**Оглавление**

[Аннотация 4](#_Toc8739487)

[Введение 5](#_Toc8739488)

[1 ОБЩИЕ СВЕДЕНИЯ О ПРОГРАММНОМ СРЕДСТВЕ 6](#_Toc8739489)

[1.1 Основное функциональное назначение программного средства 6](#_Toc8739490)

[1.2 Полное наименование программного средства 6](#_Toc8739491)

[1.3 Условное обозначение программного средства 6](#_Toc8739492)

[1.4 Разработчики программного средства 6](#_Toc8739493)

[2 ТЕХНИЧЕСКОЕ ЗАДАНИЕ 7](#_Toc8739494)

[2.1 Основание для разработки 7](#_Toc8739495)

[2.2 Назначение разработки 7](#_Toc8739496)

[2.3 Требование к программному средству 7](#_Toc8739497)

[2.4 Требования к программной документации 8](#_Toc8739498)

[2.5 Требования к эргономике и технической эстетике 8](#_Toc8739499)

[2.6 Стадии и этапы разработки 9](#_Toc8739500)

[2.7 Порядок контроля и приемки 10](#_Toc8739501)

[3 ПОЯСНИТЕЛЬНАЯ ЗАПИСКА К ПРОГРАММНОМУ ПРОДУКТУ 11](#_Toc8739502)

[3.1 Декомпозиция поставленной задачи 11](#_Toc8739503)

[3.2 Общая архитектура программного средства 12](#_Toc8739504)

[3.3 Разработка алгоритма решения задачи 15](#_Toc8739505)

[3.4 Реализация функционального назначения программного средства 16](#_Toc8739506)

[3.5 Структурная организация данных 17](#_Toc8739507)

[3.6 Разработка интерфейса ПС 23](#_Toc8739508)

[3.7 Описание структуры выходной информации 34](#_Toc8739509)

[4 РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ 35](#_Toc8739510)

[4.1 Назначение программного средства 35](#_Toc8739511)

[4.2 Условия выполнения программного средства 35](#_Toc8739512)

[4.3 Эксплуатация программного средства 35](#_Toc8739513)

[4.4 Сообщения пользователю 48](#_Toc8739514)

[5 РАЗРАБОТКА ТЕСТОВОГО НАБОРА 49](#_Toc8739515)

[5.1 Определение верхней границы количества тестов 49](#_Toc8739516)

[5.2 Разработка тестовых вариантов 50](#_Toc8739517)

[5.3 Составление отчета о тестировании 65](#_Toc8739518)

[ЗАКЛЮЧЕНИЕ 66](#_Toc8739519)

[Оценка качества программного средства с помощью метрик 66](#_Toc8739520)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 69](#_Toc8739521)

[Приложение А Текст кода программного средства 71](#_Toc8739522)

[Приложение Б Спецификация 93](#_Toc8739523)

[Приложение B Диск с программным средством 95](#_Toc8739524)

**Аннотация**

На дипломную работу студента Булатова Г. С. на тему: Автоматизированная информационная система «Мелкооптовый книжный магазин». Дипломная работа представлена пояснительной запиской на 95 листов машинописного текста. Состоит из введения, 5 глав, заключения и списка литературных источников.

В главе 1 содержится общие сведения о программном средстве Автоматизированная информационная система «Мелкооптовый книжный магазин». В главе 2 приведено техническое задание на проектирование БД согласно ГОСТ 19.201. Глава 3 содержит пояснительную записку к программному продукту согласно ГОСТ 19.404-79. В главе 4 содержится руководство пользователя согласно ГОСТ 19.504-79. Глава 5 содержит обоснование количества необходимых тестов и тестовые пакеты для оценки работоспособности программного продукта. В главе 6 приведены результаты ручного и автоматического тестирования программного продукта и выполнена оценка качества с помощью метрик.

В работе содержится общее описание программного средства, предназначенного для объекта проектирования – Автоматизированная информационная система «Мелкооптовый книжный магазин», изложены требования к программному средству, описаны его архитектура, реализация функций, интерфейс, приведено руководство пользователя, обоснованы тестовые пакеты и выполнена оценка качества программного продукта с помощью метрик.

**Введение**

В дипломной работе была поставлена задача разработки Автоматизированная информационная система «Мелкооптовый книжный магазин», который позволяет автоматизировать деятельность мелкооптового книжного магазина по автоматизации работы с товарами.

Для достижения поставленной цели был составлен основной алгоритм работы программы, а также разработаны алгоритмы вспомогательных процедур.

Для реализации алгоритмов был выбран язык программирования C#. Выбор обусловлен тем, что возможностей этого языка достаточно для достижения поставленной цели моей дипломной работы.

При разработке интерфейса программы, я учитывал наиболее нужные и важные функциональные возможности, которые должна выполнять данная программа, а также тип пользователей, которые с ней будут работать, в данном случае менеджер мелкооптового книжного магазина.

**1 ОБЩИЕ СВЕДЕНИЯ О ПРОГРАММНОМ СРЕДСТВЕ**

**1.1 Основное функциональное назначение программного средства**

Программа предназначена для автоматизации деятельности мелкооптового книжного магазина. Программа не требует владения современными информационными технологиями для эффективного использования данного программного средства, т.е. подходит для любого пользователя.

**1.2 Полное наименование программного средства**

Автоматизированная информационная система «Мелкооптовый книжный магазин»

**1.3 Условное обозначение программного средства**

«BookStoreApp»

**1.4 Разработчики программного средства**

Программное средство разработал студент Зеленодольского Механического колледжа группы 197 Булатов Г. С.

**2 ТЕХНИЧЕСКОЕ ЗАДАНИЕ**

**2.1 Основание для разработки**

Работа ведётся на основании задания на дипломную работу по специальности 09.02.03 — «Программирование в компьютерных системах».

**2.2 Назначение разработки**

Программа "BookStoreApp" работает под управлением операционных систем Windows 7, Windows 8 и Windows 10. Объект проектирования - база данных и модуль обработки данных книжного магазина "BookStore" (подсистема работы с книгами). Подсистема для работы с книгами магазина включает в себя следующий функционал:

• просмотр списка книг;

• добавление/удаление/редактирование данных о книгах;

• просмотр истории заказов книг в магазине.

**2.3 Требование к программному средству**

«BookStoreApp» устанавливается на персональный компьютер, имеющий представленные минимальные системные требования:

Оперативная память: 4 Гб;

Процессор: Intel Core i / Phenom II X4 / AMD Athlon64 X2 4600 + 2.4 ГГц;

Видео карта: nVidia GeForce 8600 / ATI Radeon HD 2400;

Звуковая карта: Совместимая с DirectX;

Программный продукт «BookStoreApp» является кроссплатформенным продуктом, что позволяет не ограничиваться в работе с определенной версией ОС.

«BookStoreApp» написана на объектно-ориентированном языке программировании C# (си-шарп) в интегрированной среде разработки Visual Studio 2019. В качестве СУБД используется MS SQL. Программа не может подлежать сторонней оптимизации и редактированию программного кода.

**2.4 Требования к программной документации**

Состав программной документации должен включать в себя:

1. техническое задание;

2. сроки выполнения работ;

3. программу и методику испытаний;

4. эксплуатационные инструкции пользователю;

**2.5 Требования к эргономике и технической эстетике**

Система обеспечивает удобный для пользователей системный интерфейс, отвечающий следующим требованиям:

- единый стиль оформления для пользовательских интерфейсов;

- удобная, интуитивно понятная навигация в интерфейсе пользователя;

- взаимодействие пользователя с системой осуществляется на русском языке

Пользовательские интерфейсы системы спроектированы и разработаны с применением единых принципов графического представления информации и организации доступа к функциональным возможностям и сервисам. Разработан графический дизайн пользовательских интерфейсов, цветовые, шрифтовые и композиционные решения для отображения текстов, изображений, таблиц, гиперссылок, управляющих и навигационных элементов (меню, кнопок, форм и т.п.).

Экранные формы спроектированы с учетом требований унификации:

- все экранные формы пользовательского интерфейса выполнены в едином графическом дизайне, с одинаковым расположением основных элементов управления и навигации;

- для обозначения сходных операций использованы сходные графические значки, кнопки и другие управляющие (навигационные) элементы. Термины, используемые для обозначения типовых операций (добавление информационной сущности, редактирование поля данных), а также последовательности действий пользователя при их выполнении унифицированы;

- внешнее поведение сходных элементов интерфейса (реакция на наведение указателя «мыши», переключение фокуса, нажатие кнопки) реализованы одинаково для однотипных элементов.

**2.6 Стадии и этапы разработки**

Разработка должна быть проведена в три стадии:

1. разработка технического задания;

2. рабочее проектирование;

3. внедрение.

На стадии разработки технического задания должен быть выполнен этап разработки, согласования и утверждения настоящего технического задания.

На стадии рабочего проектирования должны быть выполнены перечисленные ниже этапы работ:

1. разработка программы;

2. разработка программной документации;

3. испытания программы.

На этапе разработки технического задания должны быть выполнены перечисленные ниже работы:

1. постановка задачи;

2. определение и уточнение требований к техническим средствам;

3. определение требований к программе;

4. определение стадий, этапов и сроков разработки программы и документации на нее;

5. выбор языков программирования;

6. согласование и утверждение технического задания.

На этапе разработки программы должна быть выполнена работа по программированию (кодированию) и отладке программы.

На этапе разработки программной документации должна быть выполнена разработка программных документов в соответствии с пунктом Предварительный состав программной документации настоящего технического задания.

На этапе испытаний программы должны быть выполнены перечисленные ниже виды работ:

1. разработка, согласование и утверждение порядка и методики испытаний;

2. проведение испытаний;

3. корректировка программы и программной документации по результатам испытаний.

**2.7 Порядок контроля и приемки**

Сдача-приёмка работ производится поэтапно, в соответствии с рабочей программой и календарным планом. Сдача-приемка осуществляется комиссией, в состав которой входят представители ГАПОУ и работодатель. По результатам приемки подписывается акт приемочной комиссии.

