МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования

«Гродненский государственный университет имени Янки Купалы»

Факультет математики и информатики

Кафедра программного обеспечения интеллектуальных и компьютерных систем

САСИМОВИЧ АНДРЕЙ СТАНИСЛАВОВИЧ

СЕМЕРИКОВ ВИКТОР ВЛАДИМИРОВИЧ

**Разработка клиентской части Интернет-ресурса для сбора, обмена и агрегации медиа данных**

Курсовая работа

студентов 4 курса специальности

1-40 01 01 «Программное обеспечение информационных технологий»

дневной формы обучения

|  |  |
| --- | --- |
| Допущен к защите  Заведующий кафедрой  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  Рудикова Л. В.  «\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2014 г. | **Научный руководитель**  Рудикова Лада Владимировна,  зав. кафедрой программного обеспечения интеллектуальных  и компьютерных систем,  канд. физ.-мат. наук, доцент |
|  |  |

Гродно, 2014

Оглавление

[Введение 2](#_Toc406411959)

[Цели и задачи курсового проекта 2](#_Toc406411960)

[Глава 1. Выбор программных средств для реализации приложения «сеть сто» 3](#_Toc406411961)

[1.1. Microsoft SQL Server 2012 3](#_Toc406411962)

[1.2. Среда разработки Visual Studio 2013 4](#_Toc406411963)

[1.3. Платформа .NET Framework 4.5 5](#_Toc406411964)

[1.4. Язык программирования C# 6](#_Toc406411965)

[1.5. Entity Framework 6.0 7](#_Toc406411966)

[1.6. Cреда программирования ASP.NET MVC 8](#_Toc406411967)

[Глава 2. Анализ предметной области и создание концептуальной модели 11](#_Toc406411968)

[2.1. Сбор информации о предметной области 11](#_Toc406411969)

[2.2. Выделение сущностей 11](#_Toc406411970)

[2.3. Создание моделей 14](#_Toc406411972)

[Глава 3. Программная реализация приложения «Галерея» 19](#_Toc406411973)

[3.1. Общие сведения о структуре приложения. 19](#_Toc406411974)

[3.2 Функционал клиента. 19](#_Toc406411975)

[3.3 Функционал мастера 20](#_Toc406411976)

[3.4. Функционал администратора 21](#_Toc406411977)

[Заключение 22](#_Toc406411978)

[Список использованных источников 23](#_Toc406411979)

# 

# Введение

Проект «Книжный магазин» призван в первую очередь решать учебные задачи разработки базы данных. Приложение даёт возможность управлять базой данных книг и авторов, а также предоставляет интерфейс пользователя для заказа и скачивания электронной литературы. Отличительной особенностью проекта, выделяющей его среди других подобных проектов, является интуитивность и простота интерфейса, построенного по принципу «ничего лишнего».

# Цели и задачи курсового проекта

Цель работы – создание интернет-комплекса для управления базой данных книг и авторов, а также предоставляющего интерфейс для заказа и скачивания электронной литературы.

Предусмотрено решение следующих задач:

1. Обзор материалов, касающихся систем и технологий для их реализации.

2. Моделирование предметной области и обоснование выбранных средств разработки проекта

3. Программная реализация проекта.

4. Подготовка презентации и отчёта по курсовой работе.

# Глава 1. Выбор программных средств для реализации приложения «сеть сто»

## 1.1. Microsoft SQL Server 2012

1. Реляционная база данных - это совокупность отношений, содержащих всю информацию, которая должна храниться в БД. Пользователи воспринимают такую базу данных как совокупность таблиц, связанных между собой.
2. Основными задачами системы SQL Server являются: организация одновременного доступа к данным большого количества пользователей; манипуляция информацией, хранимой в БД и др.
3. SQL Server поддерживает реляционную модель данных и выполняет функции создания объектов БД (таблиц, индексов, представлений и т.д.), осуществляет проверку целостности БД и отвечает за безопасность данных в системе.
4. Задачи, связанные с администрированием БД системы SQL Server, обращение к системе удобно выполнять непосредственно с компьютера-сервера. В SQL Server для манипулирования данными используется язык Transact SQL, переработанной компанией Microsoft версией языка SQL.
5. Система SQL Server сохраняет создаваемые объекты в соответствующих файлах на дисках компьютера сервера. При этом для БД создаются специальные таблицы, в которых хранится информация о различных элементах базы данных: индексах, таблицах, пользователях и т.д.
6. Файлы БД сохраняются с расширением MDF, а системные файлы с расширением LDF.

