Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Магнитогорский государственный технический университет   
им Г.И. Носова»

Многопрофильный колледж

Отделение №1 Информационные технологии и документоведение

**КУРСОВОЙ ПРОЕКТ**

**РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

по ПМ.02. Разработка и администрирование баз данных

МДК.02.02. Технология разработки и защиты баз данных

на тему: Разработка автоматизированной информационной системы деятельности мелкооптового книжного магазина.

Исполнитель: Клюшин М.А. студент 4 курса, группа ППр-14-1

Руководитель: Зорина И.Г., преподаватель МпК

Работа допущена к защите «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_ 20\_\_ г. \_\_\_\_\_\_\_\_\_\_\_\_\_\_

Работа защищена «\_\_\_» \_\_\_\_\_\_\_ 20\_\_ г. с оценкой \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Магнитогорск, 2017

**Содержание**

[Введение 3](#_Toc501727635)

[1 Проектирование базы данных 4](#_Toc501727636)

[1.1 Анализ предметной области 5](#_Toc501727637)

[1.2 Логическое и физическое проектирование 6](#_Toc501727638)

[2 Разработка базы данных 10](#_Toc501727639)

[2.1 Создание таблиц и связей между ними 12](#_Toc501727640)

[2.2 Создание запросов и отчетов 20](#_Toc501727641)

[2.3 Разработка интерфейса пользователя 22](#_Toc501727642)

[2.4 Создание пользователей и разграничение прав доступа 27](#_Toc501727643)

[3 Руководство пользователя 28](#_Toc501727644)

[Заключение 30](#_Toc501727645)

[Список используемой литературы 31](#_Toc501727646)

[Приложение А 32](#_Toc501727647)

# Введение

# Научно-технический прогресс, набравший к концу ХХ в. головокружительную скорость, послужил причиной появления такого чуда современности как компьютер и компьютерные технологии. С совершенствованием компьютеров совершенствовались и игры, привлекая все больше и больше людей. Позже появились мобильные телефоны, на которые с большим успехом стали создаваться игры различных жанров, ничем не уступающие своим компьютерным аналогам.

Мобильные телефоны давно уже превратились из средств связи в многофункциональные мультимедийные устройства, которым по силам удовлетворить очень многие запросы своих владельцев - касается ли это музыки, видео, доступа в интернет или игр. Современная мобильная игровая индустрия является важным направлением в развитии сотовых телефонов, о чем свидетельствуют хотя бы цифры многочисленных исследований. На многих рынках мобильные развлечения занимают, вслед за компьютерными играми, ведущие позиции в игровой индустрии.

# Играют не только дети, но и взрослые. И играют, как правило, в рабочее время. Сейчас мобильные устройства распространены повсеместно. Наверное, нет ни одного ученика, который бы не любил играть в игры. Некоторые из нас готовы играть день и ночь. Родители же напротив, запрещают играть в компьютерные и мобильные игры или устанавливают для игры очень маленький промежуток времени, хотя и сами порой непрочь поиграть.

# На сегодняшний день компьютерная техника достигла такого уровня развития, что позволяет программистам разрабатывать очень реалистичные игры с хорошим графическим и звуковым оформлением не только для компьютеров, но и для мобильных устройств.

Цель данного курсового проекта разработка игры в жанре платформер.

Исходя из поставленной цели, были сформулированы следующие задачи:

1. Анализ предметной области и выбор технологии разработки;
2. Создание спрайтов персонажа, врагов и игровых объектов;
3. Создание анимаций движения персонажа и врагов;
4. Разработка логики движений и взаимодействий персонажа с игровым миром;
5. Разработка логики движений и взаимодействий врагов с персонажем;
6. Создание механики генерации уровней;
7. Реализовать 1 «боса».;
8. Создать логику перехода между уровнями;
9. Создание клиентской части (интерфейс для пользователя);
10. Создание формирования отчета.

В настоящее время очень популярна автоматизация рабочего процесса в различных отраслях человеческой деятельности. В связи с этим актуальным стало программное обеспечение с использованием баз данных и средствами позволяющими автоматизировать выполнение большого количества операций. Этим требованиям наибольшим образом соответствуют различные информационные системы.

Информационные системы ассоциируются с системой методов, способов и средств регистрации, сбора, хранения, поиска, обработки и передачи информации на основе применения средств вычислительной техники. Результатом функционирования информационных систем является информационная продукция — документы, информационные массивы, базы данных и информационные услуги, на основе которых принимаются решения.

Информационные системы должны удовлетворять следующим требованиям: гибкость, надежность, эффективность и безопасность, а также должны быть удобными и понятными для обычного пользователя. В связи с этим информационные системы стали использоваться не только на крупных предприятиях, но и получили свою популярность среди мелких оптовых баз, магазинов и различных сервисов.

Современные условия функционирования мелкооптовых книжных магазинов требует разработки более совершенных систем учета и управления. Это связано с тем, что информационные системы значительно упрощают выполнение учета поставляемой и продаваемой продукции, позволяют отслеживать потребности клиентов. Информационная система в целом, позволяет оптимизировать работу магазина, тем самым увеличивая ему прибыть.

Цель данного курсового проекта разработка автоматизированной информационной системы для деятельности мелкооптового книжного магазина. Предметом деятельности является оказание помощи клиентам в приобретении книг.

Исходя из поставленной цели, были сформулированы следующие задачи:

1. Анализ предметной области и выбор технологии разработки;
2. Проектирование и создание структуры базы данных;
3. Разработка базы данных в выбранной СУБД;
4. Разработка представлений, запросов;
5. Разграничение прав пользователей;
6. Разработка клиентской часи приложения (интерфейс для пользователя);
7. Создание формирования отчета.

Платформер (англ. platformer) — жанр компьютерных игр, в которых основной чертой игрового процесса является прыгание по платформам, лазанье по лестницам, собирание предметов, обычно необходимых для завершения уровня. Некоторые предметы, называемые пауэр-апами (англ. power-up), наделяют управляемого игроком персонажа особой силой, которая обычно иссякает со временем (к примеру: силовое поле, ускорение, увеличение высоты прыжков). Коллекционные предметы, оружие и «пауэер-ап» собираются обычно простым прикосновением персонажа и для применения не требуют специальных действий со стороны игрока. Реже предметы собираются в «инвентарь» героя и применяются специальной командой (такое поведение более характерно для аркадных головоломок). Сходный жанр компьютерных игр SideScroller.

Противники (называемые «врагами»), всегда многочисленные и разнородные, обладают примитивным искусственным интеллектом, стремясь максимально приблизиться к игроку, либо не обладают им вовсе, перемещаясь по круговой дистанции или совершая повторяющиеся действия. Соприкосновение с противником обычно отнимает жизненные силы у героя или вовсе убивает его. Иногда противник может быть нейтрализован либо прыжком ему на голову, либо из оружия, если им обладает герой. Смерть живых существ обычно изображается упрощённо или символически (существо исчезает или проваливается вниз за пределы экрана).

Уровни, как правило, изобилуют секретами (скрытые проходы в стенах, высокие или труднодоступные места), нахождение которых существенно облегчает прохождение и подогревает интерес игрока.

Игры подобного жанра характеризуются нереалистичностью, рисованной мультяшной графикой. Героями таких игр обычно бывают мифические существа (к примеру: драконы, гоблины) или антропоморфные животные.

# 1 Проектирование базы данных

Выбор СУБД является одним из важнейших моментов в разработке проекта базы данных, так как он принципиальным образом влияет на весь процесс проектирования БД и реализацию Информационной системы.

Под базой данных понимается множество взаимосвязанных элементарных групп данных (информации), которые могут обрабатываться одной или несколькими прикладными системами.

Современная база данных должна удовлетворять следующим основным требованиям:

* Высокое быстродействие (малое время отклика на запрос). Время отклика - промежуток времени от момента запроса к БД до фактического получения данных.
* Простота обновления данных.
* Независимость данных - возможность изменения логической и физической структуры БД без изменения представлений пользователей. Независимость данных предполагает инвариантность к характеру хранения данных, программному обеспечению и техническим средствам. Она обеспечивает минимальные изменения структуры БД при изменениях стратегии доступа к данным и структуры самих исходных данных.
* Совместное использование данных многими пользователями.
* Безопасность данных - защита данных от преднамеренного или непреднамеренного нарушения секретности, искажения или разрушения.