В указанном подразделе, согласно таблице «Сроки выполнения работ» технического задания, будет описаны перечень участвующих организаций, место и сроки проведения работ, согласно п. 2.8 ГОСТ 34.602-89.

Порядок согласования и утверждения приемочной документации должен регламентироваться организационно-распорядительной документацией организации, принимающей участие в создании программного продукта. Согласно разделу «Приемка результатов разработки» ГОСТ 15.001-88 для согласования и утверждения приемочной документации создается приемочная комиссия (приказом).

Статус приемочной комиссии определяется Заказчиком до проведения испытаний.

**3 ПОЯСНИТЕЛЬНАЯ ЗАПИСКА К ПРОГРАММНОМУ ПРОДУКТУ**

**3.1 Декомпозиция поставленной задачи**

Для достижения цели работы была поставлена следующая задача:

Создать модуль «CodeDatabase» отвечающий за сохранение БД. Результат работы модуля - сохранение БД.

Поставленные задачи наглядно отображаются на диаграмме деятельности.

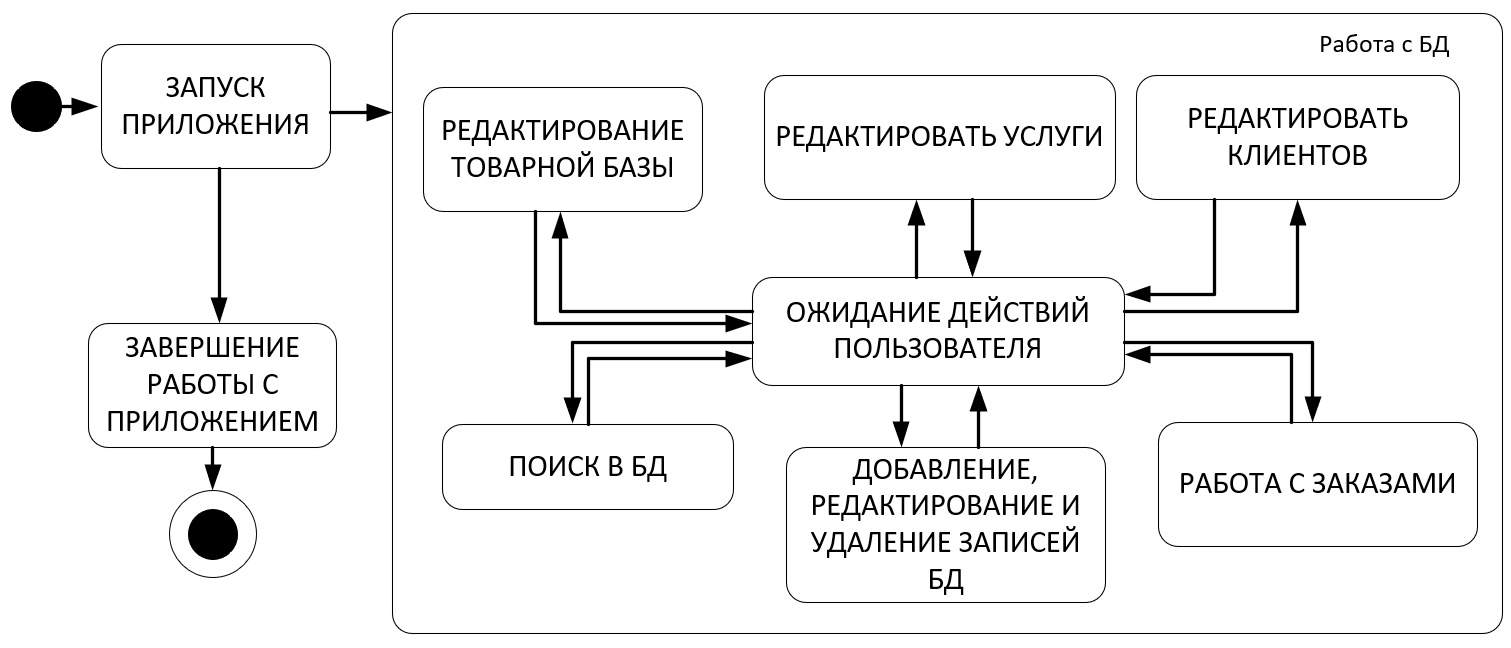


Рис. 1 Декомпозиция поставленной задачи

**3.2 Общая архитектура программного средства**

Объектом моделирования является деятельность мелкооптового книжного магазина.

Рассматриваемые процессы: движение товара.

Объекты моделирования представлены на диаграмме классов, рис. 2.

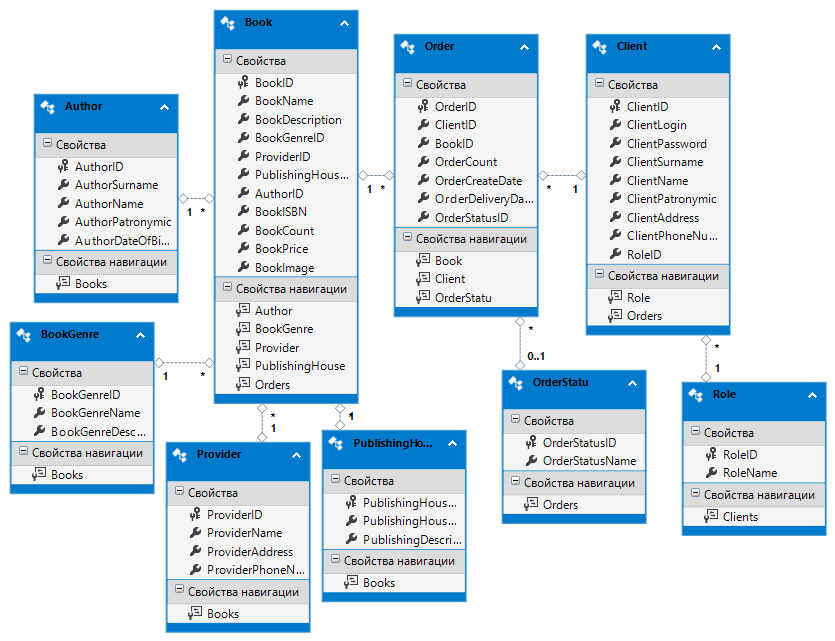


Рис. 2 Диаграмма классов

По полученной декомпозиции задач была спроектирована следующая архитектура программного средства.



Рис. 3 Схема событийно-ориентированной архитектуры

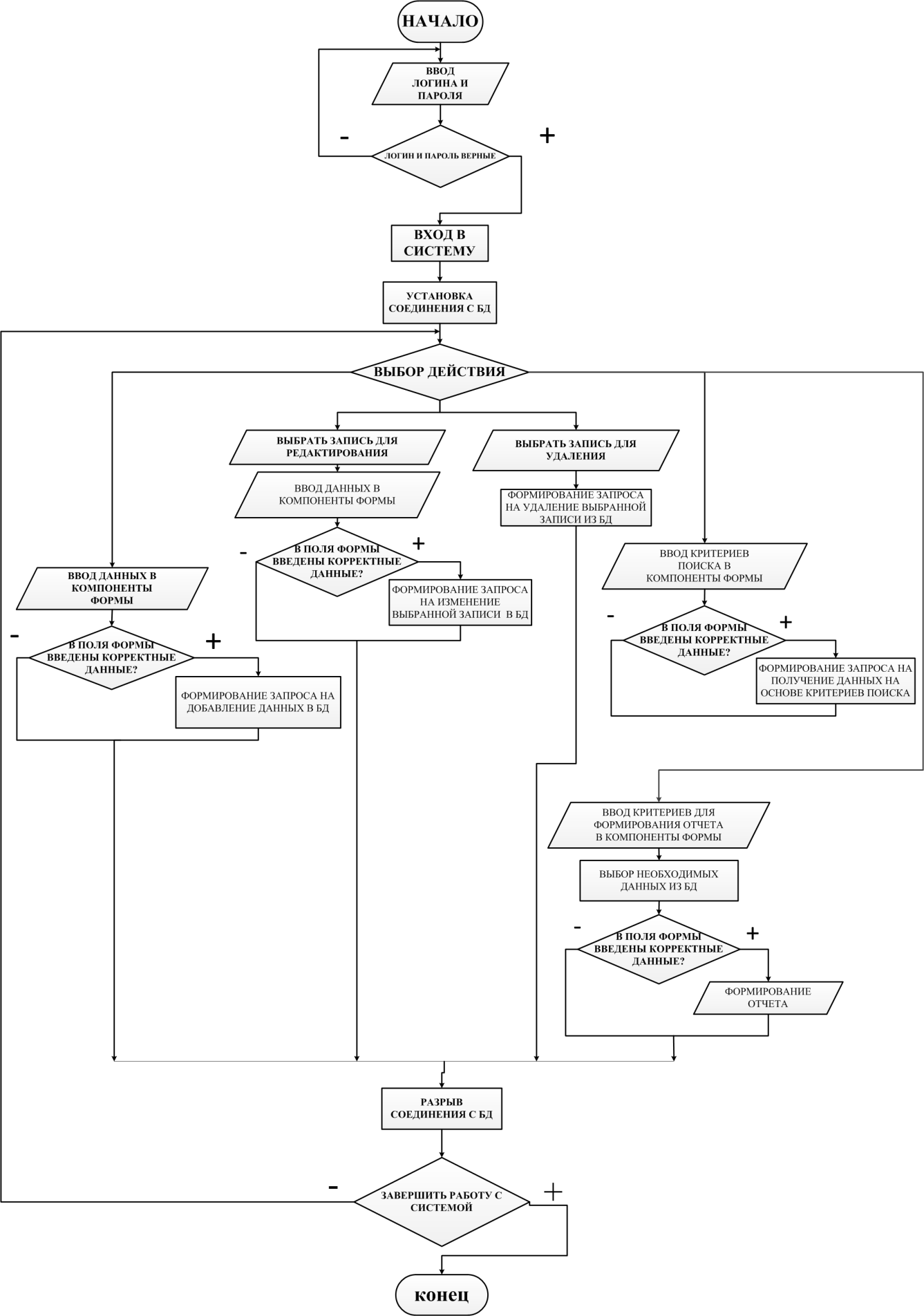
В «BookStoreApp» используется событийно-ориентированная архитектура (рис. 3). Роль агента (источник событий) в программе выполняют: пользователи программы, за роль стоков (потребители событий) отвечают таблицы, входящие в базу данных. Например, когда пользователь выбирает определенное действие: добавить, удалить, редактировать, вывести в Excel и т.д. система осуществит выбранные действия, и база отреагирует соответствующим образом: запись добавлена, удалена, отредактирована, либо отчет выведен в Excel. Системная архитектура пользователя рассматривает это изменение состояния как событие, создаваемое, публикуемое, определяемое и потребляемое различными приложениями в составе архитектуры.