## 1.2. Среда разработки Visual Studio 2013

Visual Studio — это набор инструментов разработки, основанных на использовании компонентов, и других технологий для создания мощных, производительных приложений. Кроме того, среда Visual Studio оптимизирована для совместного проектирования, разработки и развертывания корпоративных решений.

Среда разработки Visual Studio представляет собой полный набор средств разработки для создания веб-приложений ASP.NET, XML (веб-службы), настольных приложений и мобильных приложений. Visual Basic, Visual C# и Visual C++ используют единую интегрированную среду разработки (IDE), которая позволяет совместно использовать средства и упрощает создание решений на базе нескольких языков. Кроме того, в этих языках используются функциональные возможности платформы .NET Framework, которая позволяет получить доступ к ключевым технологиям, упрощающим разработку веб-приложений ASP и XML (веб-службы).

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

## 1.3. Платформа .NET Framework 4.5

Платформа .NET Framework — это интегрированный компонент Windows, который поддерживает создание и выполнение нового поколения приложений и веб-служб XML. При разработке платформы .NET Framework учитывались следующие цели.

* Обеспечение согласованной объектно-ориентированной среды программирования для локального сохранения и выполнения объектного кода, для локального выполнения кода, распределенного в Интернете, либо для удаленного выполнения.
* Обеспечение среды выполнения кода, минимизирующей конфликты при развертывании программного обеспечения и управлении версиями.
* Обеспечение среды выполнения кода, гарантирующей безопасное выполнение кода, включая код, созданный неизвестным или не полностью доверенным сторонним изготовителем.
* Обеспечение среды выполнения кода, исключающей проблемы с производительностью сред выполнения сценариев или интерпретируемого кода.
* Обеспечение единых принципов работы разработчиков для разных типов приложений, таких как приложения Windows и веб-приложения.
* Разработка взаимодействия на основе промышленных стандартов, которое обеспечит интеграцию кода платформы .NET Framework с любым другим кодом.

Двумя основными компонентами платформы .NET Framework являются общеязыковая среда выполнения (CLR) и библиотека классов .NET Framework. Основой платформы .NET Framework является среда CLR. Среду выполнения можно считать агентом, который управляет кодом во время выполнения и предоставляет основные службы, такие как управление памятью, управление потоками и удаленное взаимодействие. При этом накладываются условия строгой типизации и другие виды проверки точности кода, обеспечивающие безопасность и надежность. Фактически основной задачей среды выполнения является управление кодом. Код, который обращается к среде выполнения, называют управляемым кодом, а код, который не обращается к среде выполнения, называют неуправляемым кодом. Другой основной компонент платформы .NET Framework, библиотека классов, представляет полную объектно-ориентированную коллекцию типов, которые применяются для разработки приложений, начиная от обычных, запускаемых из командной строки или с графическим интерфейсом пользователя, и заканчивая приложениями, использующими последние технологические возможности ASP.NET, такие как Web Forms и веб-службы XML.

## 1.4. Язык программирования C#

C# предназначен для построения самых разнообразных приложений. Язык C# простой, мощный, типобезопасный и объектно–ориентированный. Благодаря множеству нововведений C# обеспечивает возможность быстрой разработки приложений, но при этом сохраняет выразительность и элегантность, присущую языкам С.

Как объектно–ориентированный язык, C# поддерживает понятия инкапсуляции, наследования и полиморфизма. Класс может наследовать непосредственно из одного родительского класса, но может реализовывать любое число интерфейсов. Язык C# упрощает разработку компонентов программного обеспечения благодаря нескольким инновационным конструкциям языка, в число которых входят следующие:

Инкапсулированные сигнатуры методов, называемые делегатами, которые поддерживают типобезопасные уведомления о событиях.

