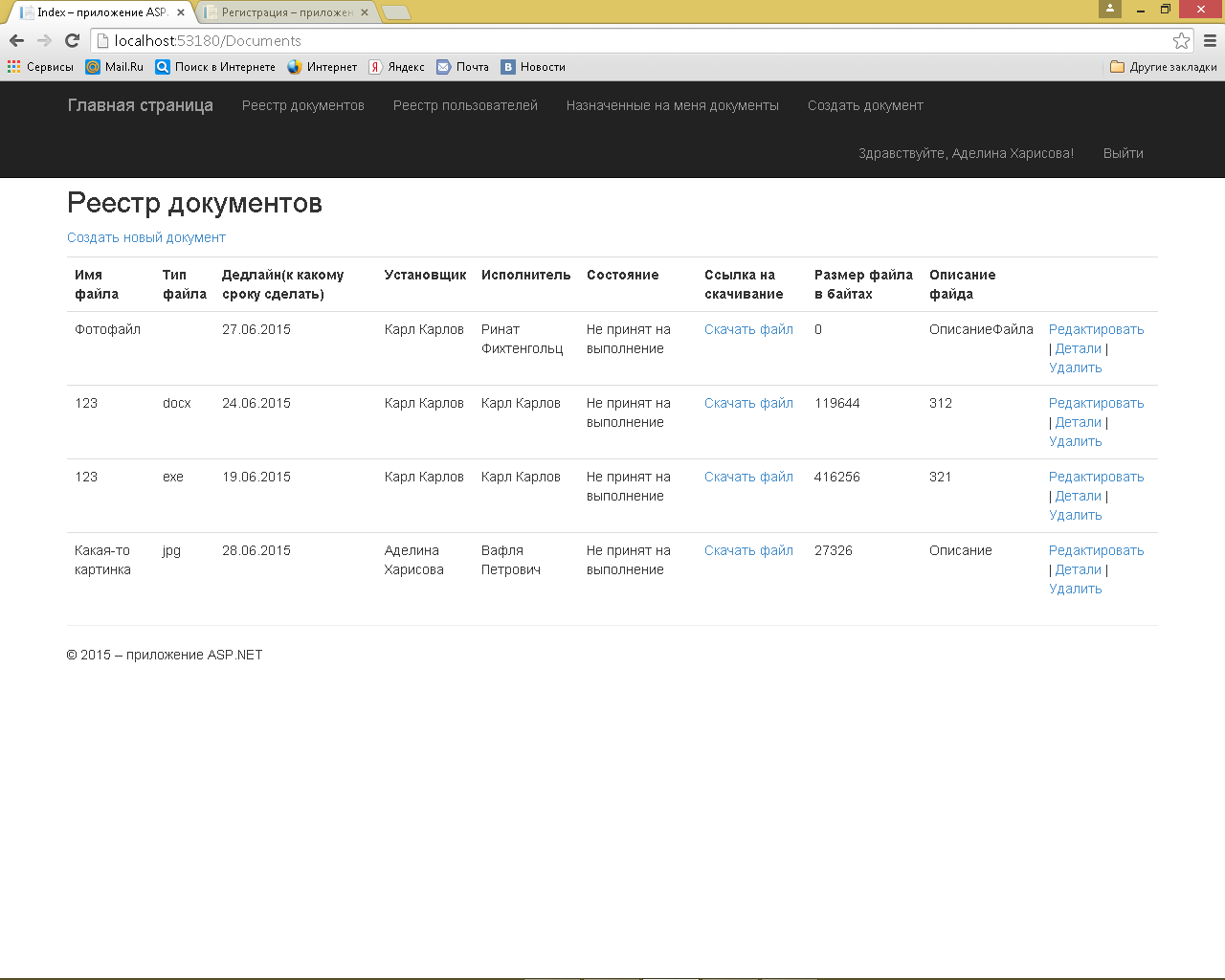


Рис. 2.6 – Реестр документов

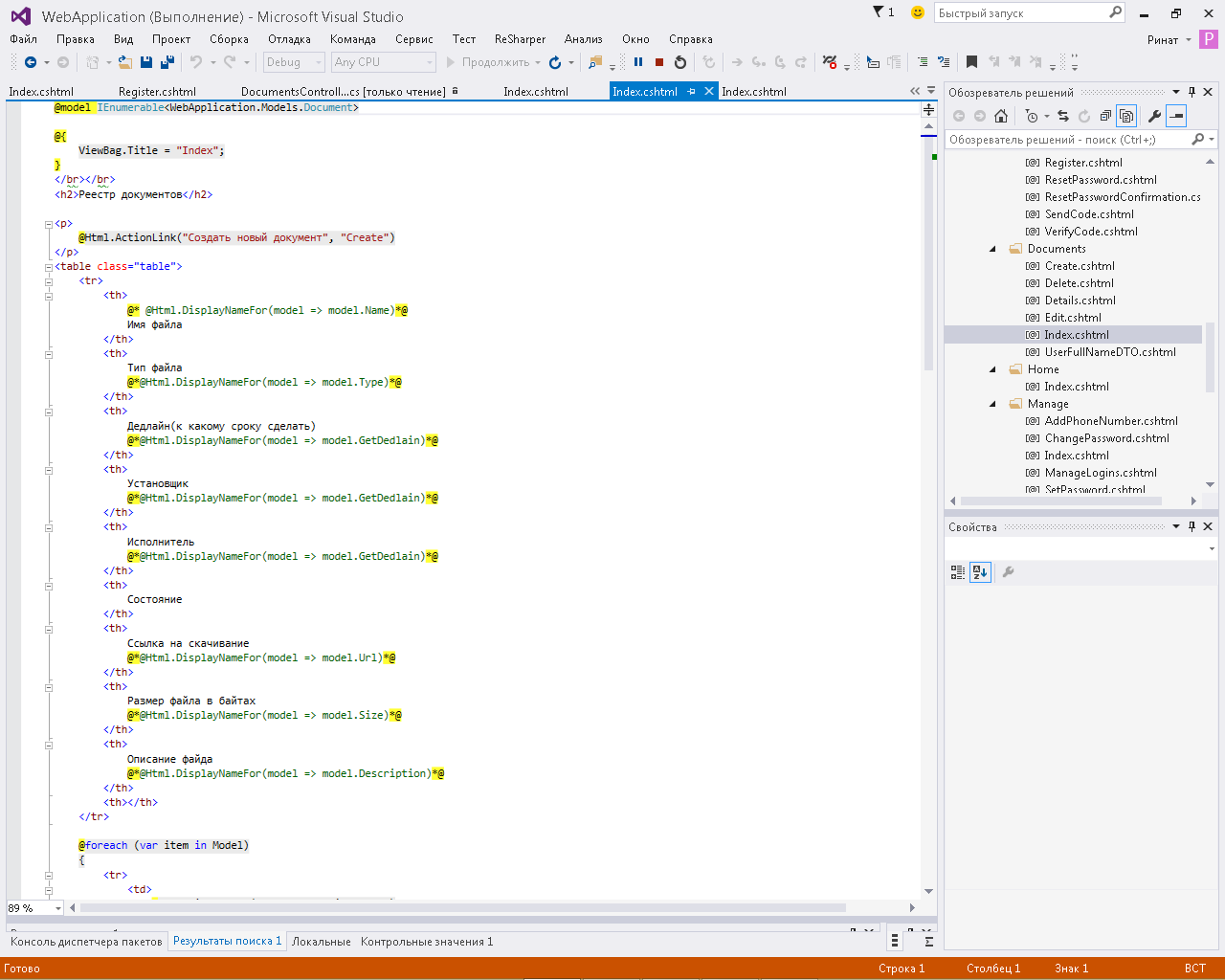


Рис. 2.7 – Код представления Документа

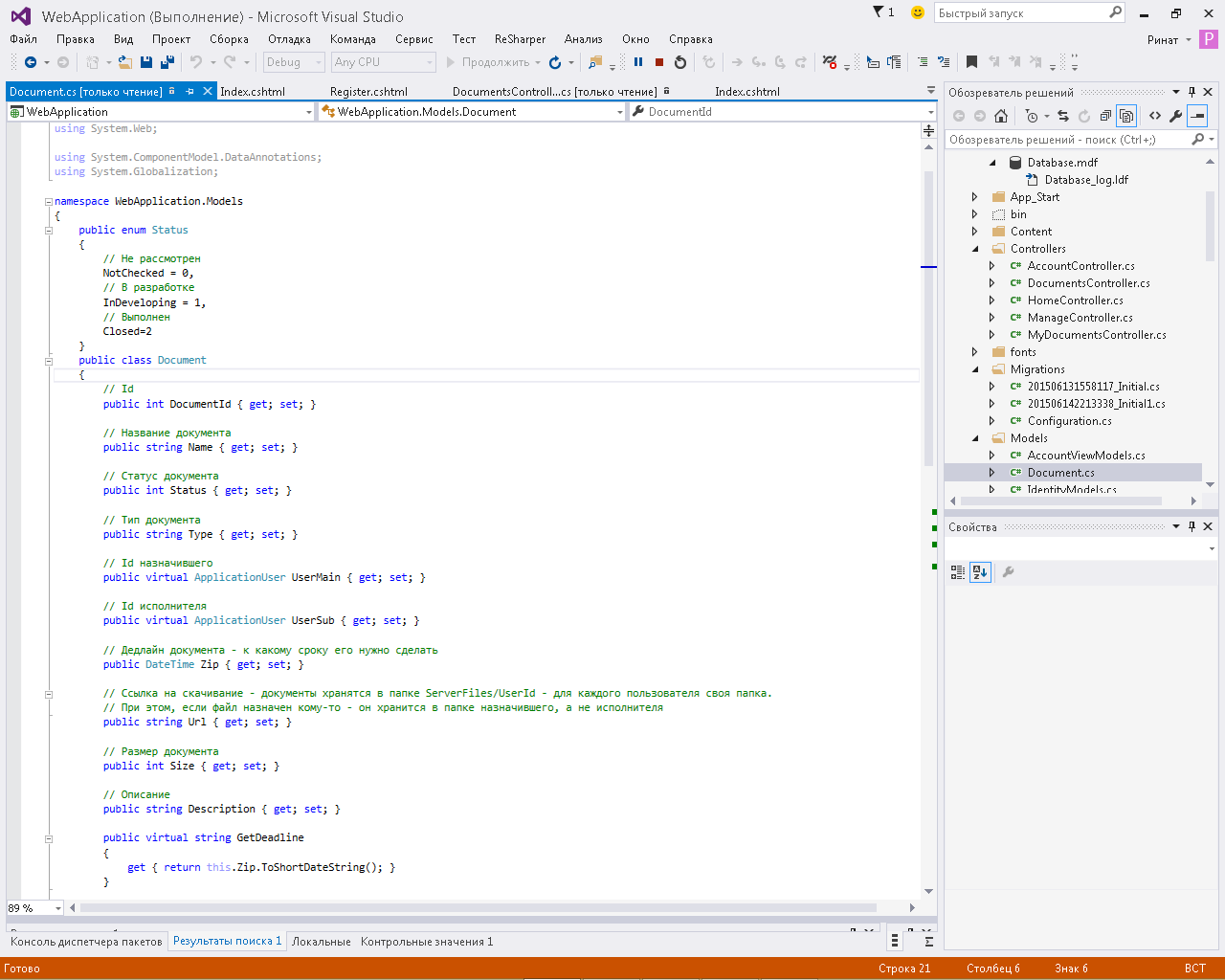


Рис. 2.8 – Код модели Документа

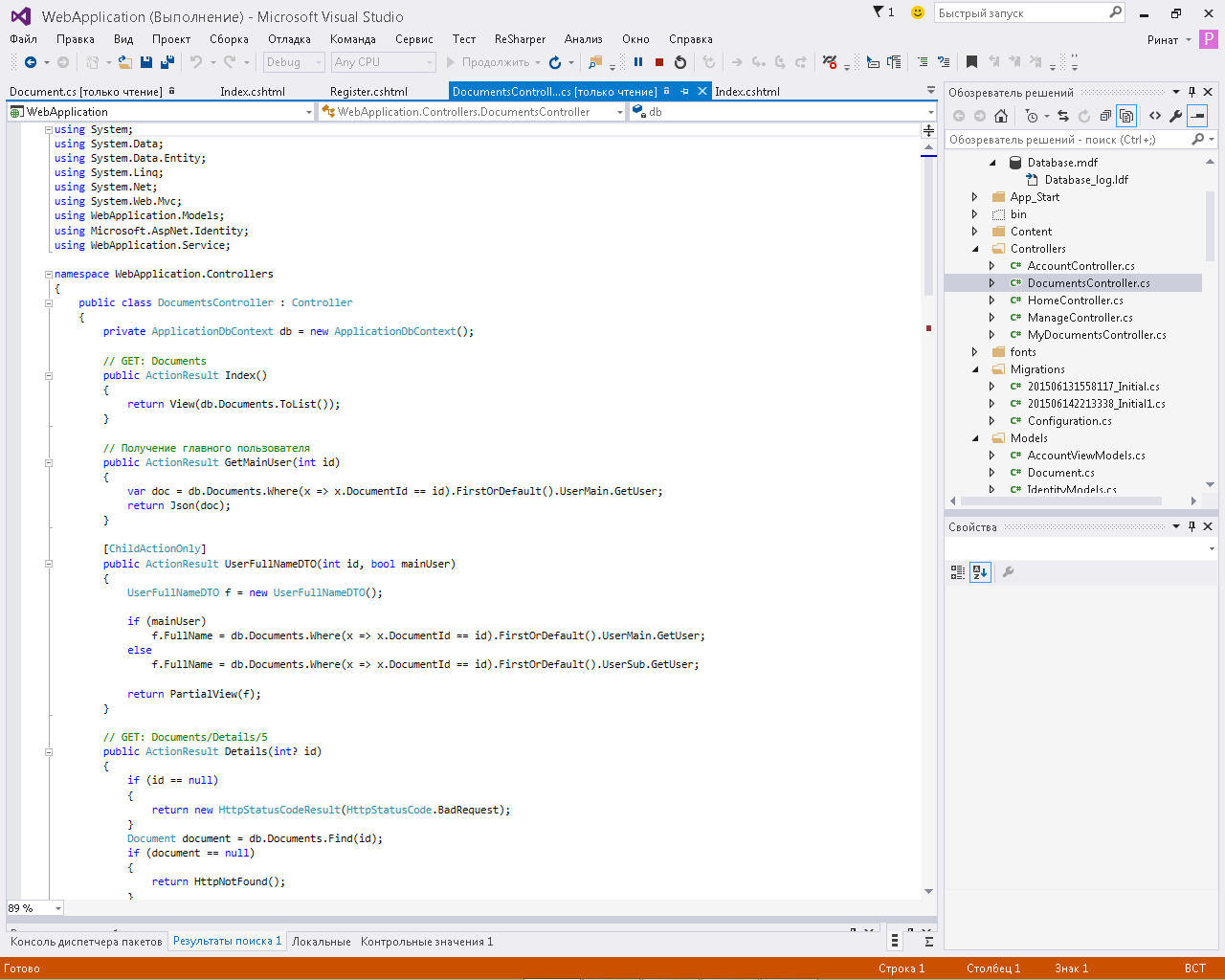


Рис. 2.9 – Код контроллера Документа

Общий вид сущности «Документ» представлен на рисунке 2.10