Рис. 4 Диаграмма вариантов использования

**3.3 Разработка алгоритма решения задачи**

**Общий алгоритм**

****Рис. 5 Алгоритм программы

**3.4 Реализация функционального назначения программного средства**

Программа «BookStoreApp» имеет следующий набор входных данных, такие как: авторы книг, книги, жанры книг, клиенты, статусы заказов, поставщики, издательства.

Данные вводятся администратором в соответствующие поля ввода, снабженные всплывающими подсказками.

Выходными данными являются: каталог книг, информация о движении книг.

Данные вводятся пользователем в соответствующие поля вывода, снабженные всплывающими подсказками.

Выходные данные редактировать вручную невозможно, т.к. они служат в качестве отчетной информации для пользователя.

**3.5 Структурная организация данных**

Для создания БД необходимо определиться с данными, которые необходимы для полноценного функционирования системы. Все эти данные указаны в реляционной модели Информационная система «Организация проверки ассортимента магазина» представленная на рисунке. Любая реляционная база данных и называется реляционной, что характеризуется отношениями (relation) между таблицами. На рисунке изображены таблицы моей базы данных. При этом одна таблица является родительской (главной), а вторая – дочерней (подчиненной). Главной таблицей является «Книга». Реляционная модель автоматизированной системы соответствует всем 12 правилам Кодда. (рис. 6)

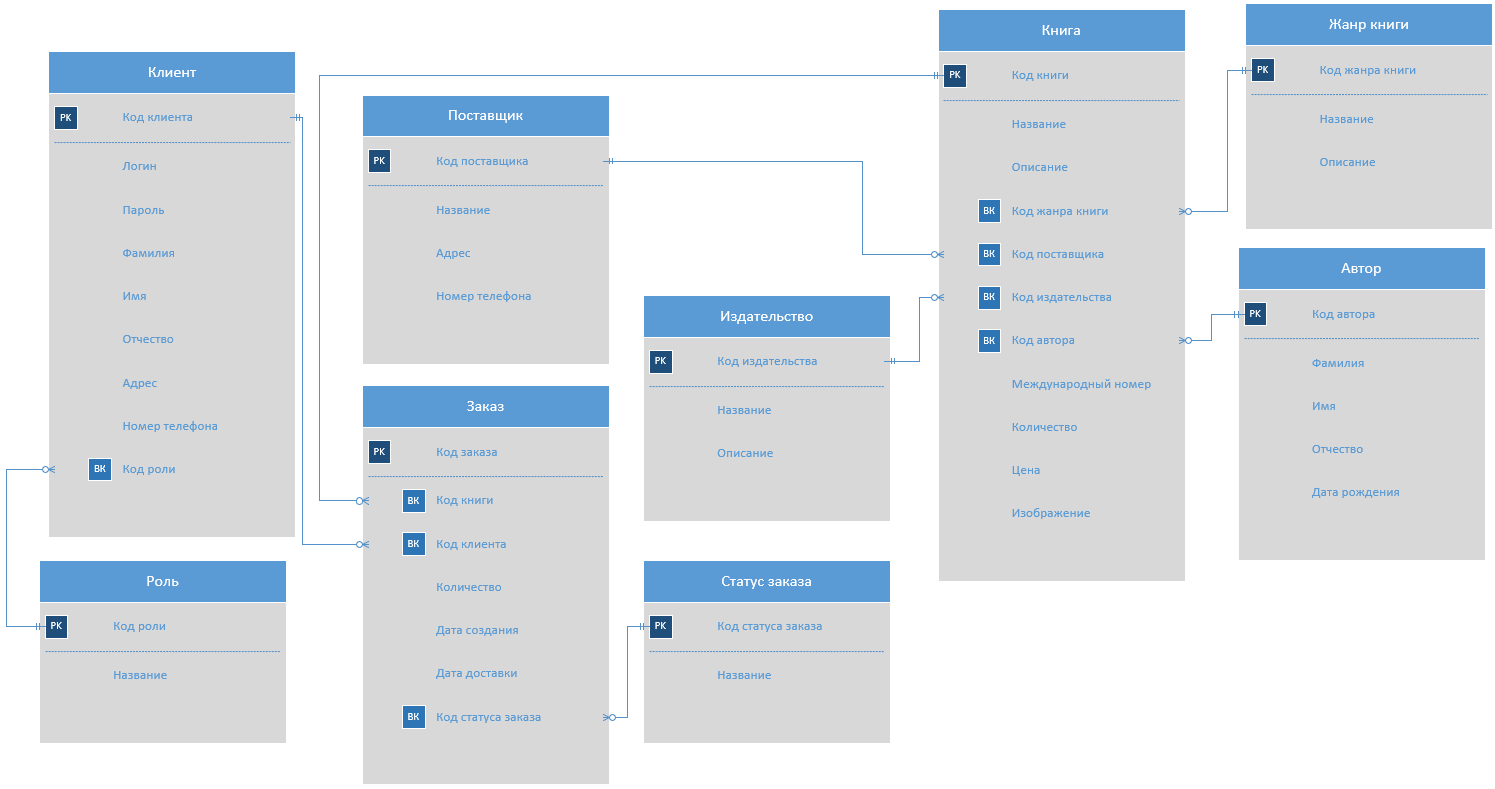


Рис. 6 Реляционная модель

Первичный ключ в базе уникален, используется для организации отношений между таблицами, который не может иметь пустых и повторяющихся значений. Первичными ключами в базе являются поля: Код клиента (таблица «Клиент»), Код роли (таблица «Роль»), Код поставщика (таблица «Поставщик»), Код заказа (таблица «Заказ»), Код издательства (таблица «Издательство»), Код статуса заказа (таблица «Статус заказа»), Код книги (таблица «Книга»), Код жанра книги (таблица «Жанр книги»), Код автора (таблица «Автор»). Остальные ID-ключи являются внешними ключами.

Для организации более эффективной обработки данных применяется нормализация. Таблицы моей БД находятся в 3НФ:

* БД находится в форме -1НФ потому, что
  + Таблица не имеет повторяющихся записей;
  + Каждый атрибут отношения хранит одно-единственное значение и не является списком, ни множеством значений;
  + Таблица не имеет повторяющихся групп полей.
    - Вторая нормальная форма (2НФ):
      * Устранены атрибуты, зависящие только от части уникального (первичного) идентификатора, т.е. ID.
        + Третья нормальная форма (3НФ):

Отсутствуют атрибуты, зависящие от атрибутов, не входящих в уникальный (первичный) ключ.

На основе реляционной модели базы данных мною в MS SQL была построена следующая база данных (рис. 7). В ней первичным ключом является поле таблицы «Book» – «BookID». Связь таблицы «BookGenre» и «Book» по полю «BookGenreID» один ко многим (1:М). Связь таблицы «Provider» и «Book» по полю «ProviderID» один ко многим (1:М). Связь таблицы «PublishingHouse» и «Book» по полю «PublishingHouseID» один ко многим (1:М). Связь таблицы «Author» и «Book» по полю «AuthorID» один ко многим (1:М). Связь таблицы «Role» и «Client» по полю «RoleID» один ко многим (1:М). Связь таблицы «Client» и «Order» по полю «ClientID» один ко многим (1:М). Связь таблицы «OrderStatus» и «Order» по полю «OrderStatusID» один ко многим (1:М). Связь таблицы «Book» и «Order» по полю «BookID» один ко многим (1:М). Все связи между таблицами базы данных типа один ко многим.



Рис. 7 Схема базы данных

Таблица «Author» (Автор)

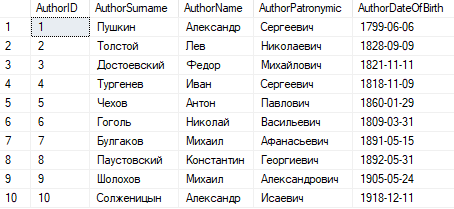


Рис. 8 Таблица Author

Таблица «Book» (Книга)

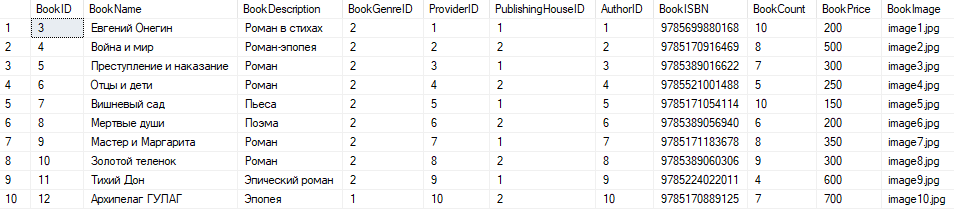


Рис. 9 Таблица Book

Таблица «BookGenre» (Жанр книги)

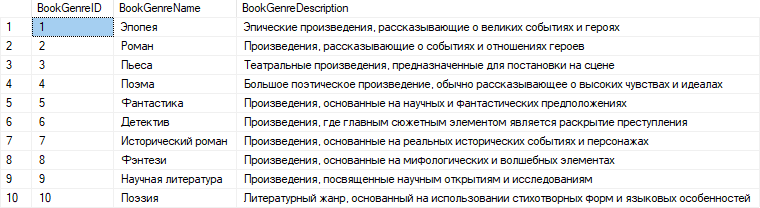


Рис. 10 Таблица BookGenre

Таблица «Client» (Клиент)