Свойства, выступающие в роли методов доступа для закрытых переменных членов.

Атрибуты с декларативными метаданными о типах во время выполнения.

Встроенные комментарии XML-документации.

Так же осуществлена реализация мощного асинхронного программирования с помощью **async/await конструкций**.

## 1.5. Entity Framework 6.0

Платформа Entity Framework представляет собой набор технологий, обеспечивающих разработку приложений, связанных с обработкой данных. В Entity Framework разработчики получают возможность работать с данными, представленными в форме относящихся к конкретным доменам объектов и свойств, таких как клиенты и их адреса, не будучи вынужденными обращаться к базовым таблицам и столбцам базы данных, где хранятся эти данные. Entity Framework дает разработчикам возможность работать с данными на более высоком уровне абстракции, создавать и сопровождать приложения, ориентированные на работу с данными, одновременно с этим сокращая объем кода по сравнению с традиционными приложениями.

Entity Framework — это объектно-реляционный модуль сопоставления, который позволяет уменьшить различия между объектно-ориентированным миром .NET Framework и миром реляционных баз данных. Прежде всего эта платформа позволяет разработчикам взаимодействовать с концептуальной моделью приложения с помощью хорошо знакомых приемов объектно-ориентированного программирования. Платформа Entity Framework позволяет работать с данными в форме специфических для домена объектов и свойств (например, с клиентами и их адресами) без необходимости обращаться к базовым таблицам и столбцам базы данных, где хранятся эти данные. Разработчики могут инициировать операции доступа к данным в концептуальной модели, а Entity Framework преобразует их в действия с реляционной базой данных.

В приложении Entity Framework есть два основных уровня:

* Уровень моделирования
* Уровень объектов

Уровень моделирования состоит из трех компонентов:

* Концептуальная модель, состоящая из типов сущностей домена и связей, которая основана на модели EDM
* Схема базы данных, в которой определены таблицы и связи
* Сопоставление между концептуальной моделью и схемой базы данных

В платформе Entity Framework для преобразования операций с объектами сущности, таких как создание, чтение, обновление и удаление, в соответствующие им операции в базе данных используется компонент сопоставления.

## 1.6. Cреда программирования ASP.NET MVC

ASP.NET является единой моделью для разработки веб-приложений с применением минимума кода, которая содержит службы, необходимые для построения веб-приложений для предприятий. ASP.NET является частью платформы .NET Framework, а потому обеспечивает доступ к классам этой платформы. Приложения могут быть написаны на любом языке среды CLR, включая Microsoft Visual Basic, C#, JScript .NET и J#. Эти языки позволяют разрабатывать приложения ASP.NET, которые могут использовать все преимущества среды CLR, типовой безопасности, наследования и т. д.

Шаблон архитектуры Model-View-Controller (MVC) разделяет приложение на три основных компонента: модель, представление и контроллер. Платформа ASP.NET MVC представляет собой альтернативу схеме веб-форм ASP.NET при создании веб-приложений. Платформа ASP.NET MVC является легковесной платформой отображения с широкими возможностями тестирования и, подобно приложениям на основе веб-форм, интегрирована с существующими функциями ASP.NET, например с главными страницами и проверкой подлинности на основе членства.

MVC представляет собой стандартный шаблон разработки, знакомый многим специалистам. Некоторые типы веб-приложений имеют преимущества при создании на платформе MVC. Для других может быть целесообразно использование традиционной схемы приложения ASP.NET, основанной на веб-формах и обратной передаче. В некоторых случаях возможно сочетание двух подходов: применение одной схемы не исключает использования другой.

В состав платформы MVC входят следующие компоненты.

* Модели. Объекты моделей являются частями приложения, реализующими логику для домена данных приложения. Объекты моделей часто получают и сохраняют состояние модели в базе данных.
* Представления. Представления служат для отображения пользовательского интерфейса приложения. Пользовательский интерфейс обычно создается на основе данных модели.
* Контроллеры. Контроллеры осуществляют взаимодействие с пользователем, работу с моделью, а также выбор представления, отображающего пользовательский интерфейс. В приложении MVC представления только отображают данные, а контроллер обрабатывает вводимые данные и отвечает на действия пользователя.