Информационная система состоит из следующих компонентов:

* Серверная часть – программное обеспечение, которое управляет доступом к данным и обеспечивает указанные функциональные возможности системы с базой данных.
* Клиентская часть – приложение, которое формирует запросы к серверной части и обрабатывает ответы от него.
* Пользователи – люди, которые работают с приложением.

# 1.1 Анализ предметной области

В данном курсовом проекте в качестве предметной области рассматривается организация обслуживания клиентов мелкооптового книжного магазина.

Одним из основных принципов обслуживания клиентов является предоставление актуальной информации о товаре: название товара, количество на складе и краткое описание.

При необходимости получения дополнительной информации или же желании приобрести товар, клиент обращается в магазин.

В магазине при первичном обращении на клиента заводится электронная учетная запись магазина (с присвоением идентификационного номера), в которую вносятся следующие сведения о клиенте:

* фамилия, имя, отчество (полностью);
* телефон;

Это позволяет выполнить хранение и обработку данных о приобретаемых клиентом книгах и дает ему возможность просмотреть на какой стадии находится его заказ.

Предполагается, что база данных будет использоваться сотрудниками магазина такими как продавцы-консультанты, менеджеры и администраторы а также зарегистрированными клиентами магазина и не зарегистрировавшимися пользователями (удаленно).

Заказ формируется на каждую новую покупку клиента.

При просмотре списка книг клиенту предоставляется актуальная информация. При формировании заказа клиент должен указать книги которые он хочет приобрести а так же их количество. При успешном оформлении заказа должен производиться учет оставшихся книг на складе.

Организация рабочего процесса в книжном магазине является довольно трудоемкой. Продавцы-консультанты должны знать наиболее полную информацию о продаваемых книгах, а также иметь возможность найти ее в кратчайшие сроки. Администратор магазина должен иметь возможность быстрого доступа к информации о сотрудниках, следить за выполнениями заказов и т.д.

Также сотрудники магазина могут делать заказы на поставку книг, соответственно необходимо обеспечить возможность доступа к информации о проводимых заказах.

# 1.2 Логическое и физическое проектирование

Логическое проектирование базы данных заключается в преобразовании концептуальной модели данных в логическую модель данных предприятия с учетом выбранного типа СУБД. Логическая модель данных является источником информации для этапа физического проектирования.

Основной целью этапа создания логической модели базы данных является преобразование информационной модели предметной области базы данных в логическую модель реляционной базы данных.

Построенная логическая модель данных (рис. 1) в дальнейшем будет востребована на этапе физического проектирования, а также на этапе эксплуатации и сопровождении уже готовой системы, позволяя наглядно представлять любые вносимые в базу данных зменения.

1

N

1

N

1

1

1

1

1

1

1

N

1

N

1

1

N

1

1

1

N

1

1

1

N

1

1

1

1

1

1

N

1

N

1

1

N

1

1

1

Авторы

Книги

Издательства

Поставки

Клиенты

Поставщики

Заказы

Заказанные книги

Статусы

Имеет

Издает

Входит

Входят

Входит

Входят

Входят

Входят

Входят

Входит

Рисунок 1 – Логическая структура базы данных «Мелкооптовый книжный магазин»

1. Authors.

Данная таблица содержит данные о авторах книг и является главной по отношению к таблице «Books».

Структура таблицы «Authors»:

* idAuthor – идентификационный номер автора (ключевое поле)
* nameAuthor – ФИО автора (в формате И.О. Фамилия)

1. Company

Данная таблица содержит данные о издательствах книг и является главной по отношению к таблице «Books».

Структура таблицы «Company»:

* idCompany – идентификационный номер издательства (ключевое поле)
* nameCompany – название компании

1. Providers.

Данная таблица содержит данные о поставщиках книг и является главной по отношению к таблице «Supplys».

Структура таблицы «Providers»:

* idProvider – идентификационный номер поставщика (ключевое поле)
* nameProvider – название поставщика

1. Statuses.

Данная таблица содержит данные о статусе выполнения заказа, и книг в заказе, является главной по отношению к таблицам: «Orders», «List».

Структура таблицы «Statuses»:

* idStatus – идентификационный номер статуса (ключевое поле)
* nameStatus – название статуса

1. Books

Данная таблица содержит данные о книгах и является главной по отношению к таблицам «Supplys» и «List».

Структура таблицы «Books»:

* idBook – идентификационный номер книги (ключевое поле)
* nameBook – название статуса
* Author – является внешним ключом по отношению к таблице «Authors»
* Comment – краткое описание книги
* Pages – количество страниц
* Company – является внешним ключом по отношению к таблице «Company»
* Year – год издания книги
* Disc – наличие диска

1. Supplys

Данная таблица содержит данные о поставках книг и является главной по отношению к таблице «List».

Структура таблицы «Supplys»:

* idSupply – идентификационный номер поставки (ключевое поле)
* Book – является внешним ключом по отношению к таблице «Books»
* QtyBook – количество книг
* Provider – является внешним ключом по отношению к таблице «Providers»
* Price – цена за одну книгу

1. Clients

Данная таблица содержит данные о клиентах и является главной по отношению к таблице «Orders».

Структура таблицы «Clients»:

* idClient – идентификационный номер клиента (ключевое поле)
* Surname – Фамилия клиента
* Name – Имя клиента
* Lastname – Отчество клиента
* Phone – телефон клиента
* Login – логин клиента
* Password – пароль клиента

1. Orders

Данная таблица содержит данные о заказах и является главной по отношению к таблице «List».

Структура таблицы «Clients»:

* idOrder – идентификационный номер заказа (ключевое поле)
* Client – является внешним ключом по отношению к таблице «Clients»
* dateOrder – дата заказа
* timeOrder – время заказа
* Status - является внешним ключом по отношению к таблице «Statuses»

1. List

Данная таблица содержит данные о заказанных книгах.

Структура таблицы «List»:

* idList – идентификационный номер заказанной книги (ключевое поле)
* Orders – является внешним ключом по отношению к таблице «Orders»
* Supplys – является внешним ключом по отношению к таблице «Supplys»
* Book – является внешним ключом по отношению к таблице «Books»
* listQty – количество заказанных книг
* listStatus - является внешним ключом по отношению к таблице «Statuses»

1. Staff

Данная таблица содержит данные о сотрудниках магазина.

Структура таблицы «Staff»:

* idStaff – идентификационный номер сотрудника (ключевое поле)
* Surname – Фамилия сотрудника
* Name – Имя сотрудника
* Lastname – Отчество сотрудника
* Address – Адрес сотруднка
* Pasport – паспортные данные сотрудника
* Phone – телефон сотрудника
* Role – должность сотрудника
* Login – логин сотрудника
* Password – пароль сотрудника

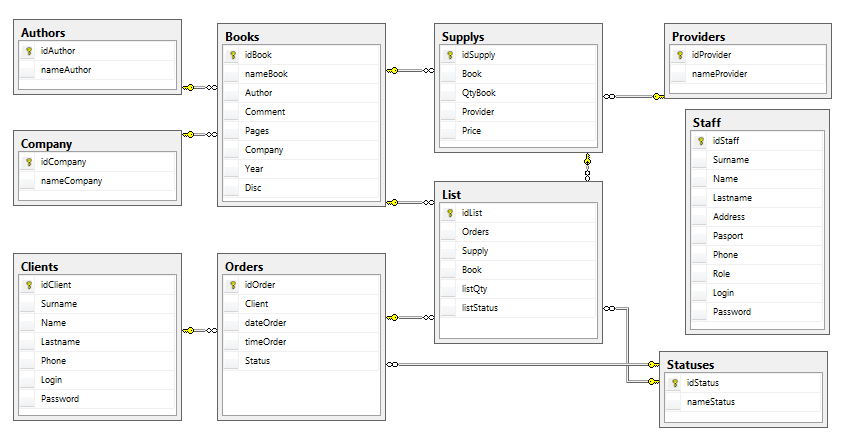


Рисунок 2 – Схема базы данных «Мелкооптовый книжный магазин»

# 2 Разработка базы данных

Microsoft SQL Server — система управления реляционными базами данных (РСУБД), разработанная корпорацией Microsoft. Основной используемый язык запросов — Transact-SQL, создан совместно Microsoft и Sybase. Transact-SQL является реализацией стандарта ANSI/ISO по структурированному языку запросов (SQL) с расширениями. Используется как платформа для работы с базами данных, обеспечивающая возможность крупномасштабной оперативной обработки транзакций, хранения данных и работы с приложениями для электронной торговли; а также является платформой бизнес-аналитики для создания решений по интеграции данных, анализу и составлению отчетов.