Рис. 11 Таблица Client

Таблица «Order» (Заказ)

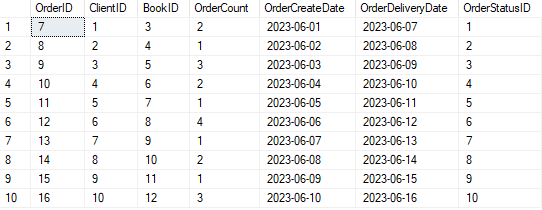


Рис. 12 Таблица Order

Таблица «OrderStatus» (Статус заказа)

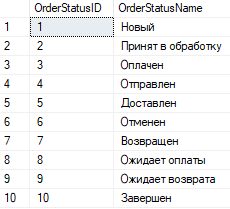


Рис. 13 Таблица OrderStatus

Таблица «Provider» (Поставщик)

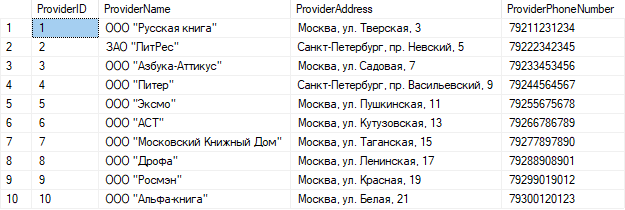


Рис. 14 Таблица Provider

Таблица «PublishingHouse» (Издательство)



Рис. 15 Таблица PublishingHouse

Таблица «Role» (Роль)



Рис. 16 Таблица Role

**3.6 Разработка интерфейса ПС**

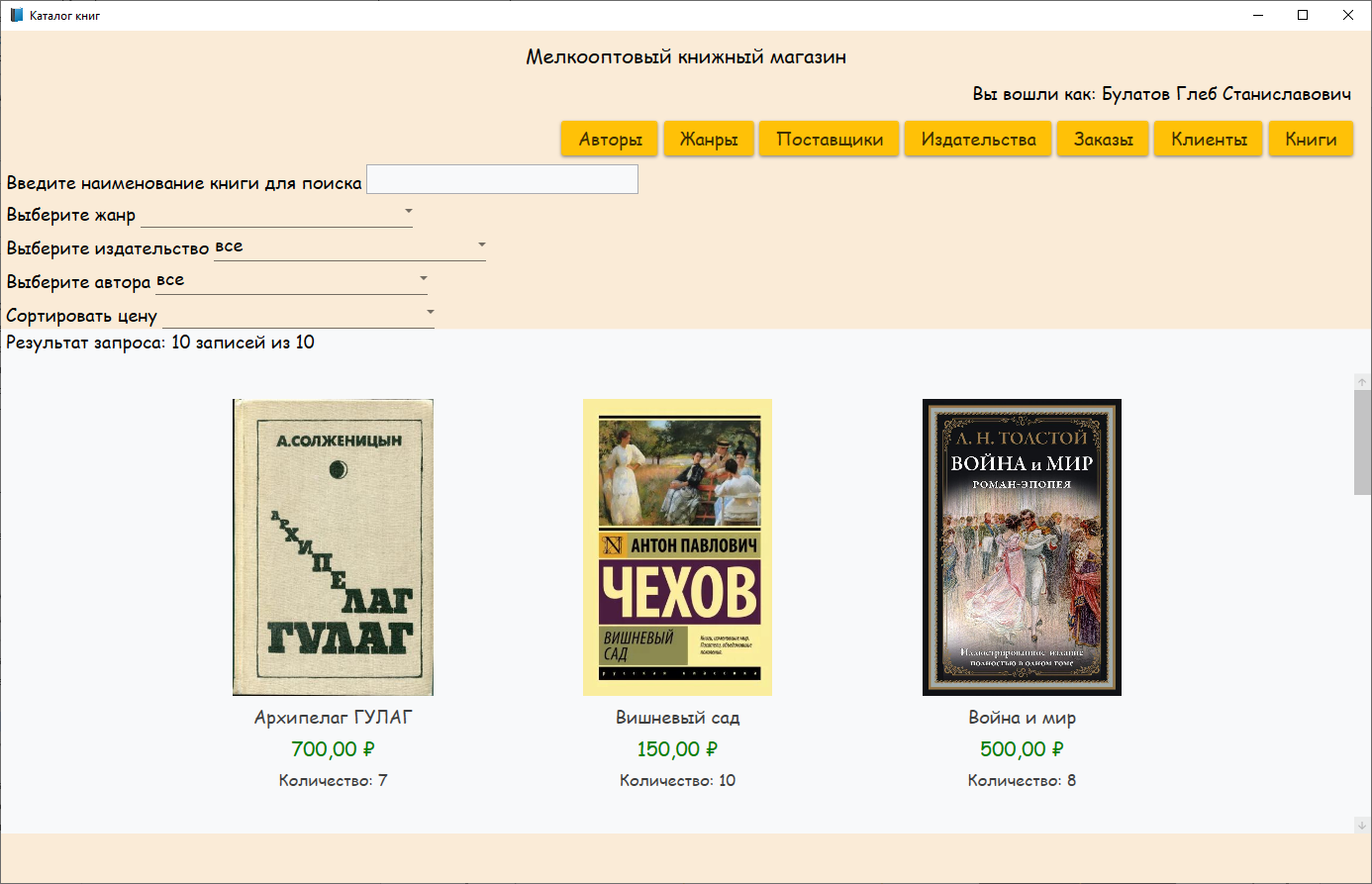
Приложение предназначено для трех типов пользователей: администратора, менеджера и клиента. Для входа в панель администратора необходимо ввести свои учётные данные. Для запуска приложения достаточно щелкнуть дважды левой кнопкой мыши по ярлыку.

После запуска программы пользователь приступит к работе с главной формой программы, на которой отбражается каталог товаров в виде плиточного интерфейса. «BookStoreApp» обладает удобным интерфейсом, так что даже неопытному в обращение с пользовательскими компьютерными программами пользователю не составит труда быстро привыкнуть к работе с данным программным средством.

На рабочем экране расположены несколько кнопок для комфортной работы с приложением.

**Форма Автоматизированная информационная система «Мелкооптовый книжный магазин»**. Основная форма приложения (рис. 17). При разработке интерфейса этой формы было использовано множество различныx компонентов:

1. ComboBox – выпадающий список;
2. Image – для отображения изображения;
3. Button – кнопка;
4. TextBox – поле для ввода.



Button

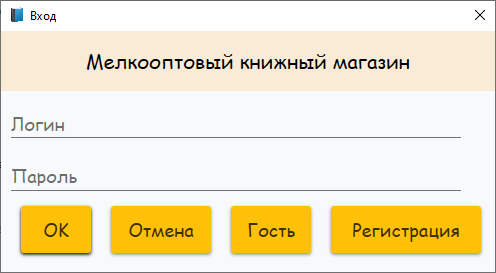
ComboBox

Image

TextBox

Рис. 17 Автоматизированная информационная система «Мелкооптовый книжный магазин»

**Форма входа в систему**. Интерфейс данной формы состоит из компонентов: Button – кнопки «ОК», «Отмена», «Гость», «Регистрация», Grid – сетка для аккуратного размещения компонентов, TextBox – поле для ввода логина, PasswordBox – поле для ввода пароля и TextBlock – текстовые метки (рис. 18).



Button

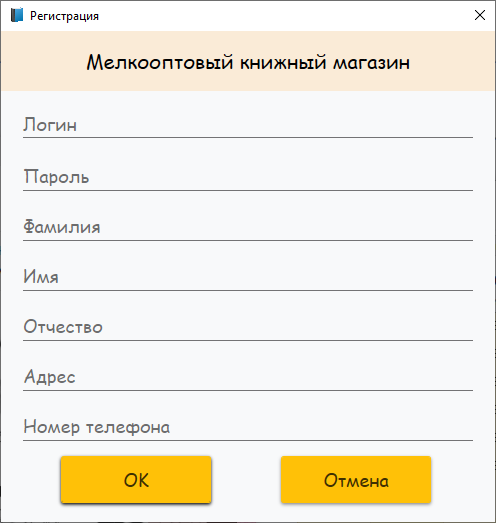
TextBox

PasswordBox

TextBlock

Рис. 18 Форма входа в систему

**Окно «Регистрация»**. Интерфейс данного окна состоит из компонентов: Button – кнопки «ОК», «Отмена»», TextBox – поле для ввода логина, PasswordBox – поле для ввода пароля и TextBlock – текстовая метка (рис. 19).