Шаблон MVC позволяет создавать приложения, различные аспекты которых (логика ввода, бизнес-логика и логика интерфейса) разделены, но достаточно тесно взаимодействуют друг с другом. Эта схема указывает расположение каждого вида логики в приложении. Пользовательский интерфейс располагается в представлении. Логика ввода располагается в контроллере. Бизнес-логика находится в модели. Это разделение позволяет работать со сложными структурами при создании приложения, так как обеспечивает одновременную реализацию только одного аспекта. Например, разработчик может сконцентрироваться на создании представления отдельно от бизнес-логики.

# Глава 2. Анализ предметной области и создание концептуальной модели

## 2.1. Сбор информации о предметной области

На сегодняшний день электронные книги приобретают всё большую популярность. Это связано как с ростом популярности и развитием глобальной сети Интернет, так и с широким выбором цифровых устройств, дающих возможность комфортно читать электронную литературу.

Создание приложения для поиска и скачивания электронной литературы предполагает разработку соответствующей структуры базы данных. Эта структура, должна обеспечивать с одной стороны возможность хранения всей необходимой информации о книгах и их авторах, с другой – удобный и быстрый доступ к хранимым данным.

База будет содержать информацию об книжной продукции, на каждую единицу продукции также будет содержаться информация об авторах, краткое описание, а также другая мета-информация для работы системы ( количество скачиваний данной книге , скидки , цена).

Все клиенты долдны зарегестрироваться в системе, будет существовать несколько типов клиентов (обычный пользователь, администратор).

Администратор будет управлять системой назначать новые цены или скидки на товары,редактировать данные о клиентах и книгах.

Пользователь имеет возможность найти книгу и скачать ее себе на компьютер для последующего чтения.

Поиск можно выполнить с помощью фильтров по названию , автору , isbn номеру .

Оформление заказа производится на отдельной странице.При оформлении заказа будет регистрироваться книга, стоимость которую оплатил клиент клиента , дата заказа.

Проект содержит информацию о книгах , авторах , пользователях , заказах , расценках .

С целью спроектировать базу данных, соответствующую этим требованиям, имеет смысл разработать концептуальную и физическую модели базы данных.

## 2.2. Выделение сущностей

**Сущности:**

1) Роли – сущность, которая представляет информацию о роли пользователя.

2) Персона - информация о конкретной персоне. Таблица от которое наследуются таблица Авторов и Пользователей.

3) Авторы - информация об авторах наследник от персоны.

4) Цены - цена книги вместе со скидкой на данную книгу.

5) Заказы - информация об уже сделанных заказах

6) Скидки - скидка которую сможет использовать пользователь при покупке книги.

7) Пользователи - информация о пользователе наследник от персоны. Имеет Роли , список Заказов , Скидки.

8) Книга - информация о книге . Описание , ISBN номер , адрес файла , Заказы, Цена, Авторы , также информация о следующем или предыдущем томе если есть.

## 2.3. Создание концептуальной и физической модели

Таким образом, построена физическая модель для данной предметной области, выполненная с использованием MS Visual Studio 2012, представлена на Рис. 2.1.