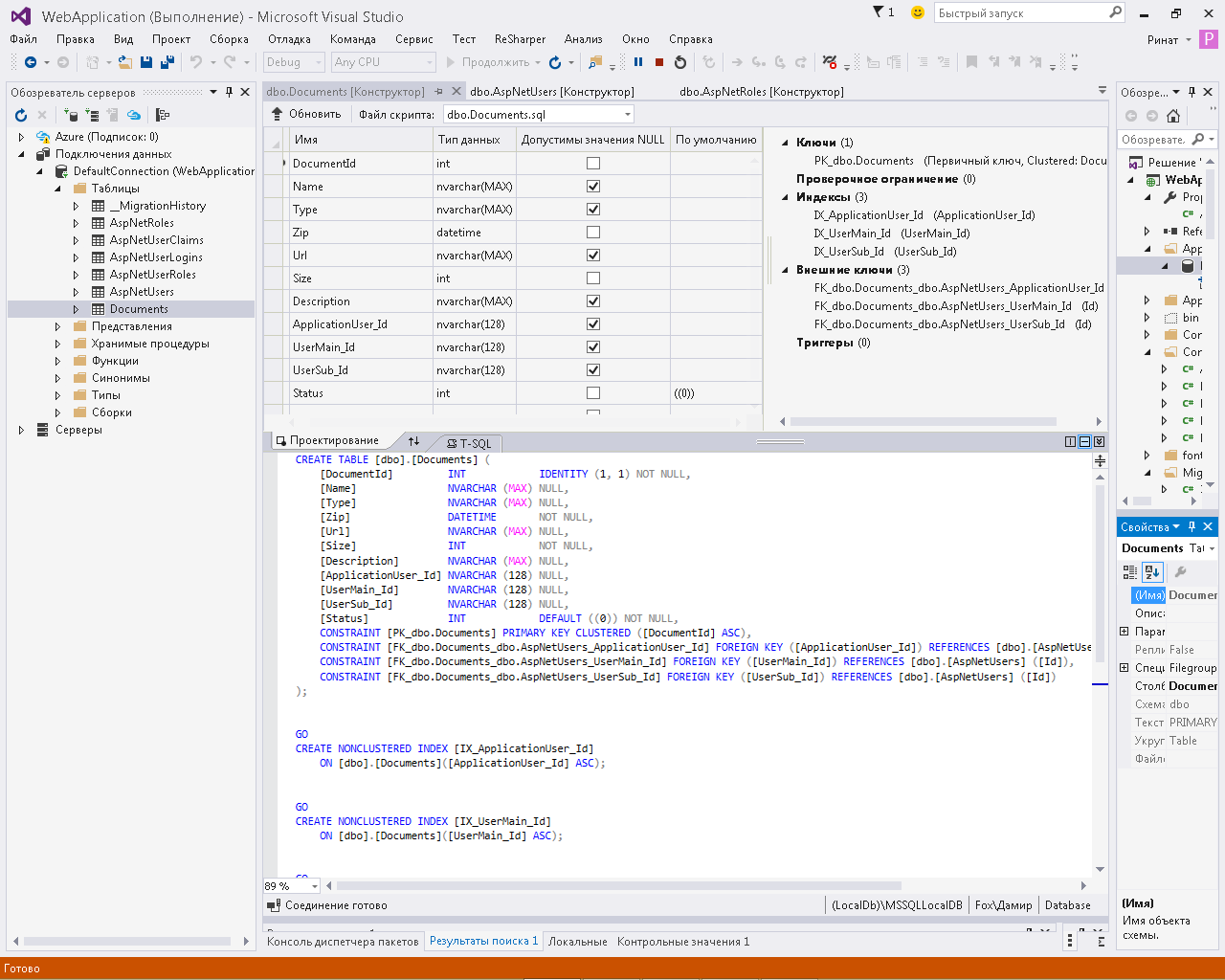


Рис. 2.10 – Сущность «Документ»

**2.3.4. Реестр пользователей**

Так же в приложении представлен раздел под названием «Реестр пользователей», в котором в виде таблицы так же отображаются пользователи, зарегистрированные в системе. Т.к. это строго конфиденциальный раздел, то там отображаются только два поля:

1. Имя пользователя
2. Почтовый адрес пользователя – который можно использовать для связи с ним в случае его отсутствия.

Пользователей нельзя удалить, редактировать или изменять обычным рядом пользователям. Их редактирование возможно только администратору, имеющему доступ к файлу базы данных «Database.mdf», расположенному в каталоге «WebApplication/App\_Data»