Button

TextBox

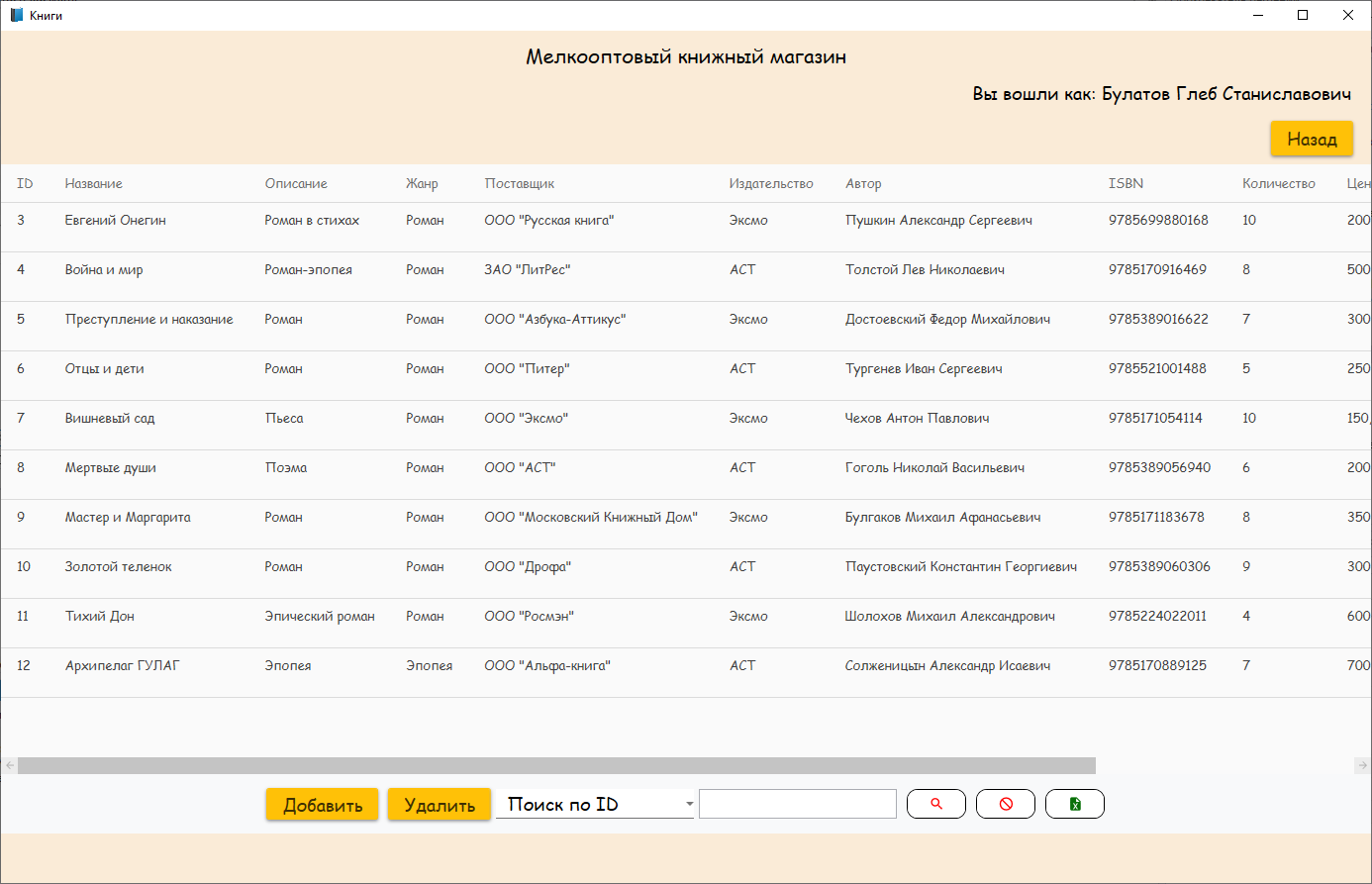
PasswordBox

TextBlock

Рис. 19 Форма входа в систему

Страница «**Книги»**. Интерфейс данной формы состоит из компонентов: DataGrid, ComboBox – выпадающий список, TextBox – поле ввода и трех кнопок: Назад, Добавить и Удалить (рис. 20).

Компонент DataGrid - таблица



Button

Button – кнопка Назад

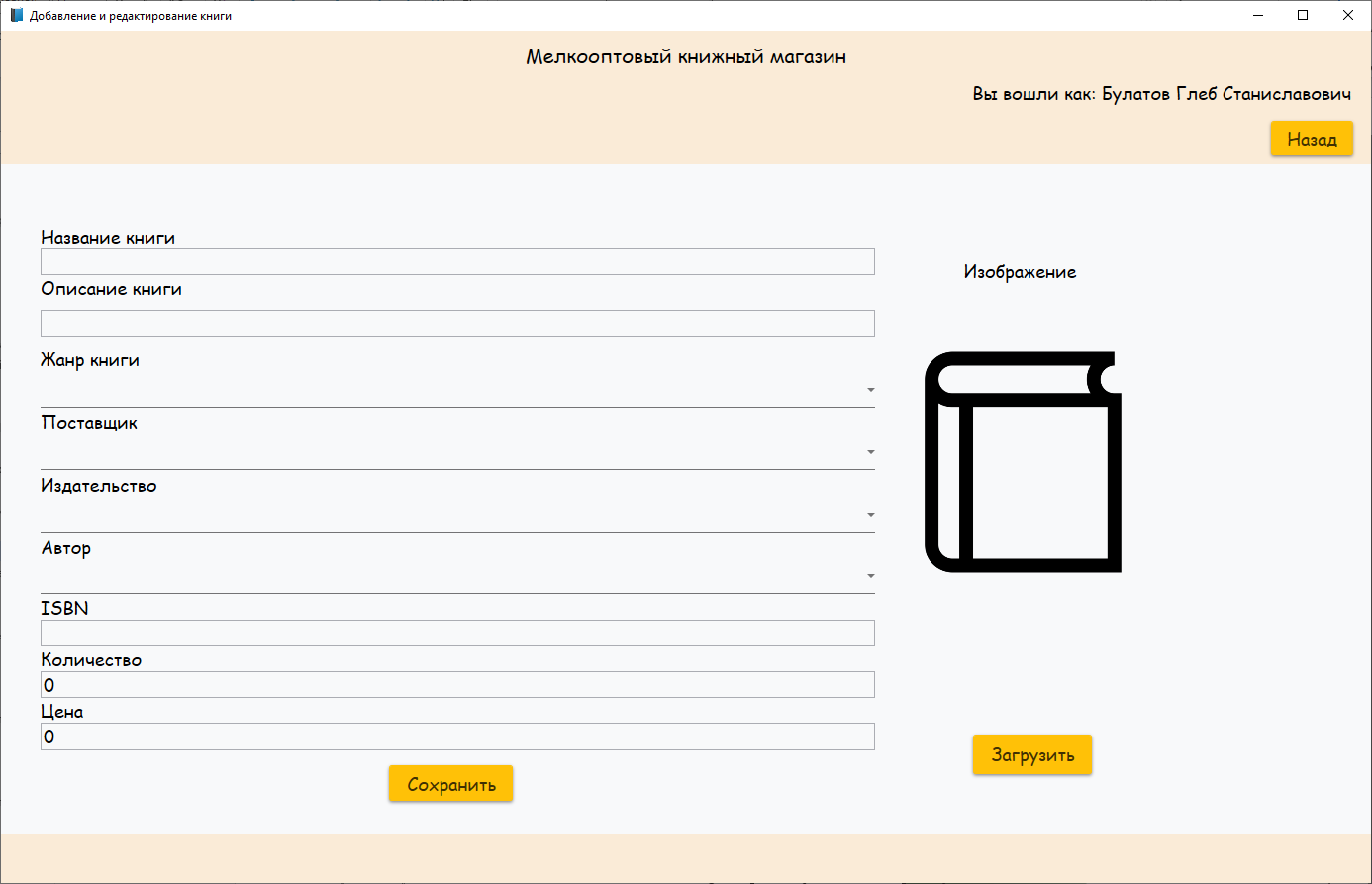
ComboBox

TextBox

Button

Рис. 20 Страница «Книги»

Страница **«Добавление и редактирование книги»**. Интерфейс данной формы состоит из компонентов: TextBox – поля ввода, ComboBox – выпадающие списки, Image – для отображения изображения и Button – кнопок (рис. 21).



TextBox

Button

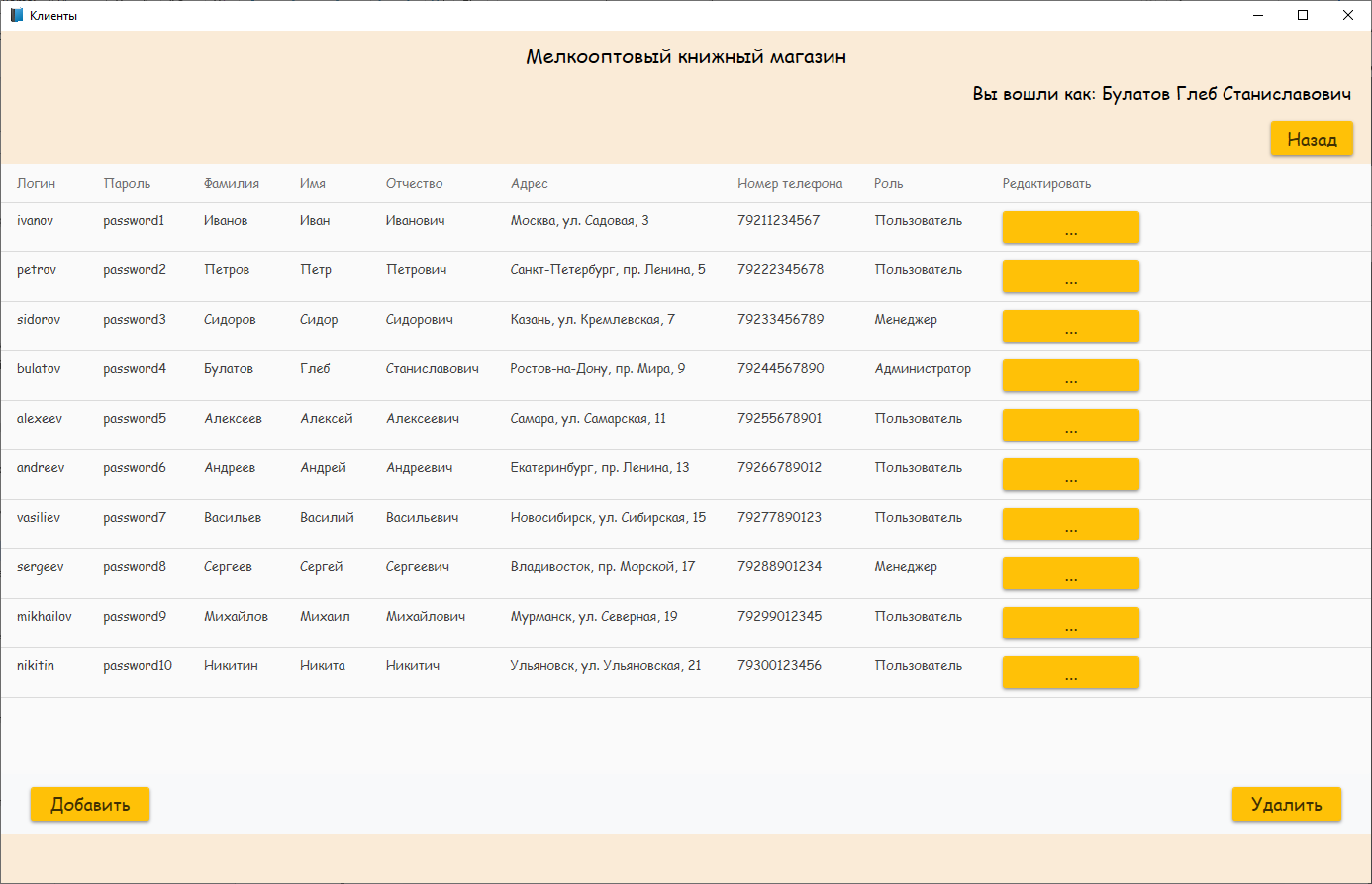
ComboBox

Image

Рис. 21 Страница «Добавление и редактирование книги»

Страница «**Клиенты»**. Интерфейс данной формы состоит из компонентов: DataGrid и трех кнопок: Назад, Добавить и Удалить (рис. 22).