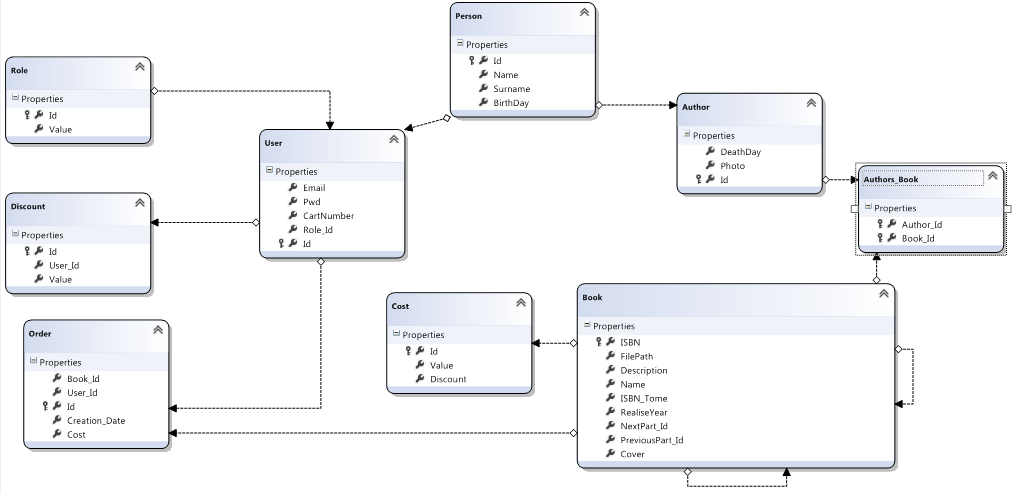
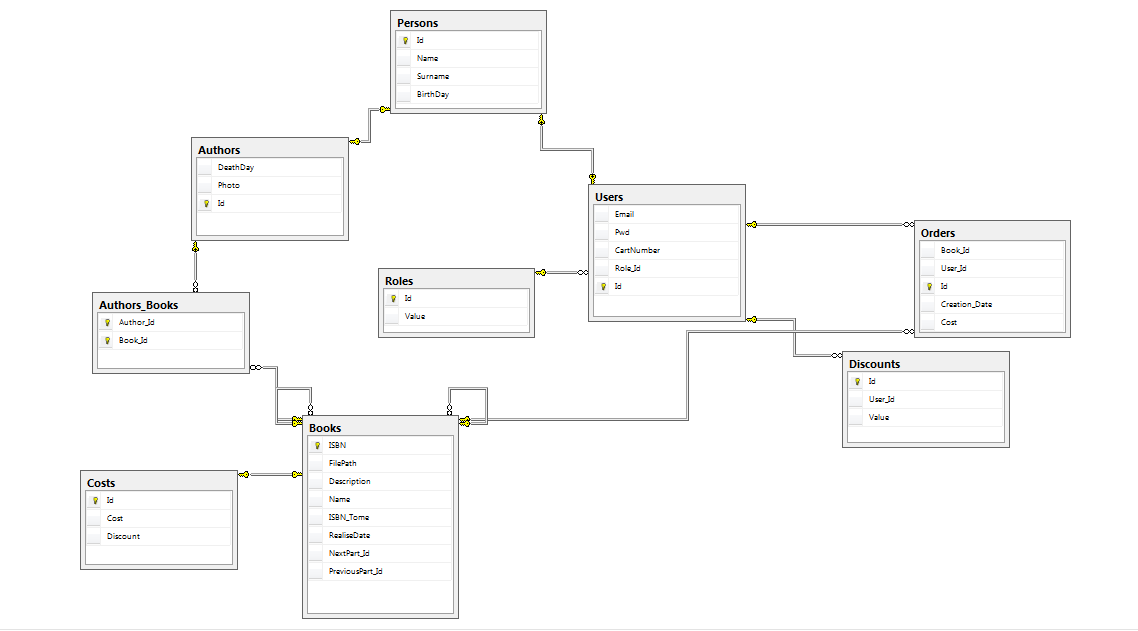
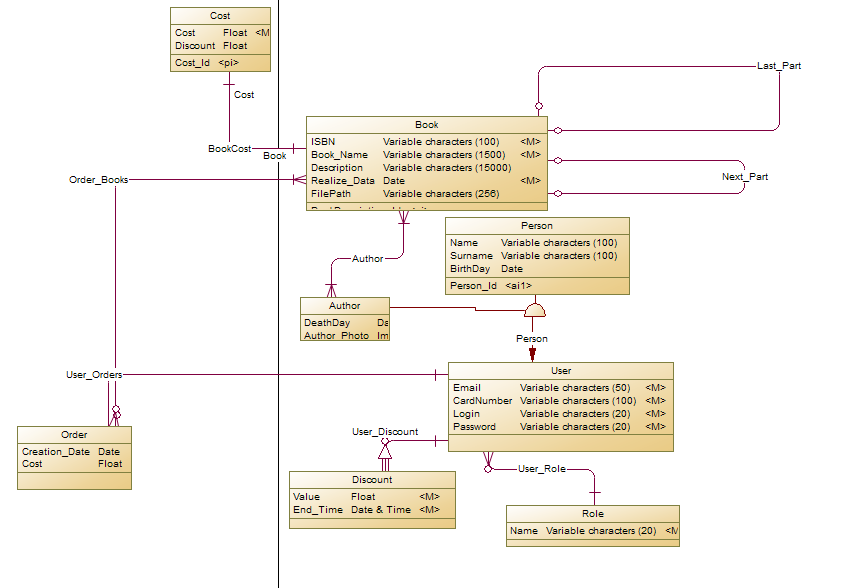