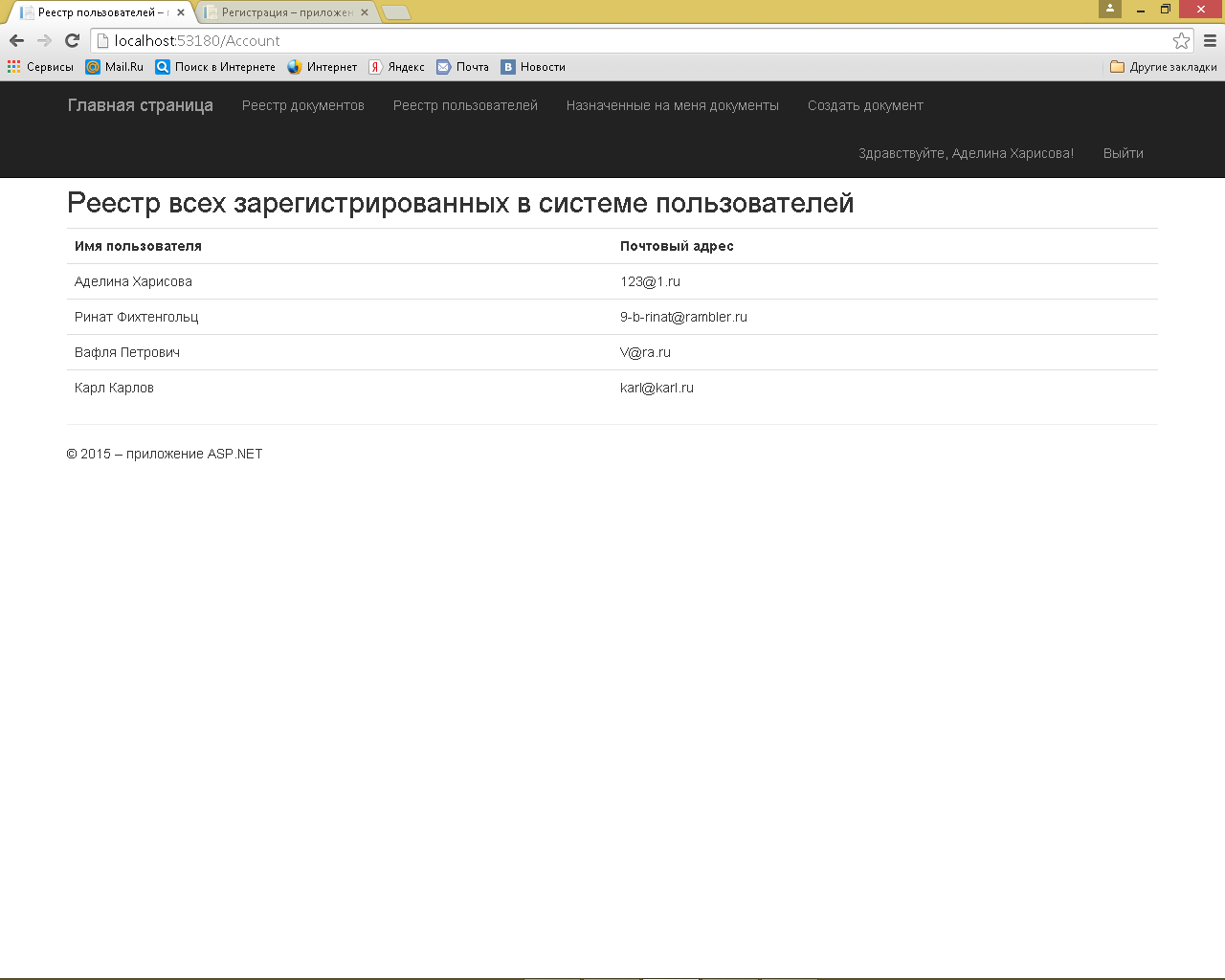


Рис. 2.11 – Реестр пользователей

Для каждого пользователя существует несколько определяющих его таблиц:

* AspNetRoles
* AspNetUserClaims
* AspNetUserLogins
* AspNetUserRoles
* AspNetUsers

Очень важно, что при создании документа и назначении пользователя – нельзя переназначать друго человека: принцип защищенности назначенных документов. Хотя, в ином случае, всё может зависеть от пожелания клиента-фирмы, под нужды которой может использоваться данное приложение

* + 1. **Реестр назначенных документов**
* У каждого пользователя так или иначе в скором времени появляются назначенные на него документы. Это могут быть как просто записи о работе, так и какие-либо «тяжелые» файлы наподобие docx, avi, pdf и т.п. И все эти документы можно просмотреть, зайдя во вкладку «Реестр назначенных документов»  
   Каждый документ имеет такое же отображение, как и в общем реестре документов:  
  Имя файла – содержит название файла
* Тип файла – типы могут быть самые разные(например doc, avi и т.п.)
* Дедлайн – срок, к которому задача должна быть исполнена
* Установщик – пользователь, который загрузил задание
* Исполнитель – пользователь, на которого назначено задание. Этот пользователь должен выполнить задание и отправить отчет установившему задание.
* Состояние – показывается текущее состояние файла(«Не принят на выполнение», «Выполняется», «Выполнен»)
* Ссылка на скачивание – позволяет скачать файл с сервера для просмотра на локальном компьютере
* Размер файла в байтах – показывает размер загруженного файла

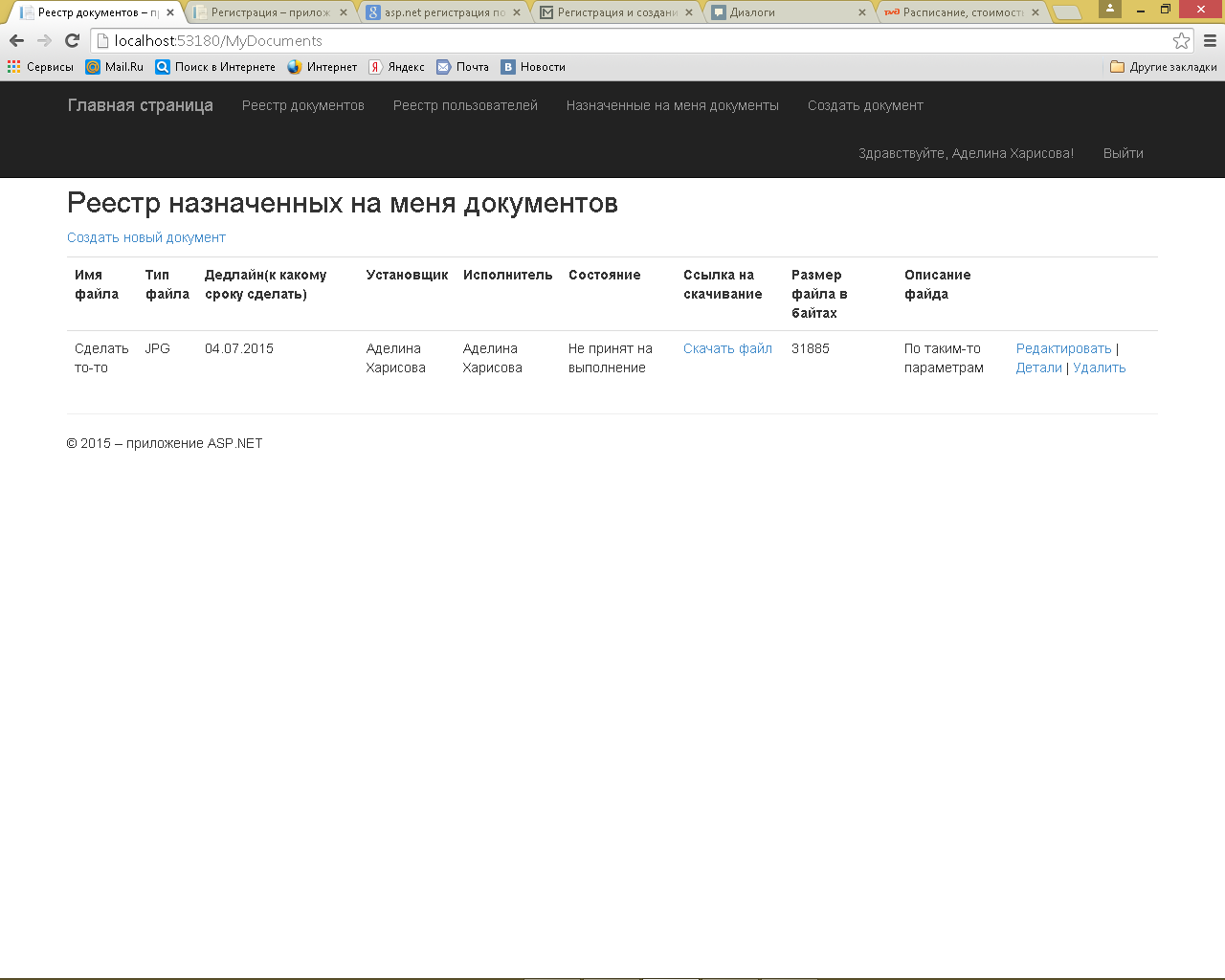


Рис. 2.12 – Назначенные на пользователя документы

Через данный раздел так же можно создавать и редактировать документы.

# Основные этапы создания документов

* + 1. **Создание новой записи о файле**

Любой зарегистрированный в системе пользователь может создать, изменить(редактировать) и удалить запись о документе. Но хранение документов на сервере реализовано так, что удаляется только запись о файле, но сам файл всегда остаётся на сервере и доступен администрации.