SQL Server содержит множество настраиваемых средств безопасности, действующих с высокой точностью. Эти функции позволяют администраторам реализовать всестороннюю защиту, которая оптимизирована для конкретных рисков безопасности в их системах.

SQL Server представляет «среды» для облегчения выполнения задач разработки и управления: среда SQL Server Management Studio и среда Business Intelligence Development Studio. В среде Management Studio можно разрабатывать решения компонента SQL Server Database Engine и решения по уведомлению и управлять ими, управлять развернутыми решениями служб Analysis Services, запускать пакеты служб Integration Services и управлять ими, управлять серверами, отчетами и моделями отчетов служб Reporting Services. В среде BI Development Studio можно разрабатывать решения бизнес-аналитики: проекты служб Analysis Services используются для разработки кубов, измерений и структур интеллектуального анализа данных; проекты служб Reporting Services — для создания отчетов; проекты моделей отчетов — для определения моделей отчетов и проекты служб Integration Services — для создания пакетов.

В этих средах SQL Server предоставляет графические инструменты, необходимые для проектирования, разработки, развертывания и администрирования реляционных баз данных, аналитических объектов, пакетов преобразования данных, топологий репликации, отчетов, серверов отчетов и серверов уведомлений.

Компонент Database Engine — основная служба для хранения, обработки и защиты данных. Он обеспечивает управляемый доступ и быструю обработку транзакций, достаточную даже для самых требовательных к предоставляемым данным приложений.

Компонент Database Engine используется для создания реляционных баз данных для оперативной обработки транзакций или интерактивной аналитической обработки. Сюда входит создание таблиц для хранения данных и объектов баз данных, таких как индексы, представления и хранимые процедуры для просмотра и защиты данных и для управления ими. Для управления объектами баз данных можно использовать среду SQL Server Management Studio, для фиксации событий сервера — SQL Server Profiler.

# 2.1 **Создание таблиц и связей между ними**

Создание таблицы «Authors» при помощи SQL запроса. Характеристики полей представлены в таблице 1.

CREATE TABLE [dbo].[Authors](

[idAuthor] [int] IDENTITY(1,1) NOT NULL,

[nameAuthor] [varchar](50) NULL,

CONSTRAINT [PK\_Authors] PRIMARY KEY CLUSTERED

(

[idAuthor] ASC

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

) ON [PRIMARY]

Таблица 1 – Характеристика полей таблицы «Authors»

|  |  |  |
| --- | --- | --- |
| Имя поля | Тип данных | Свойства полей |
| idAuthor | int | Ключевое поле |
| nameAuthor | varchar | Подпись: имя автора |
|  |  | Размер поля: 50 |

Создание таблицы «Company» при помощи SQL запроса. Характеристики полей представлены в таблице 2.

CREATE TABLE [dbo].[Company](

[idCompany] [int] IDENTITY(1,1) NOT NULL,

[nameCompany] [varchar](50) NULL,

CONSTRAINT [PK\_Company] PRIMARY KEY CLUSTERED

(

[idCompany] ASC

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

) ON [PRIMARY]

Таблица 2 – Характеристика полей таблицы «Company»

|  |  |  |
| --- | --- | --- |
| Имя поля | Тип данных | Свойства полей |
| idCompany | int | Ключевое поле |
| nameCompany | varchar | Подпись: имя издательства |
|  |  | Размер поля: 50 |

Создание таблицы «Providers» при помощи SQL запроса. Характеристики полей представлены в таблице 3.

CREATE TABLE [dbo].[Providers](

[idProvider] [int] IDENTITY(1,1) NOT NULL,

[nameProvider] [varchar](50) NULL,

CONSTRAINT [PK\_Providers] PRIMARY KEY CLUSTERED

(

[idProvider] ASC

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

) ON [PRIMARY]

Таблица 3 – Характеристика полей таблицы «Providers»

|  |  |  |
| --- | --- | --- |
| Имя поля | Тип данных | Свойства полей |
| idProvider | int | Ключевое поле |
| nameProvider | varchar | Подпись: имя поставщика |
|  |  | Размер поля: 50 |

Создание таблицы «Statuses» при помощи SQL запроса. Характеристики полей представлены в таблице 4.

CREATE TABLE [dbo].[Statuses](

[idStatus] [int] IDENTITY(1,1) NOT NULL,

[nameStatus] [varchar](50) NULL,

CONSTRAINT [PK\_Statuses] PRIMARY KEY CLUSTERED

(

[idStatus] ASC

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

) ON [PRIMARY]

Таблица 4 – Характеристика полей таблицы «Statuses»

|  |  |  |
| --- | --- | --- |
| Имя поля | Тип данных | Свойства полей |
| idStatus | int | Ключевое поле |
| nameStatus | varchar | Подпись: название автора |
|  |  | Размер поля: 50 |

Создание таблицы «Books» при помощи SQL запроса. . Характеристики полей представлены в таблице 5.

CREATE TABLE [dbo].[Books](

[idBook] [int] IDENTITY(1,1) NOT NULL,

[nameBook] [varchar](50) NULL,

[Author] [int] NULL,

[Comment] [varchar](max) NULL,

[Pages] [int] NULL,

[Company] [int] NULL,

[Year] [varchar](4) NULL,

[Disc] [bit] NULL,

CONSTRAINT [PK\_Books] PRIMARY KEY CLUSTERED

(

[idBook] ASC

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

) ON [PRIMARY] TEXTIMAGE\_ON [PRIMARY]

Таблица 5 – Характеристика полей таблицы «Books»

|  |  |  |
| --- | --- | --- |
| Имя поля | Тип данных | Свойства полей |
| idBook | int | Ключевое поле |
|  |  | Автоинкремент |
| nameBook | varchar | Подпись: название книги |
|  |  | Размер поля: 50 |
| Author | int | Подпись: id автора |
|  |  |  |
| Comment | varchar | Подпись: краткое описание |
|  |  | Размер поля: max |
| Pages | int | Подпись: кол-во страниц |
|  |  |  |
| Company | int | Подпись: id издательства |
|  |  |  |
| Year | int | Подпись: год издательства |
|  |  | Размер поля: 4 |
| Disc | bin | Подпись: наличие диска |
|  |  | Размер поля: 1 |

Создание таблицы «Supplys» при помощи SQL запроса. Характеристики полей представлены в таблице 6.

CREATE TABLE [dbo].[Supplys](

[idSupply] [int] IDENTITY(1,1) NOT NULL,

[Book] [int] NULL,

[QtyBook] [int] NULL,

[Provider] [int] NULL,

[Price] [decimal](18, 2) NULL,

CONSTRAINT [PK\_Supplys] PRIMARY KEY CLUSTERED

(

[idSupply] ASC

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

) ON [PRIMARY]

Таблица 6 – Характеристика полей таблицы «Supplys»

|  |  |  |
| --- | --- | --- |
| Имя поля | Тип данных | Свойства полей |
| idSupplys | int | Ключевое поле |
|  |  | Автоинкремент |
| Book | int | Подпись: id книги |
|  |  |  |
| QtyBook | int | Подпись: кол-во книг |
|  |  |  |
| Provider | int | Подпись: id поставщика |
|  |  |  |
| Price | int | Подпись: цена книги |
|  |  | Размерность: 18, 2 |

Создание таблицы «List» при помощи SQL запроса. Характеристики полей представлены в таблице 7.