Рис 2.1 Физическая модель

Концептуальная модель для данной предметной области, выполненная с использованием PowerDesigner, представлена на Рис. 2.



*Рис 2-2.* *Физическая модель БД*



# Глава 3. Программная реализация приложения «КНИЖНЫЙ МАГАЗИН»

## 3.1. Общие сведения о структуре приложения.

Приложение книжный магазин представляет собой ASP.NET MVC приложение.

Все клиенты должны авторизованы в системе, в которой существует несколько типов клиентов (обычный пользователь, администратор).

Администратор будет управлять системой назначать новые цены ,  
книги , или скидки на товары,редактировать данные о клиентах и книгах.

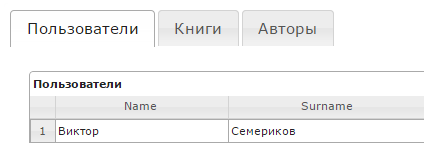
Пользователь имеет возможность найти книгу и скачать ее себе на компьютер для последующего чтения предварительно согласившись с ценой.

Оформление заказа производится на отдельной странице. При оформлении заказа будет регистрироваться книга, стоимость которую оплатил клиент, дата заказа.

Администратор имеет возможность добавления удаления книг и авторов.

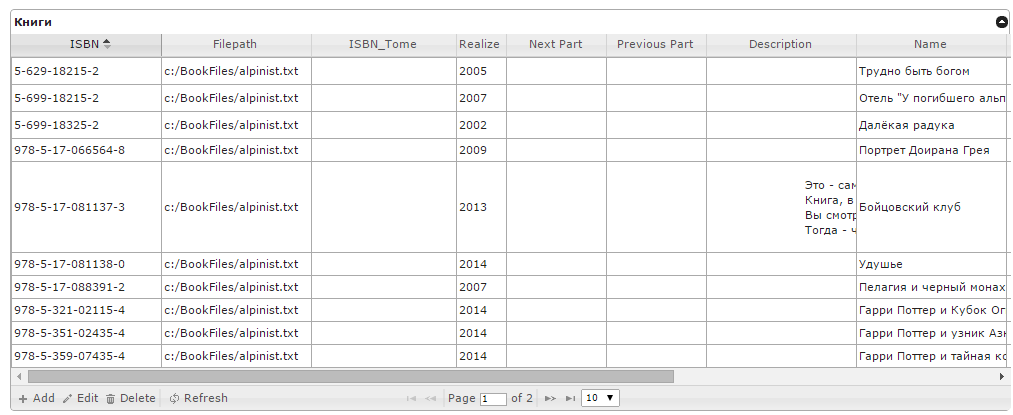
## 3.2 Интерфейс приложения

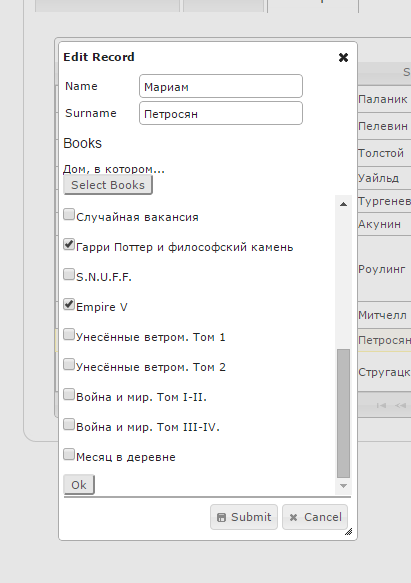
Приложение можно разделить на две большие части – администраторскую и клиентскую. Рассмотрим каждую из них.