Рис. 2.13 – представлено стандартное окно загрузки файла

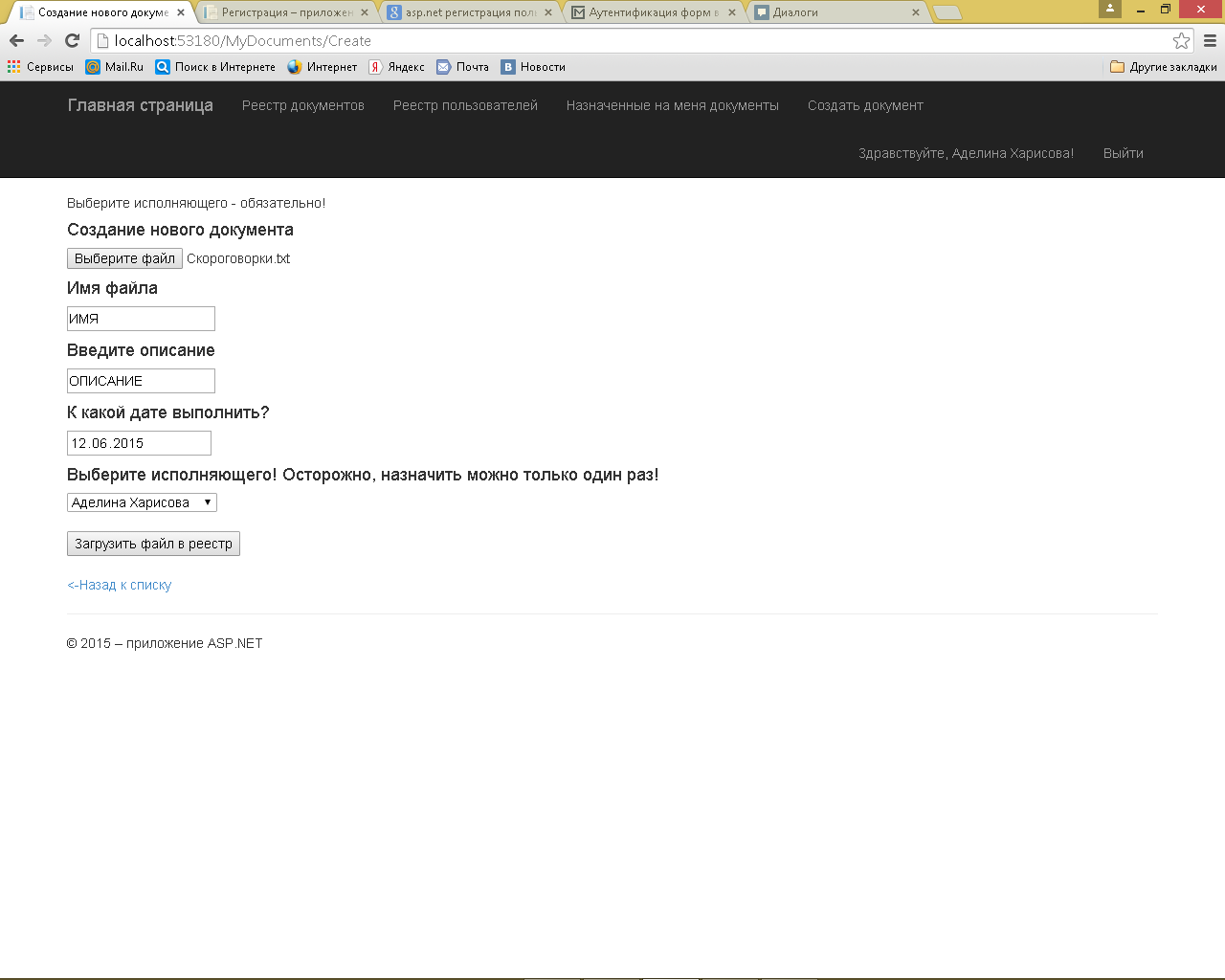
.

Рис. 2.13 – Создание записи о файле

В окне пользователю предлагается ввести данные о файле: его имя(название), описание, дату, к которой его нужно выполнить, а также – назначение исполнителя задачи. Исполнителю и назначается файл, что он может потом просмотреть в разделе «Назначенные на меня документы».

После нажатия кнопки «Загрузить файл в реестр» - он загружается на сервер и в ответ выдаётся сообщение, представленное на рисунке 2.14

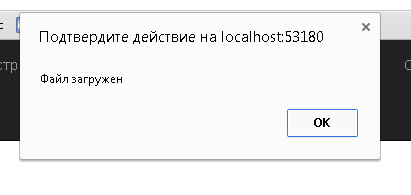


Рис. 2.14 – Подтверждение загрузки

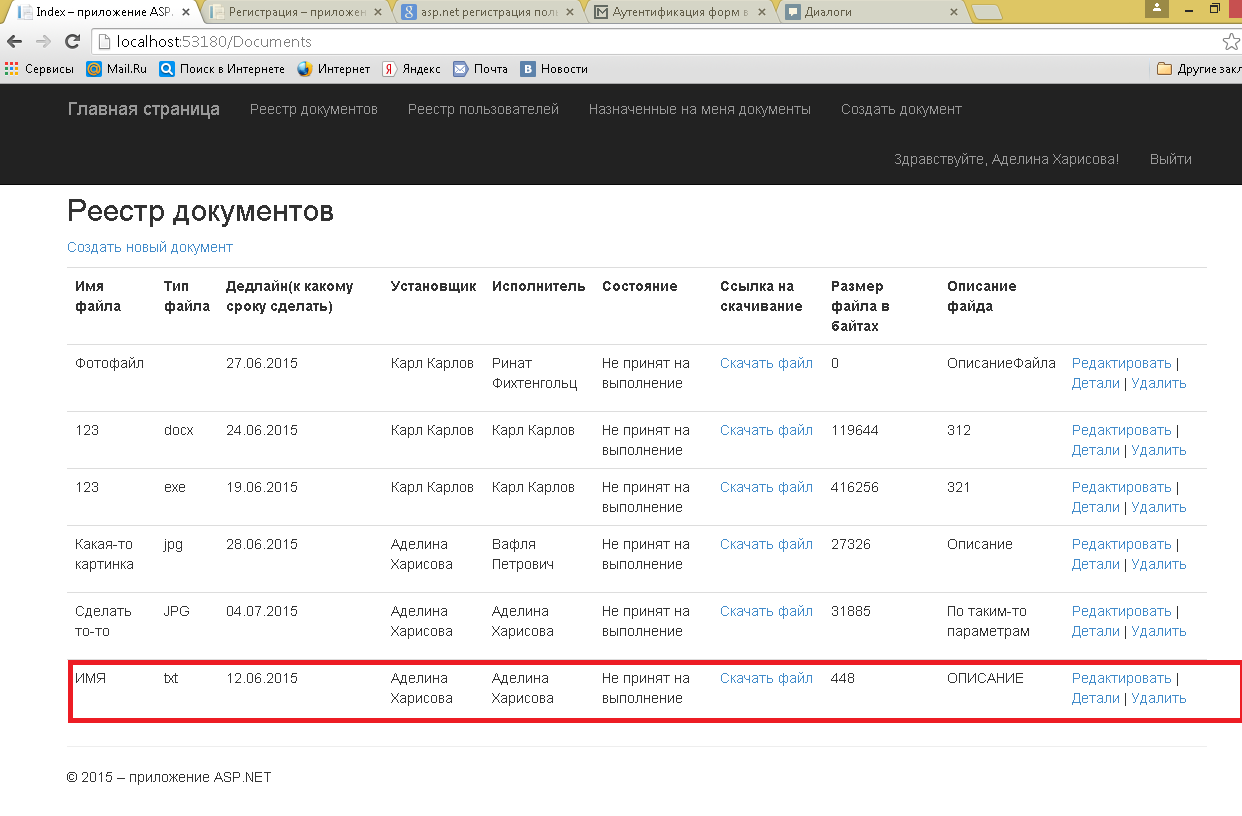
И файл появляется в общем реестр документов:  


Рис. 2.15 – Новый файл

* + 1. **Добавление, перестановка, удаление свойств загруженного документа**

У каждого загруженного документа-файла появляются справа поля:

* «Редактировать» - позволяет редактировать файл
* «Детали» - просмотр деталей о файле
* «Удалить» - удаление загруженного документа(удаление записи о файле из базы данных).