CREATE TABLE [dbo].[List](

[idList] [int] IDENTITY(1,1) NOT NULL,

[Orders] [int] NULL,

[Supply] [int] NULL,

[Book] [int] NULL,

[listQty] [int] NULL,

[listStatus] [int] NULL,

CONSTRAINT [PK\_List] PRIMARY KEY CLUSTERED

(

[idList] ASC

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

) ON [PRIMARY]

Таблица 7 – Характеристика полей таблицы «List»

|  |  |  |
| --- | --- | --- |
| Имя поля | Тип данных | Свойства полей |
| idList | int | Ключевое поле |
|  |  | Автоинкремент |
| Orders | int | Подпись: id заказа |
|  |  |  |
| Supply | int | Подпись: id поставки |
|  |  |  |
| Book | int | Подпись: id книги |
|  |  |  |
| listQty | int | Подпись: кол-во книг |
|  |  |  |
| listStatus | int | Подпись: id статуса книги |
|  |  |  |

Создание таблицы «Orders» при помощи SQL запроса. Характеристики полей представлены в таблице 8.

CREATE TABLE [dbo].[Orders](

[idOrder] [int] IDENTITY(1,1) NOT NULL,

[Client] [int] NULL,

[dateOrder] [date] NULL,

[timeOrder] [time](7) NULL,

[Status] [int] NULL,

CONSTRAINT [PK\_Orders] PRIMARY KEY CLUSTERED

(

[idOrder] ASC

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

) ON [PRIMARY]

Таблица 8 – Характеристика полей таблицы «Orders»

|  |  |  |
| --- | --- | --- |
| Имя поля | Тип данных | Свойства полей |
| idOrder | int | Ключевое поле |
|  |  | Автоинкремент |
| Client | int | Подпись: id клиента |
|  |  |  |
| dateOrder | int | Подпись: дата заказа |
|  |  |  |
| timeOrder | int | Подпись: время заказа |
|  |  |  |
| Status | int | Подпись: кол-во книг |
|  |  |  |

Создание таблицы «Orders» при помощи SQL запроса. Характеристики полей представлены в таблице 9.

CREATE TABLE [dbo].[Orders](

[idOrder] [int] IDENTITY(1,1) NOT NULL,

[Client] [int] NULL,

[dateOrder] [date] NULL,

[timeOrder] [time](7) NULL,

[Status] [int] NULL,

CONSTRAINT [PK\_Orders] PRIMARY KEY CLUSTERED

(

[idOrder] ASC

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

) ON [PRIMARY]

Таблица 9 – Характеристика полей таблицы «Clients»

|  |  |  |
| --- | --- | --- |
| Имя поля | Тип данных | Свойства полей |
| idClients | int | Ключевое поле |
|  |  | Автоинкремент |
| Surname | varchar | Подпись: id клиента |
|  |  | Размер поля: 50 |
| Name | varchar | Подпись: дата заказа |
|  |  | Размер поля: 50 |
| Lastname | varchar | Подпись: время заказа |
|  |  | Размер поля: 50 |
| Phone | varchar | Подпись: кол-во книг |
|  |  | Размер поля: 50 |
| Login | varchar | Подпись: кол-во книг |
|  |  | Размер поля: 50 |
| Password | varchar | Подпись: кол-во книг |
|  |  | Размер поля: 50 |

Создание таблицы «Staff» при помощи SQL запроса. Характеристики полей представлены в таблице 10.

CREATE TABLE [dbo].[Staff](

[idStaff] [int] IDENTITY(1,1) NOT NULL,

[Surname] [varchar](50) NULL,

[Name] [varchar](50) NULL,

[Lastname] [varchar](50) NULL,

[Address] [varchar](50) NULL,

[Pasport] [varchar](15) NULL,

[Phone] [varchar](15) NULL,

[Role] [varchar](50) NULL,

[Login] [varchar](50) NULL,

[Password] [varchar](50) NULL,

CONSTRAINT [PK\_Staff] PRIMARY KEY CLUSTERED

( [idStaff] ASC

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

) ON [PRIMARY]

Таблица 10 – Характеристика полей таблицы «Staff»

|  |  |  |
| --- | --- | --- |
| Имя поля | Тип данных | Свойства полей |
| idStaff | int | Ключевое поле |
|  |  | Автоинкремент |
| Surname | varchar | Подпись: имя сотрудника |
|  |  | Размер поля: 50 |
| Name | varchar | Подпись: фамилия сотрудника |
|  |  | Размер поля: 50 |
| Lastname | varchar | Подпись: отчество сотрудника |
|  |  | Размер поля: 50 |
| Address | varchar | Подпись: адрес сотрудника |
|  |  | Размер поля: 50 |
| Paspost | varchar | Подпись: паспорт сотрудника |
|  |  | Размер поля: 15 |
| Phone | varchar | Подпись: телефон сотрудника |
|  |  | Размер поля: 15 |
| Role | varchar | Подпись: должность |
|  |  | Размер поля: 50 |
| Login | varchar | Подпись: логин сотрудника |
|  | varchar | Размер поля: 50 |
| Password | varchar | Подпись: пароль сотрудника |
|  | varchar | Размер поля: 50 |

# 2.2 **Создание запросов и отчетов**

SQL - язык программирования, применяемый для создания, модификации и управления данными в реляционной базе данных, управляемой соответствующей системой управления базами данных.

Реляционная база данных – это тело связанной информации, сохраняемой в двухмерных таблицах.

Язык SQL предназначен для манипулирования данными в реляционных базах данных, определения структуры баз данных и для управления правами доступа к данным и многопользовательской среде.

В язык SQL в качестве составной части входа:

* Язык манипулирования данными используется, для манипулирования данными в таблицах. Несколько основных команд:

- Select (выбрать)

- Insert (вставить)

- Update (обновить)

- Delete (удалить)

* Язык манипулирования данными используется для создания и изменения структуры базы данных и ее составных частей – таблиц, индексов, представлений (виртуальных таблиц), а также триггеров сохраненных процедур. Основными его командами является:

- Create Database (создать базу данных)

- Сreate Table (создать таблицу) и т.д.

Это не отдельные языки, а различные команды одного языка. Такое деление приведено с точки зрения различного функционального назначения этих команд.

SQL запрос – это либо вызов данных из базы, либо обращение к базе данных, которое приводит к изменению данных в базе.

Запрос в SQL состоит из одного или нескольких операторов, следующих один за другим и разделенных точкой с запятой.

Примеры:

Запрос на обновление статусов книг в заказе при изменении статуса заказа:

//Подключение к серверу и базе

SqlConnection MyConnection = new SqlConnection(DataBaseConection.ConnectionString);

MyConnection.Open(); //Открываем подключение

//Запрос на изменение статуса заказанных книг

SqlCommand UpdateStatusOrder = new SqlCommand("Update [Orders] Set [Status] = '" + statusComboBox.SelectedValue + "' Where [idOrder] = '" + idOrderTextBox.Text + "'", MyConnection);

UpdateStatusOrder.ExecuteNonQuery();

SqlCommand UpdateStatusList = new SqlCommand("Update [List] Set [listStatus] = '" + statusComboBox.SelectedValue + "' Where [Orders] = '" + idOrderTextBox.Text + "'", MyConnection);

UpdateStatusList.ExecuteNonQuery();

MyConnection.Close();

Запрос на добавление выбранной книги в заказ:

//Подключение к серверу и базе

SqlConnection MyConnection = new SqlConnection(DataBaseConection.ConnectionString);

MyConnection.Open(); //Открываем подключение

SqlDataReader SqlReader = null;

//Добавляем книги в таблицу "Книги в заказе" (List)

SqlCommand InsertListCommand = new SqlCommand("Insert Into [List] (Orders, Supply, Book, listQty, listStatus) Values ('" + Users.idUserOrder + "', '" + SupplyList + "', '" + BookList + "', '" + QtyList + "', '" + 1 + "')", MyConnection);

InsertListCommand.ExecuteNonQuery();

SqlCommand UpdateOrder = new SqlCommand("Update [Supplys] Set QtyBook = QtyBook - '" + QtyList + "' Where [idSupply] = '" + SupplyList + "'", MyConnection);

UpdateOrder.ExecuteNonQuery();

Запрос на удаление заказа если клиент при выходе не завершил заказ

//Подключение к серверу и базе

SqlConnection MyConnection = new SqlConnection(DataBaseConection.ConnectionString);

MyConnection.Open(); //Открываем подключение

SqlCommand DeleteOrder = new SqlCommand("Delete From [Orders] WHERE [idOrder] = '" + Users.idUserOrder + "'", MyConnection);

DeleteOrder.ExecuteNonQuery();

MyConnection.Close();

# 2.3 **Разработка интерфейса пользователя**

Интерфейс — общая граница между двумя функциональными объектами, совокупность средств, методов и правил взаимодействия (управления, контроля и т.д.) между элементами системы.