Компонент DataGrid - таблица

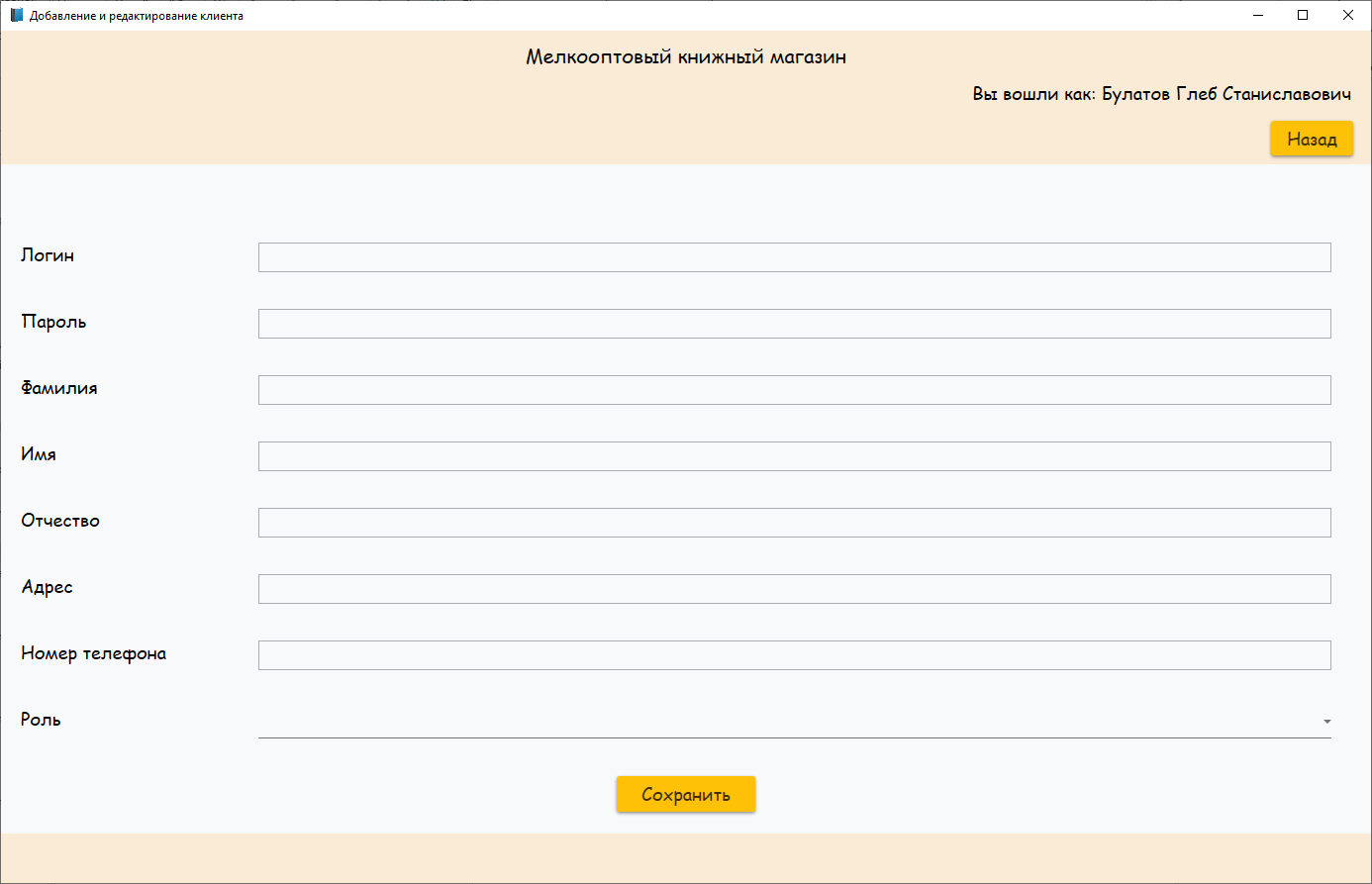


Button – кнопки Добавить и Удалить

Button – кнопка Назад

Рис. 22 Страница «Книги»

Страница **«Добавление и редактирование клиента»**. Интерфейс данной формы состоит из компонентов: TextBox – поля ввода, ComboBox – выпадающий список и Button – кнопок (рис. 23).

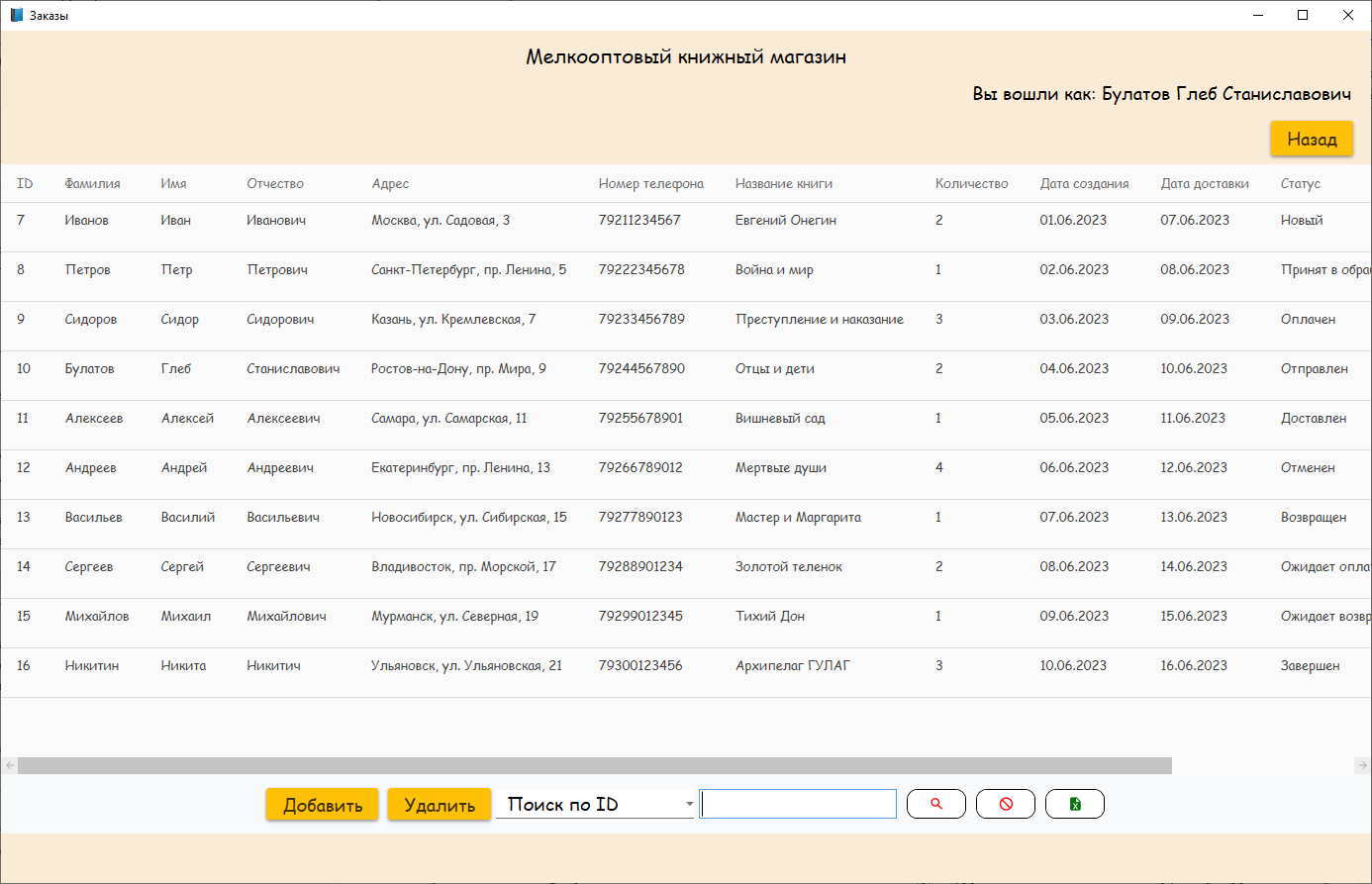


TextBox

Button

ComboBox

Рис. 23 Страница «Добавление и редактирование книги»

Страница «**Заказы»**. Интерфейс данной формы состоит из компонентов: DataGrid, ComboBox – выпадающий список, TextBox – поле ввода и трех кнопок: Назад, Добавить и Удалить (рис. 24).