Таким образом, с файлом реализованы все операции CRUD – create/read/update/delete. Рассмотрим же их.

Раздел «Редактировать» содержит поля документа, которые можно изменять:

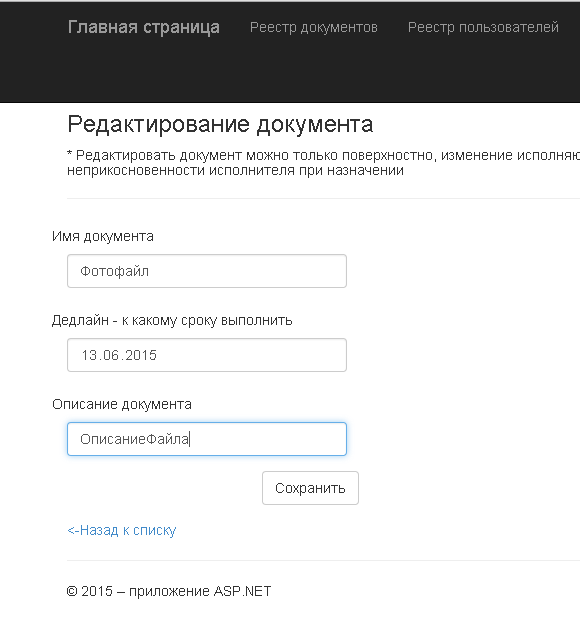


Рис. 2.16 – Редактирование через общий раздел

При нажатии на кнопку «Сохранить» - изменения внесутся в базу данных. Особенности в том, что в общем разделе при нажатии сохранить – можно изменить дедлайн(срок выполнения), но нельзя изменить статус файла. Статус файла может изменить только тот, кто должен его выполнить, а это возможно только через раздел «Назначенные на меня документы», в котором дедлайн менять уже нельзя соответственно(см. рис. 2.17). Это основа бизнес-логики данной системы, но она может быть изменена.

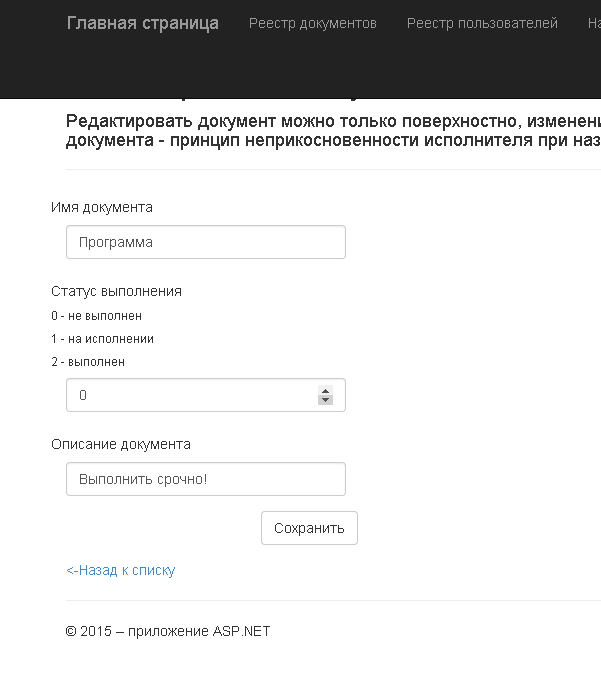


Рис. 2.17 – Редактирование через частный раздел

Раздел «Детали» содержит информацию о файле, которая может не войти в общую таблицу(свойства модели «Документ» всегда можно расширить, а его представление – ужать, и тогда роль раздела «Детали» станет большей)

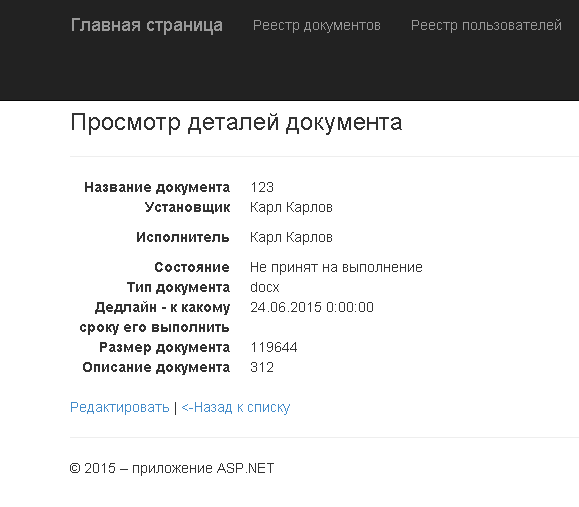


Рис. 2.17

* + 1. **Открытие документов**

Для того, чтобы открыть загруженный ранее на сервер документ – нужно нажать кнопку «Скачать» в любом из реестров документа, при этом произойдет обращение к серверу и запрос вернёт в результате запрошенный файл(как можно было увидеть выше, адрес для скачивания хранится в свойстве Url к записи документа, при этом название документа генерируется как его название при загрузке плюс текущее дата-время, дабы не было такого, что в один момент загрузили пару одинаковых файлов, один из которых может затереть другой).