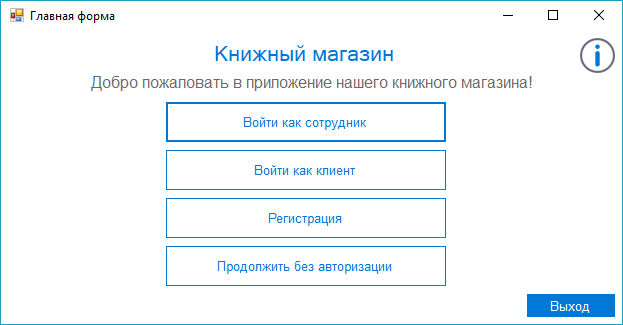
Для создания пользовательского интерфейса была выбрана  интегрированная среда разработки программного обеспечения Microsoft Visual Studio.

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.

Visual Studio включает в себя редактор исходного кода и возможностью простейшего рефакторинга кода. Встроенный отладчик может работать как отладчик уровня исходного кода, так и 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, 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.

Главной формой приложения является форма приветствия (Рис. 001.), на которой пользователь, в зависимости от того кем он является, может выполнить следующие действия:

* Войти как сотрудник – авторизация для сотрудников книжного магазина (Рис. 002.)
* Войти как клиент – авторизация для зарегистрировавшихся клиентов
* Регистрация – регистрация для новых клиентов (Рис. 003.)
* Продолжить без авторизации – просмотр ассортимента магазина не требующий авторизации с подробным выводом информации о каждой книге (Рис. 004)



Кнопка дополнительной информации

Рис. 001. Главная форма

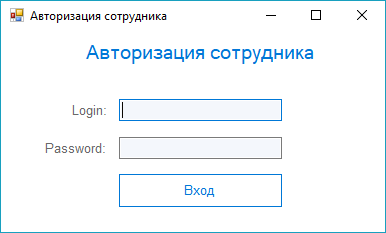


Рис.002. Авторизация сотрудника

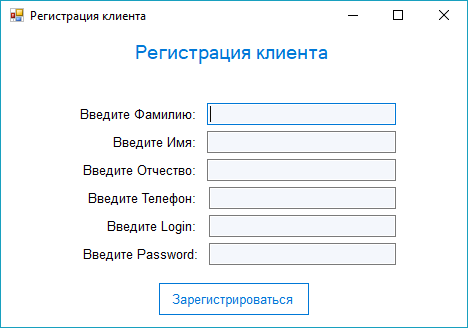


Рис. 003. Регистрация клиента

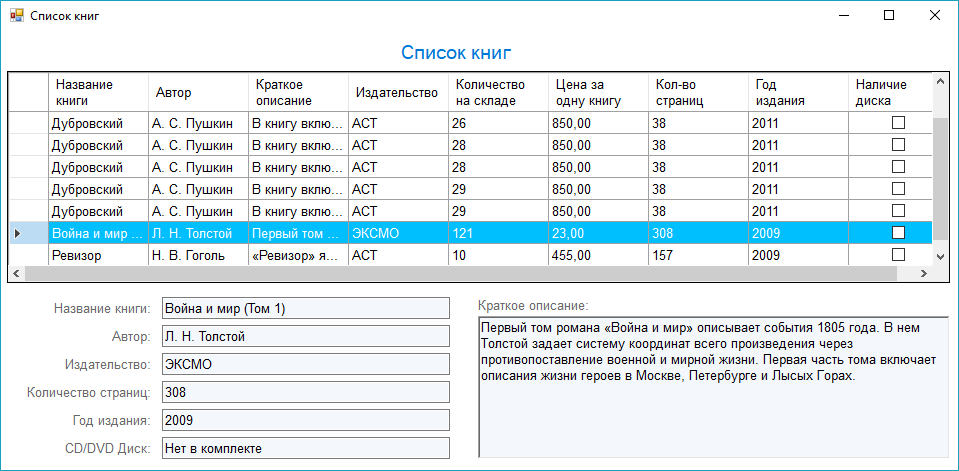


Рис. 004. Список продаваемых книг

Для авторизированного сотрудника в зависимости от должности выводится его форма управления с доступном ему функционалом (Рис. 005.). На приведенном рисунке отображена форма менеджера на которой отмечены серым недоступные ему таблицы.

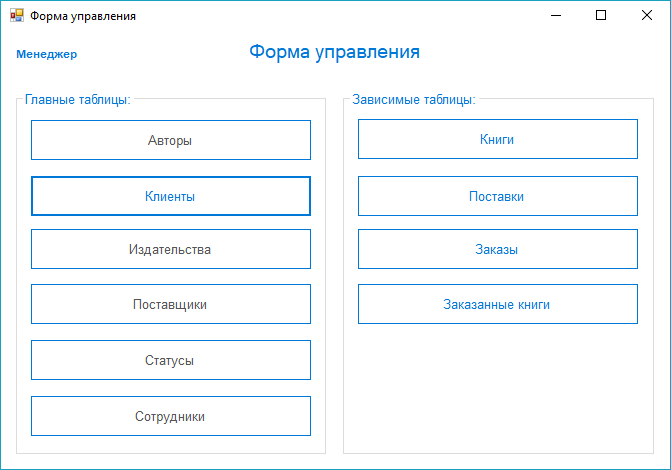


Рис. 005. Форма менеджера

Для просмотра и изменения данных в таблице используется форма вида (Рис. 006.), для главных таблиц и форма вида (Рис. 007.) с более расширенным функционалом для зависимых таблиц.