Button

Button – кнопка Назад

ComboBox

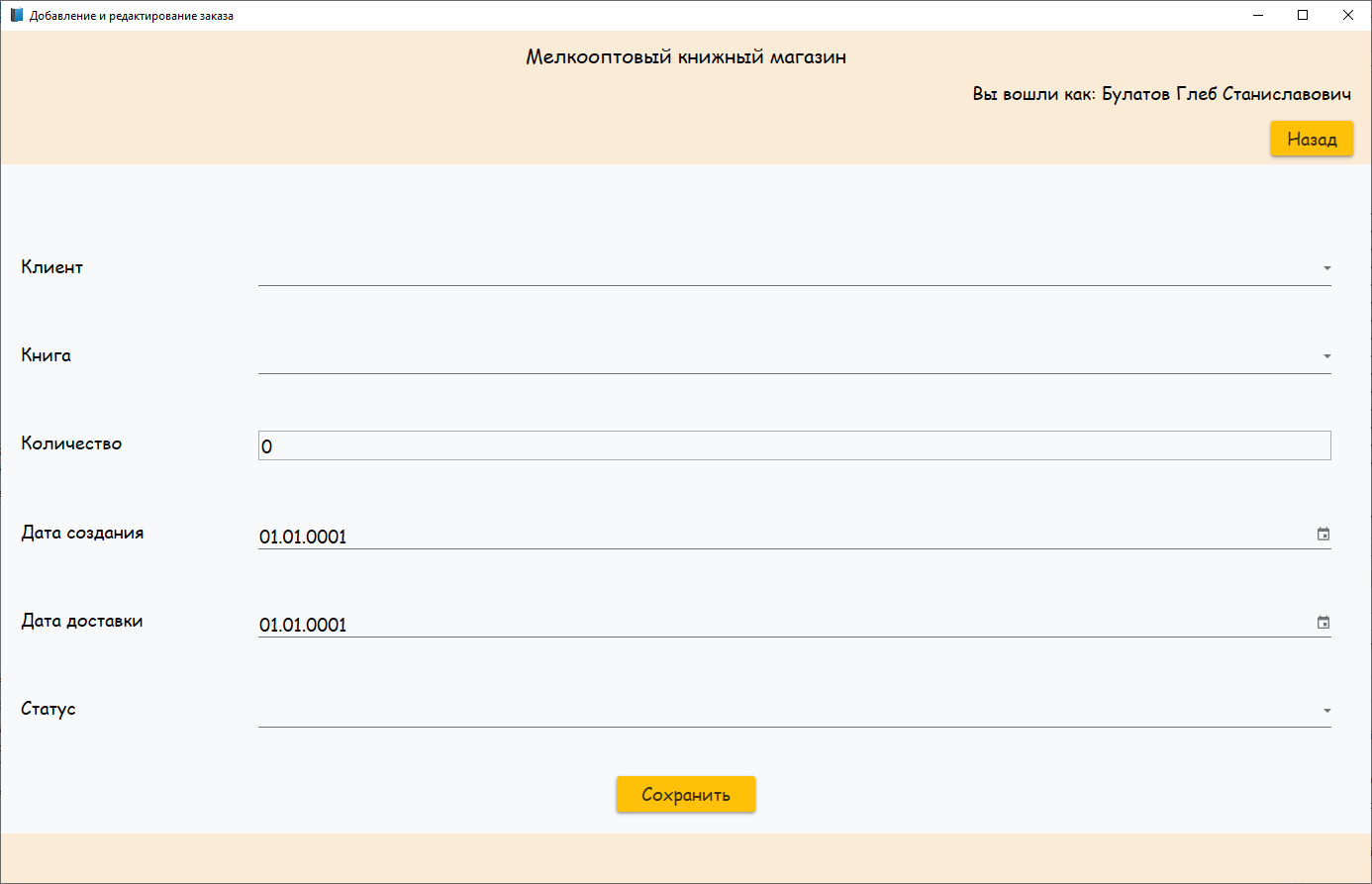
TextBox

Button

Компонент DataGrid - таблица

Рис. 24 Страница «Заказы»

Страница **«Добавление и редактирование заказа»**. Интерфейс данной формы состоит из компонентов: TextBox – поля ввода, ComboBox – выпадающий список, DatePicker – выборка даты и Button – кнопок (рис. 25).



TextBox

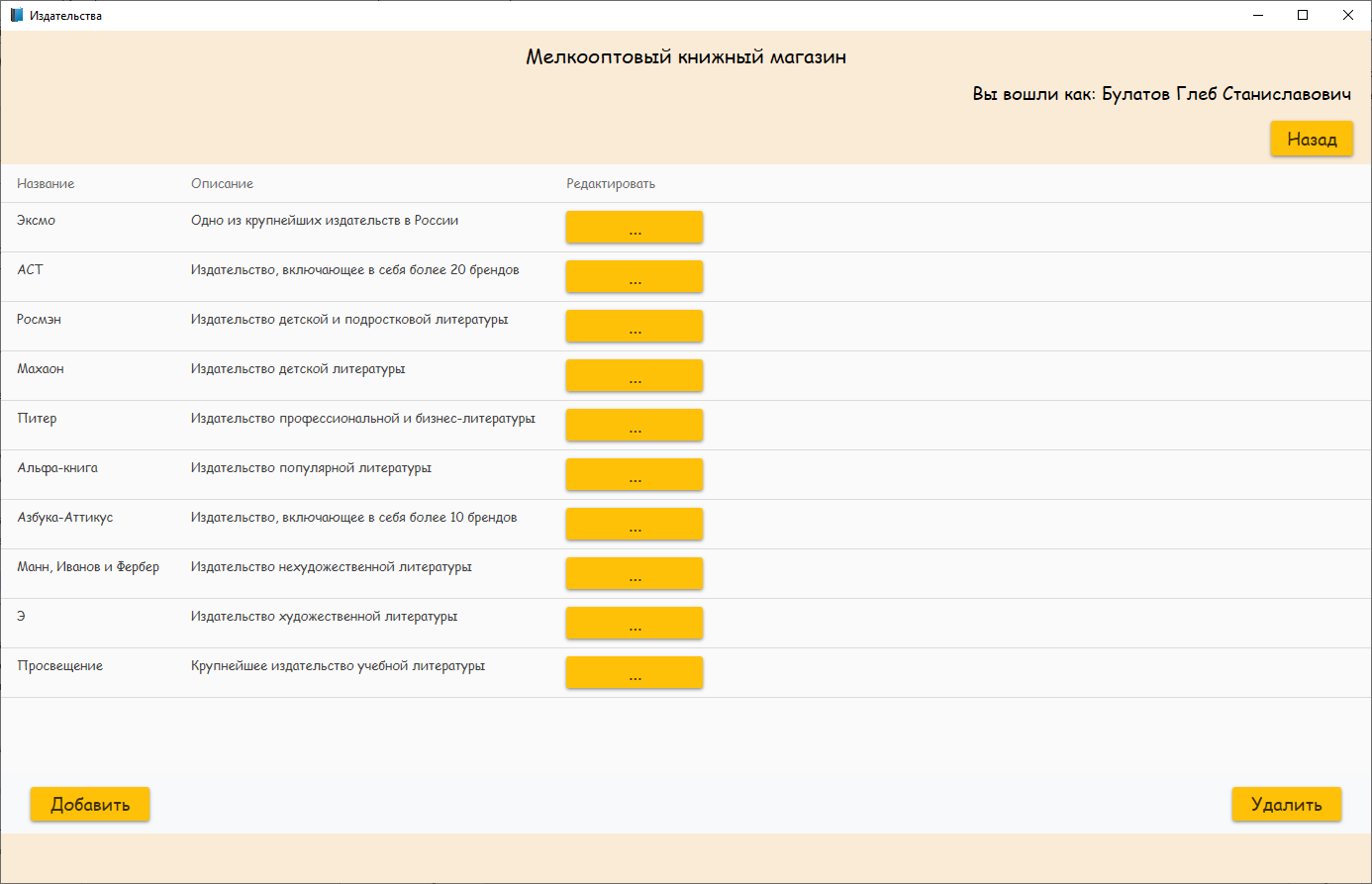
Button

ComboBox

Рис. 25 Страница «Добавление и редактирование заказа»

Страница «**Издательства»**. Интерфейс данной формы состоит из компонентов: DataGrid и трех кнопок: Назад, Добавить и Удалить (рис. 26).