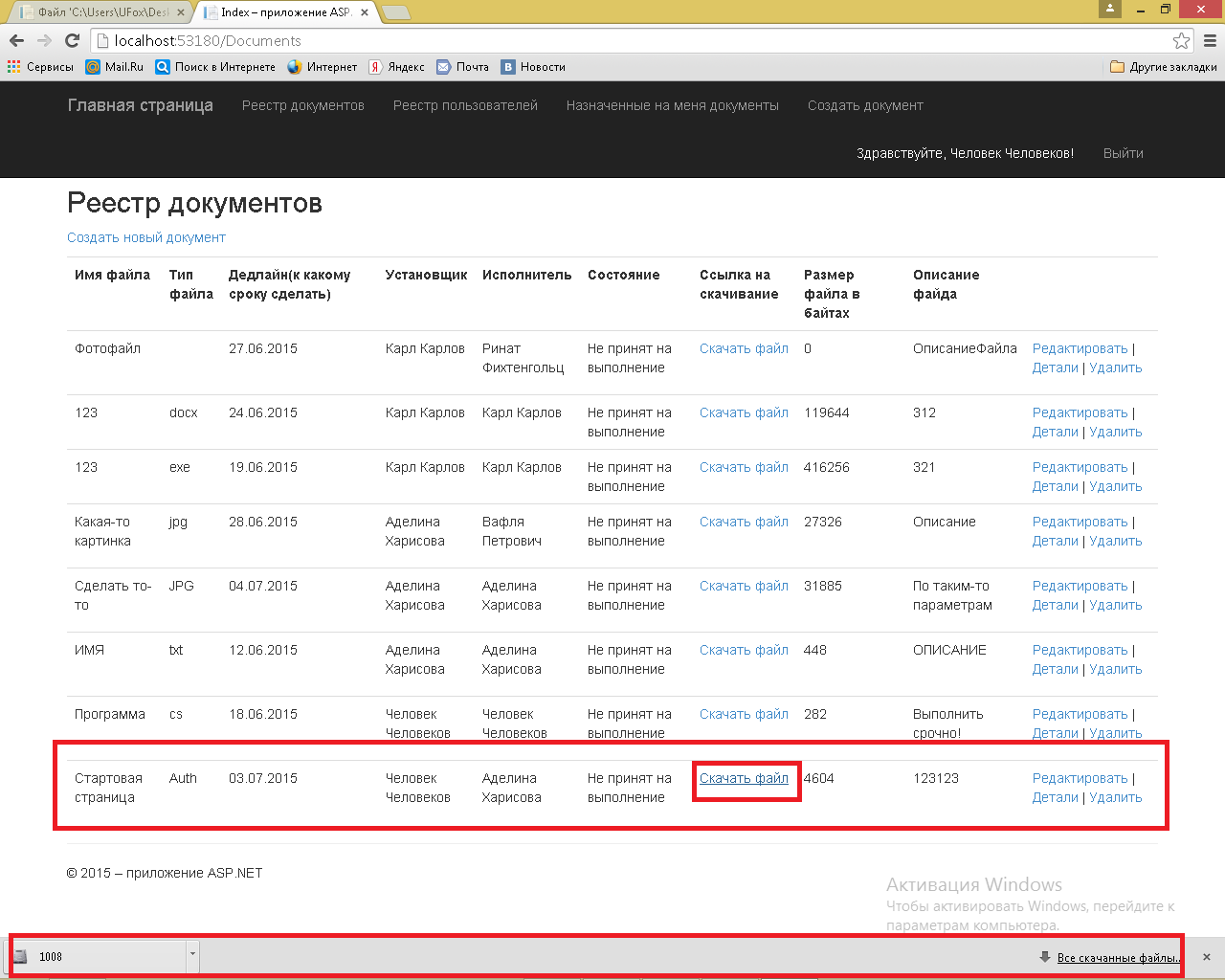


Рис 2.18.

**ГЛАВА 3. АНАЛИЗ ПРОГРАММНОГО КОДА**

# Примеры функций из контроллера DocumentsController.cs

Рассмотрим примеры кода по обработке поступающей информации с клиентской части на сервер.

Например, функция

public ActionResult Index()

{

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

}

# - возвращает как результат все документы, сохраненные базе данных.

Другая функция

// Получение главного пользователя

public ActionResult GetMainUser(int id)

{

var doc = db.Documents.Where(x => x.DocumentId == id).FirstOrDefault().UserMain.GetUser;

return Json(doc);

}

-возращает текст в виде полного имени назначившего документ пользователя.

Следующая функция так же служит как промежуточное звено между вызовом к базе данных для получения информации о текущем документе по его индивидуальному идентификационному номеру:

public ActionResult Details(int? id)

{

if (id == null)

{

return new HttpStatusCodeResult(HttpStatusCode.BadRequest);

}

Document document = db.Documents.Find(id);

if (document == null)

{

return HttpNotFound();

}

return View(document);

}

# Для загрузки же файла пользователя была написана функция, которая в качестве POST-запроса получает данные о пользователе, о файле с тех полей, которые заполнил пользователь. В ответ же на клиентскую часть уходит либо сообщение об успешной загрузке, либо сообщение о том, что загрузить файл не удалось:

[HttpPost]

public JsonResult Upload()

{

var files = Request.Files;

// Тексты - элементы документа:

var texts = Request.Form.AllKeys;

//MembershipUser currentUser = Membership.GetUser();

//Guid currentUserId = (Guid)currentUser.ProviderUserKey;

//var v= WebSecurity.GetUserId(User.Identity.Name);

var id = User.Identity.GetUserId();

var name = texts[0];

var descr = texts[1];

var zip = texts[2];

var id\_sub = texts[3];

var userSub = db.Users.Where(x => x.Id == id\_sub.ToString()).FirstOrDefault();//.FirstOrDefoault();

var userMain = db.Users.Where(x => x.Id == id.ToString()).FirstOrDefault();

foreach (string file in files)

{

var upload = Request.Files[file];

if (upload != null)

{

string fileName = System.IO.Path.GetFileName(upload.FileName);

db.Documents.Add(new Document

{

Name = name,//texts[0],

Description = descr,//texts[1],

Type = fileName.Split('.')[1],

Status = 0,

Zip = DateTime.Parse(zip),//texts[2]),

Url = Server.MapPath("~/ServerFiles/~" + DateTime.Now.ToLongDateString().Trim(new Char[] { ':', '.' }).Trim()+fileName.Trim()),

UserSub = userSub,//texts[3].ToString()).FirstOrDefault(),

UserMain = userMain,

Size = upload.ContentLength

});

db.SaveChanges();

upload.SaveAs(Server.MapPath("~/ServerFiles/~" + fileName));

}

}

//return View("Index");

return Json("Файл загружен");

}

# Примеры функций из представления Create.cshtml

Данное представление предназначено для создания документа. Для его отображения используется стандартная html-разметка:

@model WebApplication.Models.Document

@{

ViewBag.Title = "Создание нового документа";

}

</br></br></br>

<h5>Выберите исполняющего - обязательно!</h5>

<h4>Создание нового документа</h4>

<div>

<input type="file" name="upload" id="uploadFile" />

<h4>Имя файла</h4>

<input type="text" name="name" id="name" /> <br />

<h4>Введите описание</h4>

<input type="text" name="descr" id="description"/> <br/>

<h4>К какой дате выполнить?</h4>

<input type="date" name="date" id="dateFile" /> <br />

<h4>Выберите исполняющего! Осторожно, назначить можно только один раз!</h4>

@Html.DropDownList("ApplicationUser", ViewBag.ApplicationUser as SelectList,new { id ="userId"}) <br /><br />

<button id="submit">Загрузить файл в реестр</button>

<br />

</div>

<div>

<br />

@Html.ActionLink("<-Назад к списку", "Index")

</div>

В первой строке объявляется модель Документы, которая по запросу к одноименному контроллеру(DocumentsController.cs) запрашивает одноименную функцию(по имени Create), которая получает данные с формы и организует сохранение. Но в рамках данной задачи была написана своя функция Upload, т.к. логика сохранения была собственное.  
 При нажатии на кнопку «Загрузить файл в реестр» происходит срабатывание обработчика, написанного на языке JavaScript. Код его расположен в том же файле:

@section Scripts {

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

<script type="text/javascript">

$('#submit').on('click', function (e){

e.preventDefault();

var files = document.getElementById('uploadFile').files;

var name = document.getElementById('name').value;

var date = document.getElementById('dateFile').value;

var userId = document.getElementById('userId').value;

var descr = document.getElementById('description').value;

if (files.length > 0) {

if (window.FormData !== undefined) {

var data = new FormData();

var x = 0;

for (; x<files.length; x++){

data.append("file"+x,files[x]);

}

data.append(name, x++);

data.append(descr, x++);

data.append(date, x++);

data.append(userId, x++);

$.ajax({

type:"POST",

url: '@Url.Action("Upload", "Documents")',

contentType: false,

processData:false,

data: data,

success: function(result){

alert(result);

},

error: function(xhr, status, p3){

alert(xhr.responseText);

}

});

}

else{

alert("Случилась какая-то фигня и загрузить файл не вышло. Сорян -\_\_-")

}

}

})

</script>

}

В начале скрипт запрашивает все значения, введенные пользователем в окна и компонует их вместе с файлом в объект FormData. Затем, происходит отправка ajax-запроса на сервер в контроллер DocumentsController с фукнцией Upload, где, как был выше разобран код функции, и происходит загрузка всей информации. Как видно, концепция MVC позволяет разделить данные, представление и обработку действий пользователя на три отдельных компонента:

* Модель (англ. Model). Модель предоставляет знания: данные и методы работы с этими данными, реагирует на запросы, изменяя своё состояние. Не содержит информации, как эти знания можно визуализировать.
* Представление, вид (англ. View). Отвечает за отображение информации (визуализацию). Часто в качестве представления выступает форма (окно) с графическими элементами.
* Контроллер (англ. Controller). Обеспечивает связь между пользователем и системой: контролирует ввод данных пользователем и использует модель и представление для реализации необходимой реакции.