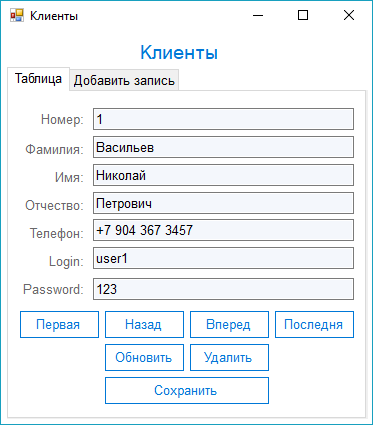


Рис. 006. Форма клиентов

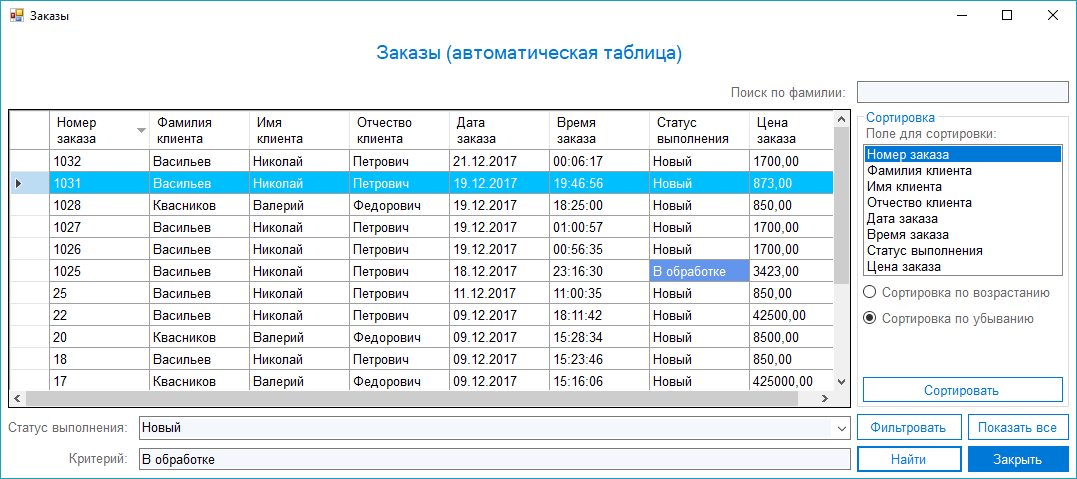


Рис. 007. Форма заказов

Для авторизированного клиента выводится форма оформления заказа (Рис. 008.), с подробным описанием книг. На которой он может посмотреть список доступных к покупке книг, создать заказ, добавить необходимые книги в заказ и завершить заказ. Если клиент выйдет с формы до завершения заказа, то заказ автоматически удаляется.

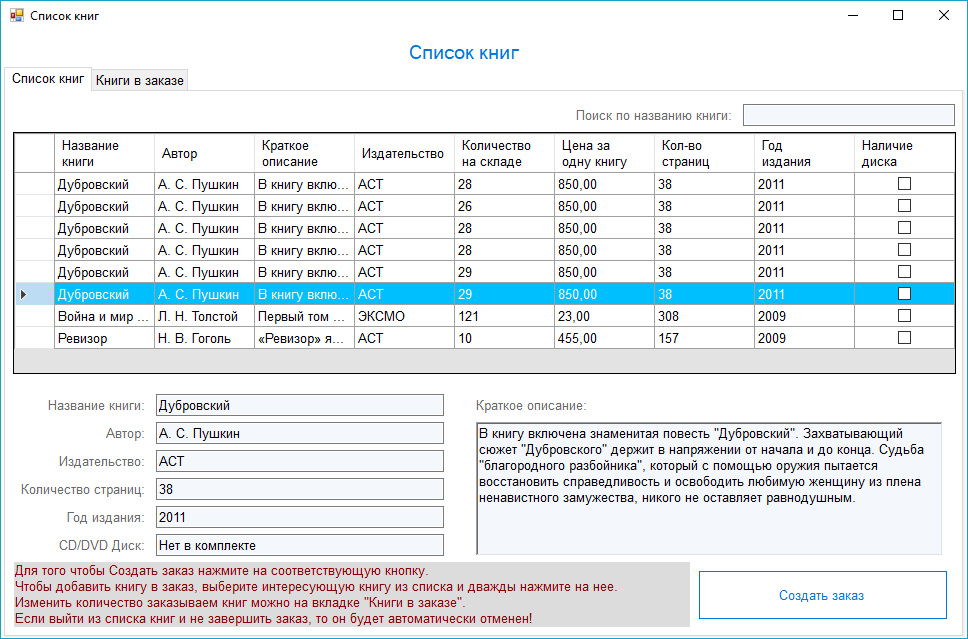


Рис. 008. Форма для авторизованного клиента

# 2.4 Создание пользователей и разграничение прав доступа

В связи с тем, что у данного приложения должно быть несколько пользователей с разными правами, необходимо разграничение прав доступа.

Всего в приложении 2 вида пользователей: Сотрудники и Обычные пользователи.

Сотрудники делятся на:

1. Администратор – имеет возможность добавления сотрудников и клиентов в базу, а также производите редактирование всех таблиц
2. Менеджер – имеет возможность на изменение данных в зависимых таблицах, а также добавление новых пользователей
3. Продавец – имеет возможность просмотра ограниченного кол-ва таблиц, как и менеджер может добавлять клиентов и производить продажу товара

Обычные пользователи делятся на:

1. Зарегистрированные пользователи – могут просматривать список книг и делать заказы
2. Незарегистрированные пользователи – могут только просматривать список имеющихся в продаже книг и получить дополнительную информацию

Разделение прав доступа для сотрудников осуществляется во время их регистрации, в ней каждому из сотрудников присваивается своя должность и в дальнейшем при авторизации проверяется какую именно должность имеет сотрудник. И на основе полученного результата открывается соответствующая должности форма.

Разделение прав пользователей ограничено лишь тем что они имеют доступ лишь к двум формам: форма списка книг с возможностью оформления заказа – для зарегистрировавшихся и форма списка книг без возможности оформления заказа – для незарегистрированных пользователей.

# 3 Руководство пользователя

Приложение для организации работы мелкооптового книжного магазина, предназначено для обеспечения учета заказов, проверки списка товара, организации поставок, предоставление списка актуального товара, имеющегося на складе и быстрого оформления заказа.

При входе в приложение вам становится доступна главная форма (Рис. 001.)

Если вы являетесь сотрудником, то нажмите на кнопку «Войти как сотрудник» и пройдите авторизацию.

Для пользователей, которые желают сделать заказ есть возможность просмотра списка книг без регистрации для этого нажмите на кнопку «Продолжить без авторизации». Если у вас нет учетной записи, то пройдите регистрацию нажав на соответствующую кнопку. Если у вас уже есть учетная запись клиента, то пройдите авторизацию нажав на кнопку «Войти как клиент».

Для авторизированных сотрудников:

После прохождения авторизации вам откроется форма вида (Рис. 005.), если на ней какие-то элементы вам недоступны, то они были заблокированы администратором.

Вы можете вносить изменения в любую из доступных вам таблицу.

Управление главными таблицами:

* Для добавления новой записи необходимо перейти на вкладку «Добавить запись» (В разных таблицах название может отличаться), введите данные во все поля и нажмите кнопку добавить.
* Для того чтобы удалить запись, выберите соответствующую запись из списка и нажмите кнопку «Удалить», а затем чтобы сохранить изменение нажмите на кнопку «Сохранить»
* Для того чтобы изменить данные, необходимо выбрать текстовое поле и после изменения в нем значения нажать кнопку «Сохранить».

Если данные поле добавления или удаления не отображаются попробуйте обновить данные, для этого нажмите на кнопку «Обновить».

Управление зависимыми таблицами:

* Для того чтобы добавить данные необходимо перейти на соответствующую вкладку и ввести необходимые значения, далее нажать на кнопку добавить.
* Для того чтобы удалить запись, выберите соответствующую запись из списка и нажмите кнопку «Удалить».
* Для изменения данных выберите необходимое поле и перейдите на вкладку «Изменить запись», после редактирования значений нажмите на кнопку «Изменить».

Для клиентов:

После того как вы авторизировались вам станет доступна форма просмотра книг и оформления заказа.

Для того чтобы сделать заказ вам необходимо нажать «Создать заказ», после чего будет создан пустой заказ. Чтобы добавить в заказ книги необходимо дважды нажать правую кнопку мыши, после чего вы увидите сообщение о добавлении книги в заказ. Чтобы изменить кол-во приобретаемых книг или завершить заказ вам необходимо перейти на вкладку «Книги в заказе», для изменения кол-ва выберите книгу из списка и поменяйте значение кол-ва, после нажмите кнопку «Записать». Для подтверждения заказа нажмите кнопку «Заказать».

Сразу после добавления заказа, заказ и книги которые в него входят имеют статус «Новый», после того как у одной из книг сотрудник поменяет значение на «Готов к выдаче», заказ будет иметь статус «В обработке». После того как все книги будут готовы к выдаче заказ примет статус «Готов к выдаче». Заказ будет считаться завершенным после того как клиент получит книги в магазине и сотрудник отметит у заказа статус «Завершен».

# Заключение

В процессе работы были сформулированы требования и задачи для разработки данной информационной системы. Большинство представленных задач, на данном этапе разработки были выполнены.