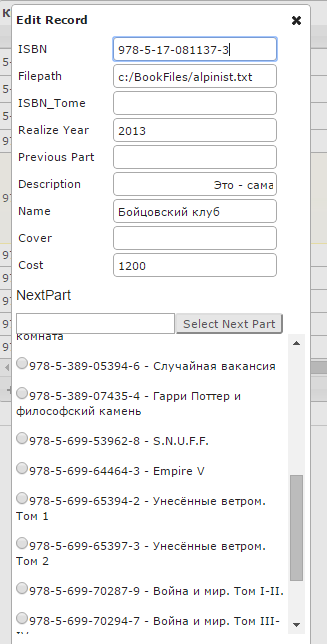
Администраторская часть приложения позволяет просматривать и редактировать информацию о книгах, авторах и пользователях системы. Интерфейс состоит из трёх вкладок – пользователи, книги и авторы.

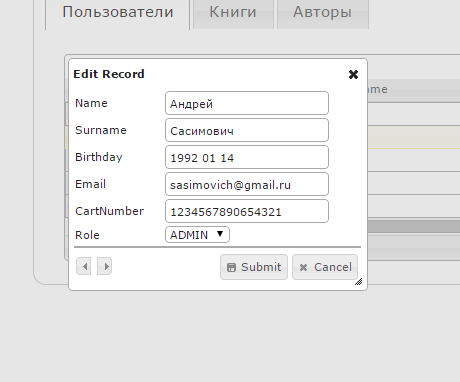
Вкладки администраторской части приложения

Каждая из вкладок содержит интерактивные таблицы, в которые можно добавлять новые записи, удалять и редактировать старые.

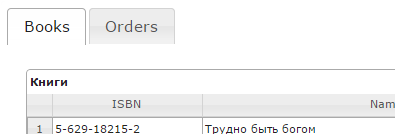
Таблица Книги

Редактирование информации о авторе

Редактирование информации о книге

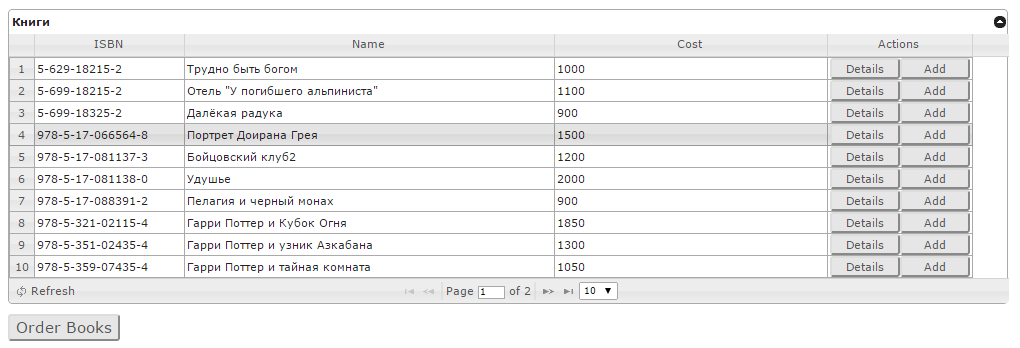
Интерфейс пользовательской части приложения также оформлен в виде страницы с несколькими вкладками – книги и заказы.

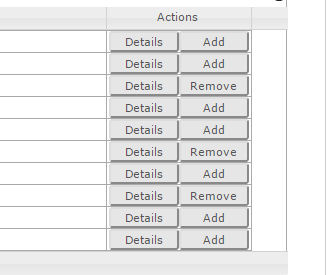
Редактирование информации о пользователе