Важно отметить, что как представление, так и контроллер зависят от модели. Однако модель не зависит ни от представления, ни от контроллера. Тем самым достигается назначение такого разделения: оно позволяет строить модель независимо от визуального представления, а также создавать несколько различных представлений для одной модели.

**ЗАКЛЮЧЕНИЕ**

Куда стремится современный мир и современное бизнес-общество? Прежде всего, к автоматизации своих процессов управления, а прежде всего - сокращению времени контроля исполнения разнообразных взаимодействий внутри свой структуры. То есть - к построению модели управления/контроля с самым высоким КПД(коэффициентом полезного действия). Чем более мобильны структуры внутри компании, чем более чётко организовано их взаимодействие - тем выше сокрость их работы. И именно для этого предназначены системы электронного документооборота, выбор которых сейчас на рынке всё больше растёт, но растёт и стоимость их использования. Моё же приложение всегда может быть растиражировано на свободной основе(Open Source) для доработки и использования в коммерческих целях малыми и средними предприятиями.

Разработанный в среде Visual Studio 2015 сводный контролирующий реестр исполняемых документов(или СКРИД, в сокращении), может быть всегда изменён под требования заказчика. Ведь особенностью работы в ASP.NET/MVC является модульность платформы: всегда можно добавить новые сущности и элементы управления к ним, которые так или иначе будут расширять функционал приложения. Как следствие, приложение можно "дорастить" до полноценной CRM-системы по управлению документооборотом на каком-нибудь промышленном предприятии. Функционал может быть полностью изменен: от обычного файлообменника до уровня системы контроля версий(в среде программистов извсетны такие системы контроля версий, как GIT, Team Foundation Server, Mercurial и другие).

Тематика или же направление развития данного приложения является очень актуальным, так как в современном обществе всё стремится к автоматизации, сокращению времени обращений между типами "человек-человек", то есть к реализации системы "человека-машина", а в идеале: "машина-машина", когда компьютеры самые автоматизируют все процессы обмена информацией между собой. И не только процесса обмена, но и процессы создания, редактирования информации! Как следствие, это увеличивает товарооборот предприятия, ведёт к росту его прибыли и повышению коэффициента полезного действия между его структурами. Подобные предложения по разработке приложений представляют такие компании, как

1. Барс Груп, о которой говорилось в начале работы

2. "1С: Предприятие"

3. "Univers-Soft" - молодая компания, являющаяся резидентом Казанского IT-Парка и реализующая системы документооборота так же по все России(CRM-системы управления)

Разработанное приложение может быть использовано как в коммерческих целях для автоматизации контроля документооборота на каком-либо предприятии, либо как тестовая учебная модель контроля знаний студентов в определенной группе(назначение им задач и контроль исполнения)

Благодаря этому решению организация, независимо от ее размеров и зрелости, получает мощный инструмент, цель применения которого - организация электронного документооборота. Вместе с ростом компании органично будет расти и ее информационное пространство.

Вот некоторые возможности этого решения:

* Документооборот вашей организации унифицирован. Документы хранятся в электронном виде в одном месте - в облаке.
* Все документы доступны из любой точки и максимально защищены от несанкционированного доступа.
* Все приложения для управления документами работают на любых устройствах. Доступ выполняется по одной учетной записи.
* Согласование и утверждение документа в системе выполняется одновременно всеми ответственными в электронной форме. Всегда известен статус документа.
* Процессы внутреннего документооборота (согласование приказов, оформление командировок), полностью автоматизированы.
* На любое устройство можно получать уведомление об изменении документа.
* Задачи по обработке электронных документов с помощью системы можно назначать сотрудникам с контролем сроков их исполнения системой.
* С помощью корпоративного портала можно обмениваться информацией с коллегами, партнерами и клиентами.

Внедрение системы электронного документооборота с использованием облачных технологий представляет качественно новый уровень оптимизации затрат вашей компании на эту сферу деятельности:

**Экономия капиталовложений.** Для разворачивания системы управления документооборотом на базе собственного ЦОД необходимы: закупка дорогостоящего оборудования и программного обеспечения, подготовка площадей и привлечение высокооплачиваемых специалистов. Заказав облачные сервисы и разместив на них этот проект, можно получить полномасштабный электронный документооборот предприятия за небольшую абонентскую плату или же условно бесплатно с дальнейшей поддержкой.

**Масштабируемость и минимизация рисков.** При расширении штата вашего предприятия придется докупать оборудование и лицензии на программное обеспечение системы электронного документооборота. При сокращении его - решать проблему реализации лишнего оборудования. Облачные сервисы позволяют в любое время наращивать или уменьшать количество абонентов. При желании можно полностью отказаться от услуги без всяких потерь.

**Минимизация стоимости владения.** Для обеспечения надежного функционирования собственной системы управления документооборотом необходимы высоконадежные серверы, каналы передачи данных и источники питания с обязательным резервированием. А также постоянный мониторинг системы, ремонт и замена вышедшего из строя оборудования. А данное приложение можно разместить за довольно небольшую плату для предприятия на каком-нибудь высокопропускном, удаленном и арендуемом сервере.

**СПИСОК ЛИТЕРАТУРЫ**

1. Ник Рендольф, Дэвид Гарднер, Майкл Минутилло, КРис. Андерсон. Visual Studio 2010 для профессионалов = Professional Visual Studio 2010. — М.: «Диалектика», 2011. — С. 1184. — ISBN 978-5-8459-1683-9.
2. Алекс Макки. Введение в .NET 4.0 и Visual Studio 2010 для профессионалов = Introducing .NET 4.0: with Visual Studio 2010. — М.: «Вильямс», 2010. — С. 416. — ISBN 978-5-8459-1639-6.
3. Карли Уотсон, Кристиан Нейгел, Якоб Хаммер Педерсен, и др. Visual C# 2008: базовый курс. Visual Studio® 2008 = Beginning Visual C# 2008. — М.: «Диалектика», 2009. — С. 1216. — ISBN 978-5-8459-1532-0.
4. Пауэрс Л., Снелл М. Microsoft Visual Studio 2008 = Microsoft Visual Studio 2008 Unleashed by Lars Powers and Mike Snell. — C.: «БХВ-Петербург», 2008. — С. 1200. — ISBN 978-5-9775-0378-5.
5. Майо Д. Самоучитель Microsoft Visual Studio 2010 = Microsoft Visual Studio 2010: A Beginner's Guide (A Beginners Guide). — C.: «БХВ-Петербург», 2010. — С. 464. — ISBN 978-5-9775-0609-0.
6. Адам Фримен. ASP.NET MVC 5 с примерами на C# 5.0 для профессионалов, 5-е издание = Pro ASP.NET MVC 5. — М.: «Вильямс», 2014. — 736 с. — ISBN 978-5-8459-1911-3.
7. Джесс Чедвик, Тодд Снайдер, Хришикеш Панда. ASP.NET MVC 4: разработка реальных веб-приложений с помощью ASP.NET MVC = Programming ASP.NET MVC 4: Developing Real-World Web Applications with ASP.NET MVC. — М.: «Вильямс», 2013. — 432 с. — ISBN 978-5-8459-1841-3.
8. Магдануров Г., Юнев В. ASP.NET MVC Framework. — C.: «БХВ-Петербург», 2010. — С. 320. — ISBN 978-5-9775-0462-1.