Разрабатываемое приложение для деятельности мелкооптового книжного магазина значительно упрощает учет товара на складе, позволяет легко добавлять данные в любую необходимую вам таблицу и осуществлять поиск и фильтрацию данных. А также позволяет отслеживать выполнения заказа, за счет автоматического изменения статуса.

Электронный каталок книг позволяет предоставлять актуальную информацию о любой имеющейся в продаже книге, что значительно упрощает формирование заказа пользователем.

Интерфейс программы не должен вызвать затруднение в освоении т.к. имеет панели дополнительной информации. А если проблемы все-таки возникнут, можно обратиться к руководству пользователя.

В дальнейшем программу можно модернизировать, добавив в нее:

* Проверку на корректность вводимых данных
* Добавление панелей поиска в некоторые формы
* Запросы на выборку по определенной дате
* Добавить обработку исключений для выполнения запросов
* Добавить возможность продажи книг для сотрудников

Интерфейс и структура таблиц базы данных были разработаны применительно к предметной области «Деятельность мелкооптового книжного магазина».

# Список используемой литературы

1. Гагарина, Л.Г. Введение в архитектуру программного обеспечения: учеб. пособие / Л.Г. Гагарина, А.Р. Федоров, П.А. Федоров. — М. : ФОРУМ : ИНФРА-М, 2017. — 320 с: [Электронный ресурс], <http://znanium.com/bookread2.php?book=615207>
2. [Гуриков, С. Р.](http://znanium.com/catalog.php?item=booksearch&code=.+%D0%A8%D0%B0%D0%BA%D0%B8%D0%BD%2C+%D0%92.+%D0%9D.+%D0%91%D0%B0%D0%B7%D0%BE%D0%B2%D1%8B%D0%B5+%D1%81%D1%80%D0%B5%D0%B4%D1%81%D1%82%D0%B2%D0%B0+%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F+%D0%BD%D0%B0+Visual+Basic+%D0%B2+%D1%81%D1%80%D0%B5%D0%B4%D0%B5+VisualStudio+Net.&page=4#none) Введение в программирование на языке Visual C# : учеб. пособие / С.Р. Гуриков. — М. : ФОРУМ : ИНФРА-М, 2017. — 447 с. [Электронный ресурс], Режим доступа: <http://znanium.com/bookread2.php?book=752394>
3. [Гагарина, Л. Г.](http://znanium.com/catalog.php?item=booksearch&code=%D0%90%D1%80%D1%85%D0%B8%D1%82%D0%B5%D0%BA%D1%82%D1%83%D1%80%D0%B0%20%D0%B8%20%D1%82%D0%B5%D1%85%D0%BD%D0%BE%D0%BB%D0%BE%D0%B3%D0%B8%D1%8F%20%D1%80%D0%B0%D0%B7%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D0%BA%D0%B8%20%D1%80%D0%B0%D1%81%D0%BF%D1%80%D0%B5%D0%B4%D0%B5%D0%BB%D0%B5%D0%BD%D0%BD%D0%BE%D0%B3%D0%BE%20%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D0%BE%D0%B3%D0%BE%20%D0%BE%D0%B1%D0%B5%D1%81%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%B8%D1%8F#none) Технология разработки программного обеспечения : учеб. пособие / Л.Г. Гагарина, Е.В. Кокорева, Б.Д. Виснадул ; под ред. Л.Г. Гагариной. — М. : ИД «ФОРУМ» : ИНФРА-М, 2018. — 400 с. [Электронный ресурс], Режим доступа: <http://znanium.com/bookread2.php?book=924760>
4. Гагарина, Л.Г. Введение в архитектуру программного обеспечения: учеб. пособие / Л.Г. Гагарина, А.Р. Федоров, П.А. Федоров. — М. : ФОРУМ : ИНФРА-М, 2017. — 320 с: [Электронный ресурс], <http://znanium.com/bookread2.php?book=615207>

# Приложение А

Листинг 1 Авторизация пользователей на примере таблицы сотрудников (Staff):

//Подключение к серверу и базе (мой)

SqlConnection MyConnection = new SqlConnection(DataBaseConection.ConnectionString); ;

MyConnection.Open(); //Открываем подключение

SqlCommand MyCommand = new SqlCommand("Select [Login], [Password], [Role] From [Staff] Where [Login] = '" + textBoxLogin.Text + "' and [Password] = '" + textBoxPassword.Text + "'", MyConnection); //Запрос к базе(Login, Password, Role)

SqlDataReader reader = null;

reader = MyCommand.ExecuteReader();

AdminForm adminf = new AdminForm();

if (reader.HasRows)

{

while (reader.Read())

{

//MessageBox.Show(query.GetSqlString(2).ToString());

switch (reader.GetSqlString(2).ToString())

{

case "Администратор":

Visible = false;

Users.Role = "admin";

adminf.Show();

break;

case "Менеджер":

Visible = false;

Users.Role = "manager";

adminf.Show();

break;

case "Продавец":

Visible = false;

Users.Role = "clerk";

ClerkForm clerkf = new ClerkForm();

clerkf.Show();

break;

}

}

}

else

{

textBoxPassword.Clear();

labelError.ForeColor = Color.Red;

labelError.Text = "Неверные Логин или Пароль!";

}

Листинг 2 Добавление данных в базу на примере таблицы книги (Books):

//Проверка полей на пустые значения или пробелы

if (!string.IsNullOrEmpty(nameBookTextBox.Text) && (!string.IsNullOrWhiteSpace(nameBookTextBox.Text)) &&

!string.IsNullOrEmpty(pagesTextBox.Text) && (!string.IsNullOrWhiteSpace(pagesTextBox.Text)) &&

!string.IsNullOrEmpty(yearTextBox.Text) && (!string.IsNullOrWhiteSpace(yearTextBox.Text)) &&

!string.IsNullOrEmpty(commentRichTextBox.Text) && (!string.IsNullOrWhiteSpace(commentRichTextBox.Text))

)

{

//Подключение к серверу и базе

SqlConnection MyConnection = new SqlConnection(DataBaseConection.ConnectionString);

//Открываем подключение

MyConnection.Open();

SqlCommand MyCommand = new SqlCommand("Insert Into [Books] (nameBook, Author, Comment, Pages, Company, Year, Disc) Values " +

"('" + nameBookTextBox.Text + "', '" + authorComboBox.SelectedValue.ToString() + "', '" + commentRichTextBox.Text + "', '" +

pagesTextBox.Text + "', '" + companyComboBox.SelectedValue.ToString() + "', '" + yearTextBox.Text + "', '" + discBookCheckBox.Checked.ToString() + "')", MyConnection);

MyCommand.ExecuteNonQuery();

MyConnection.Close();

MessageBox.Show("Книга успешно добавлена!");

this.booksViewTableAdapter.Fill(this.bookShopDataSet.BooksView);

errorLabel.Text = "";

//Отчистка всех полей

nameBookTextBox.Text = "";

commentRichTextBox.Text = "";

pagesTextBox.Text = "";

yearTextBox.Text = "";

}

else

{

errorLabel.Text = "Поля должны быть заполнены!";

}

Листинг 3 Удаление данных из базы данных на примере таблицы книги (Books):

string idbook = booksViewDataGridView.CurrentRow.Cells[0].Value.ToString();

//Подключение к серверу и базе

SqlConnection MyConnection = new SqlConnection(DataBaseConection.ConnectionString);

MyConnection.Open(); //Открываем подключение

SqlCommand DeleteOrder = new SqlCommand("Delete From [Books] WHERE [idBook] = '" + idbook + "'", MyConnection);

DeleteOrder.ExecuteNonQuery();

MyConnection.Close();

booksViewBindingSource.RemoveCurrent(); //Удаление из DGV

Листинг 4 Запрос на добавление книг в таблицу книги в заказе (List), и обновляем количество книг в поставках (Supplys):

//Подключение к серверу и базе

SqlConnection MyConnection = new SqlConnection(DataBaseConection.ConnectionString);