Вкладки на главной странице пользователя

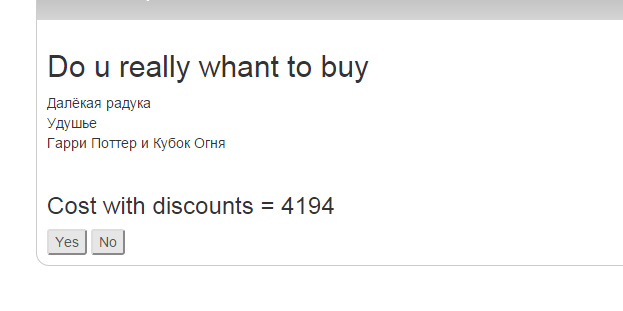
На вкладке заказы пользователь может отметить книги, которые хочет купить и подтвердить свой выбор нажав на кнопку «Order Book»



Таблица доступных книг

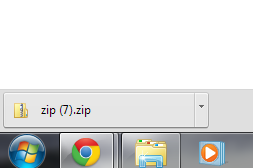
Кнопки добавления/удаления книги из заказа

После того как пользователь выбрал нужные книги и нажал кнопку «Order Books», приложения рассчитывает цену заказа с учётом скидок имеющихся у пользователя и предлагает подтвердить либо отклонить заказ.



Страница подтверждения заказа

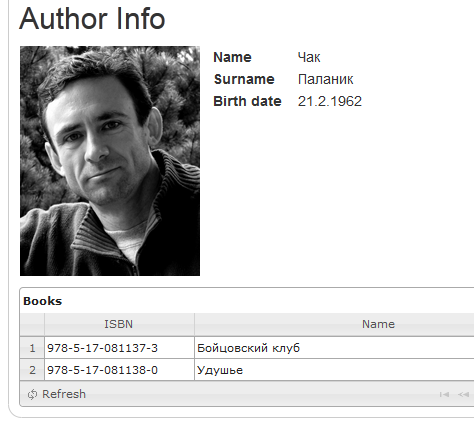
Если пользователь согласен с предлагаемой ценой, ему отправляется архив со всеми выбранными электронными книгами.



Загруженный архив с книгами

Также пользователь имеет возможность просматривать подробную информацию о книгах и авторах.

Профиль книги

Профиль автора

# Заключение

Таким образом, в курсовом проекте было реализовано приложение «Сеть СТО» с использованием программных средств Microsoft SQL Server 2012, C# и ASP.NET MVC 4.5.

Были решены следующие задачи:

1. Изучены материалы, касающиеся систем и технологий для их реализации.

2. Произведено инфологическое моделирование предметной области

3. Осуществлена программная реализация проекта.

В будущем планируется расширение функциональности приложения. В частности, расширение возможностей клиента, внедрение системы скидок, оповещение по телефону или электронной почте о приближающемся ТО. Расширение видов отчётов для администратора.

# Список использованных источников

1. Буч, Г. Объектно-ориентированный анализ и проектирование с примерами приложений / Буч Г. [и др]; – 3-е изд. – Москва: И.Д. Вильяме, 2010. – 720 с
2. Дейв Крейн, Эрик Паскарелло, Дарен Джеймс. Ajax в действии.: Пер. с англ. – М.: Издательский дом «Вильяме», 2006 – 640 с.
3. Jess Chadwick, Todd Snyder, Hrusikesh Panda. Programming ASP.NET MVC 4; O'Reilly Media, September 2012 - 492
4. Буч, Г. Объектно-ориентированный анализ и проектирование с примерами приложений / Буч Г. [и др]; – 3-е изд. – Москва: И.Д. Вильяме, 2010. – 720 с.
5. Уотсон, К. С# 4.0 и платформа .NET 4 для профессионалов. / Нейгел К., Ивьен, Б., Глинн, Дж., Уотсон, К.; Пер. с англ. – Москва: И.Д.Вильямc. - 2011. – 1440 с.
6. Nagel, C. С#4.5 и платформа .NET 4.5 для профессионалов / Nagel C., Evjen, B., Watson, K., Skinner, M., Glynn J.; Пер. с англ. – Москва: И.Д.Вильямc -2008. – 1376 c.
7. Дейв Крейн, Эрик Паскарелло, Дарен Джеймс. Ajax в действии.: Пер. с англ. – М.: Издательский дом «Вильяме», 2006 – 640 с.