MyConnection.Open(); //Открываем подключение

SqlDataReader SqlReader = null;

int RowsCount = Convert.ToInt32(selectedDataGridView.Rows.Count.ToString());

for (int element = 0; element < RowsCount; element++)

{

int SupplyList = Convert.ToInt32(selectedDataGridView.Rows[element].Cells[0].Value.ToString());

int BookList = Convert.ToInt32(selectedDataGridView.Rows[element].Cells[1].Value.ToString());

int QtyList = Convert.ToInt32(selectedDataGridView.Rows[element].Cells[6].Value.ToString());

//Добавляем книги в таблицу "Книги в заказе" (List)

SqlCommand InsertListCommand = new SqlCommand("Insert Into [List] (Orders, Supply, Book, listQty, listStatus) Values ('" + Users.idUserOrder + "', '" + SupplyList + "', '" + BookList + "', '" + QtyList + "', '" + 1 + "')", MyConnection);

InsertListCommand.ExecuteNonQuery();

SqlCommand UpdateOrder = new SqlCommand("Update [Supplys] Set QtyBook = QtyBook - '" + QtyList + "' Where [idSupply] = '" + SupplyList + "'", MyConnection);

UpdateOrder.ExecuteNonQuery();

}

MyConnection.Close();

Листинг 5 Считаем значение полной стоимости заказа:

int RowsCount = Convert.ToInt32(selectedDataGridView.Rows.Count.ToString()); //кол-во строк в DataGridView

for (int element = 0; element < RowsCount; element++)

{

float CellsQtyBookOrder = Convert.ToInt32(selectedDataGridView.Rows[element].Cells[6].Value); //Кол-во книг

float CellsPrice = Convert.ToInt32(selectedDataGridView.Rows[element].Cells[5].Value); //Цена за 1 книгу

selectedDataGridView.Rows[element].Cells[10].Value = CellsQtyBookOrder \* CellsPrice;

}

Листинг 6 Вывод информации выбранной книги из DataGridView в текстовые поля для формы (ViewBookForm):

//Подсвечиваем выбранную строку

takeBookViewDataGridView.DefaultCellStyle.SelectionBackColor = Color.DeepSkyBlue;

//Вывод информации в текстовые поля при выделении ячейки в строке

nameBookTextBox.Text = takeBookViewDataGridView.CurrentRow.Cells[2].Value.ToString(); //название книги

authorTextBox.Text = takeBookViewDataGridView.CurrentRow.Cells[3].Value.ToString(); //автор

companyTextBox.Text = takeBookViewDataGridView.CurrentRow.Cells[5].Value.ToString(); //издательство

pagesTextBox.Text = takeBookViewDataGridView.CurrentRow.Cells[9].Value.ToString(); //кол-во страниц

yearTextBox.Text = takeBookViewDataGridView.CurrentRow.Cells[10].Value.ToString(); //год издания

if (takeBookViewDataGridView.CurrentRow.Cells[11].Value.ToString() == "False") //наличие диска в комплекте

{

discTextBox.Text = "Нет в комплекте";

}

else

{

discTextBox.Text = "Есть в комплекте";

}

commentRichTextBox.Text = takeBookViewDataGridView.CurrentRow.Cells[4].Value.ToString();//краткое описание

Листинг 7 Запрос на изменение данных на примере таблицы книги (Books):

//Подсвечиваем выбранную строку

takeBookViewDataGridView.DefaultCellStyle.SelectionBackColor = Color.DeepSkyBlue;

//Вывод информации в текстовые поля при выделении ячейки в строке

nameBookTextBox.Text = takeBookViewDataGridView.CurrentRow.Cells[2].Value.ToString(); //название книги

authorTextBox.Text = takeBookViewDataGridView.CurrentRow.Cells[3].Value.ToString(); //автор

companyTextBox.Text = takeBookViewDataGridView.CurrentRow.Cells[5].Value.ToString(); //издательство

pagesTextBox.Text = takeBookViewDataGridView.CurrentRow.Cells[9].Value.ToString(); //кол-во страниц

yearTextBox.Text = takeBookViewDataGridView.CurrentRow.Cells[10].Value.ToString(); //год издания

if (takeBookViewDataGridView.CurrentRow.Cells[11].Value.ToString() == "False") //наличие диска в комплекте

{

discTextBox.Text = "Нет в комплекте";

}

else

{

discTextBox.Text = "Есть в комплекте";

}

commentRichTextBox.Text = takeBookViewDataGridView.CurrentRow.Cells[4].Value.ToString();//краткое описание

Листинг 7 Код таблицы клиенты (Clients) как пример выполнения главной таблицы:

public partial class ClientsForm : Form

{

public ClientsForm()

{

InitializeComponent();

}

private void clientsBindingNavigatorSaveItem\_Click(object sender, EventArgs e)

{

this.Validate();

this.clientsBindingSource.EndEdit();

this.tableAdapterManager.UpdateAll(this.bookShopDataSet);

}

private void ClientsForm\_Load(object sender, EventArgs e)

{

// TODO: данная строка кода позволяет загрузить данные в таблицу "bookShopDataSet.Clients". При необходимости она может быть перемещена или удалена.

this.clientsTableAdapter.Fill(this.bookShopDataSet.Clients);

errorLabel.Text = "";

}

private void nextButton\_Click(object sender, EventArgs e)

{

this.clientsBindingSource.MoveNext();

}

private void previousButton\_Click(object sender, EventArgs e)

{

this.clientsBindingSource.MovePrevious();

}

private void lastButton\_Click(object sender, EventArgs e)

{

this.clientsBindingSource.MoveLast();

}

private void firstButton\_Click(object sender, EventArgs e)

{

this.clientsBindingSource.MoveFirst();

}

private void saveButton\_Click(object sender, EventArgs e)

{

this.Validate();

this.clientsBindingSource.EndEdit();

this.tableAdapterManager.UpdateAll(this.bookShopDataSet);

}

private void deleteButton\_Click(object sender, EventArgs e)

{

clientsBindingSource.RemoveCurrent();

}

private void updateButton\_Click(object sender, EventArgs e)

{

//Проверка полей на пустые значения или пробелы

if (!string.IsNullOrEmpty(updateSurnameTextBox.Text) && (!string.IsNullOrWhiteSpace(updateSurnameTextBox.Text)) &&

!string.IsNullOrEmpty(updateNameTextBox.Text) && (!string.IsNullOrWhiteSpace(updateNameTextBox.Text)) &&

!string.IsNullOrEmpty(updateLastnameTextBox.Text) && (!string.IsNullOrWhiteSpace(updateLastnameTextBox.Text)) &&

!string.IsNullOrEmpty(updatePhoneTextBox.Text) && (!string.IsNullOrWhiteSpace(updatePhoneTextBox.Text)) &&

!string.IsNullOrEmpty(updateLoginTextBox.Text) && (!string.IsNullOrWhiteSpace(updateLoginTextBox.Text)) &&

!string.IsNullOrEmpty(updatePasswordTextBox.Text) && (!string.IsNullOrWhiteSpace(updatePasswordTextBox.Text))

)

{

//Подключение к серверу и базе

SqlConnection MyConnection = new SqlConnection(DataBaseConection.ConnectionString);

//Открываем подключение

MyConnection.Open();

SqlCommand MyCommand = new SqlCommand("Insert Into [Clients] (Surname, Name, Lastname, Phone, Login, Password) Values " +

"('" + updateSurnameTextBox.Text + "', '" + updateNameTextBox.Text + "', '" + updateLastnameTextBox.Text + "', '" + updatePhoneTextBox.Text + "', '" +

updateLoginTextBox.Text + "', '" + updatePasswordTextBox.Text + "')", MyConnection);

//MyCommand.Parameters.AddWithValue("nameAuthor", updateTextBox.Text);

MyCommand.ExecuteNonQuery();

}

else

{

errorLabel.Text = "Поля должны быть запонены!";

}

}

private void refreshButton\_Click(object sender, EventArgs e)

{

this.clientsTableAdapter.Fill(this.bookShopDataSet.Clients);

}

}