Компонент DataGrid - таблица

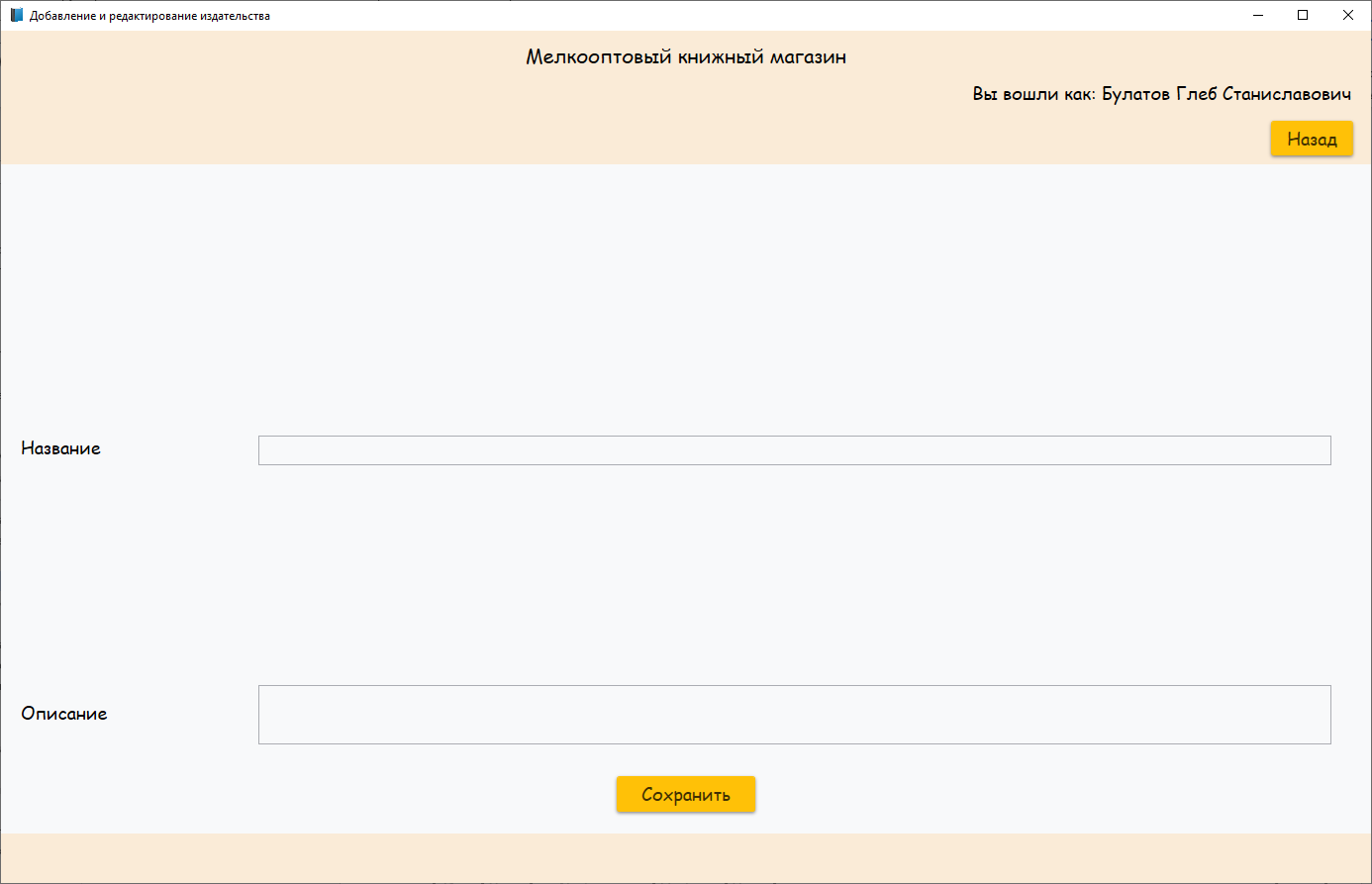


Button – кнопки Добавить и Удалить

Button – кнопка Назад

Рис. 26 Страница «Издательства»

Страница **«Добавление и редактирование издательства»**. Интерфейс данной формы состоит из компонентов: TextBox – поля ввода и Button – кнопок (рис. 27).



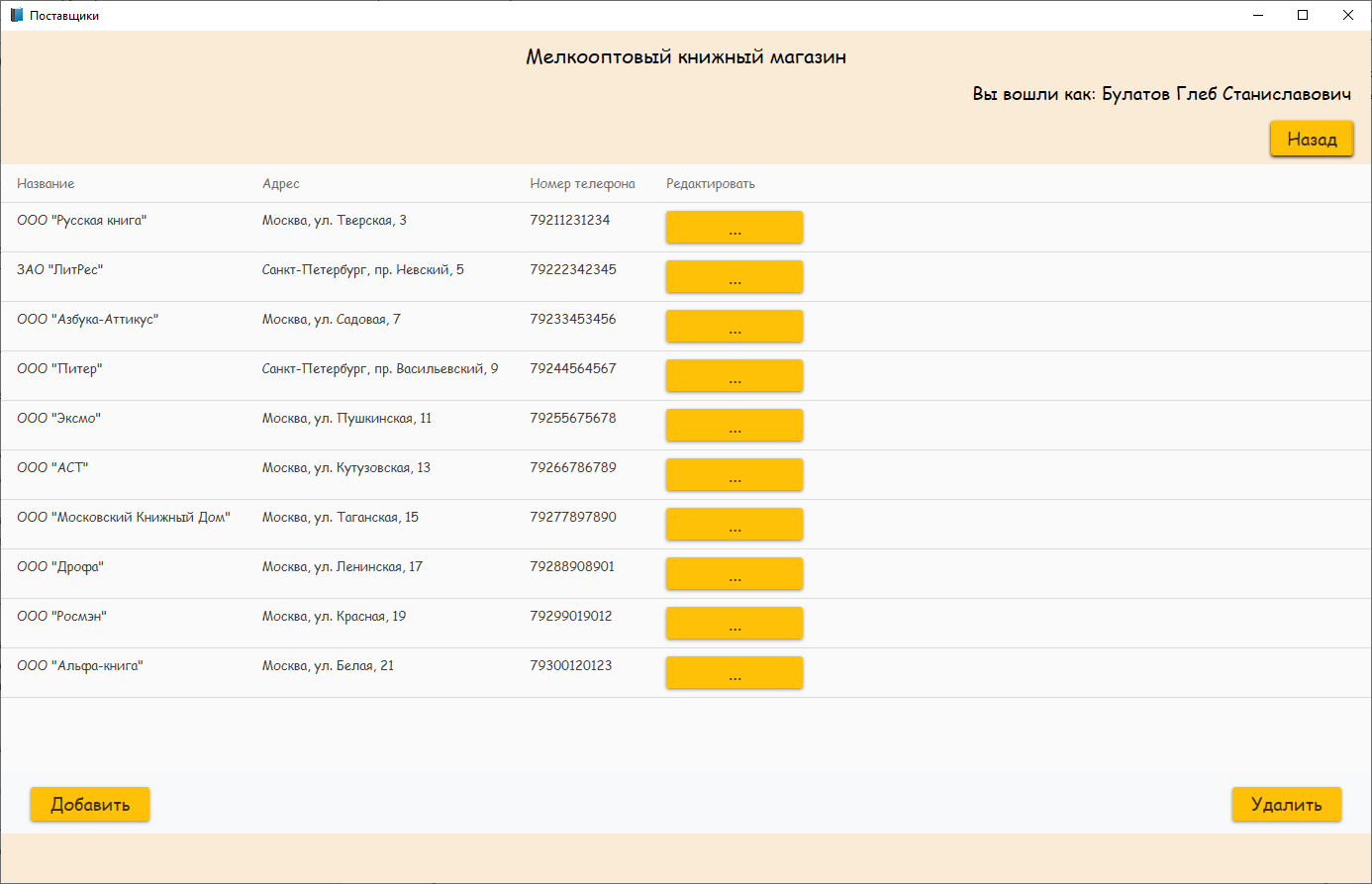
TextBox

Button

Рис. 27 Страница «Добавление и редактирование издательства»

Страница «**Поставщики»**. Интерфейс данной формы состоит из компонентов: DataGrid и трех кнопок: Назад, Добавить и Удалить (рис. 28).

Компонент DataGrid - таблица

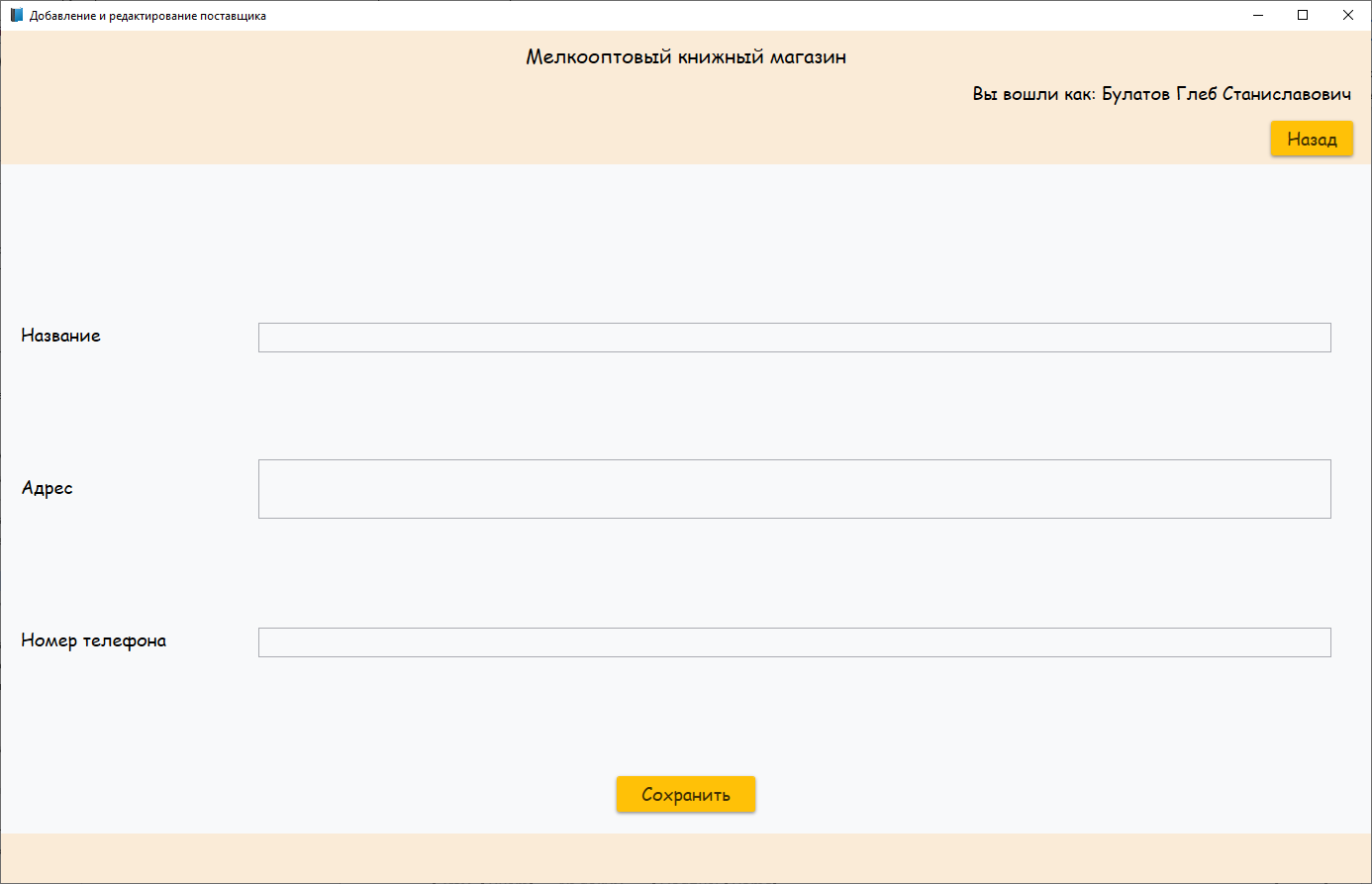


Button – кнопки Добавить и Удалить

Button – кнопка Назад

Рис. 28 Страница «Поставщики»

Страница **«Добавление и редактирование поставщика»**. Интерфейс данной формы состоит из компонентов: TextBox – поля ввода и Button – кнопок (рис. 29).



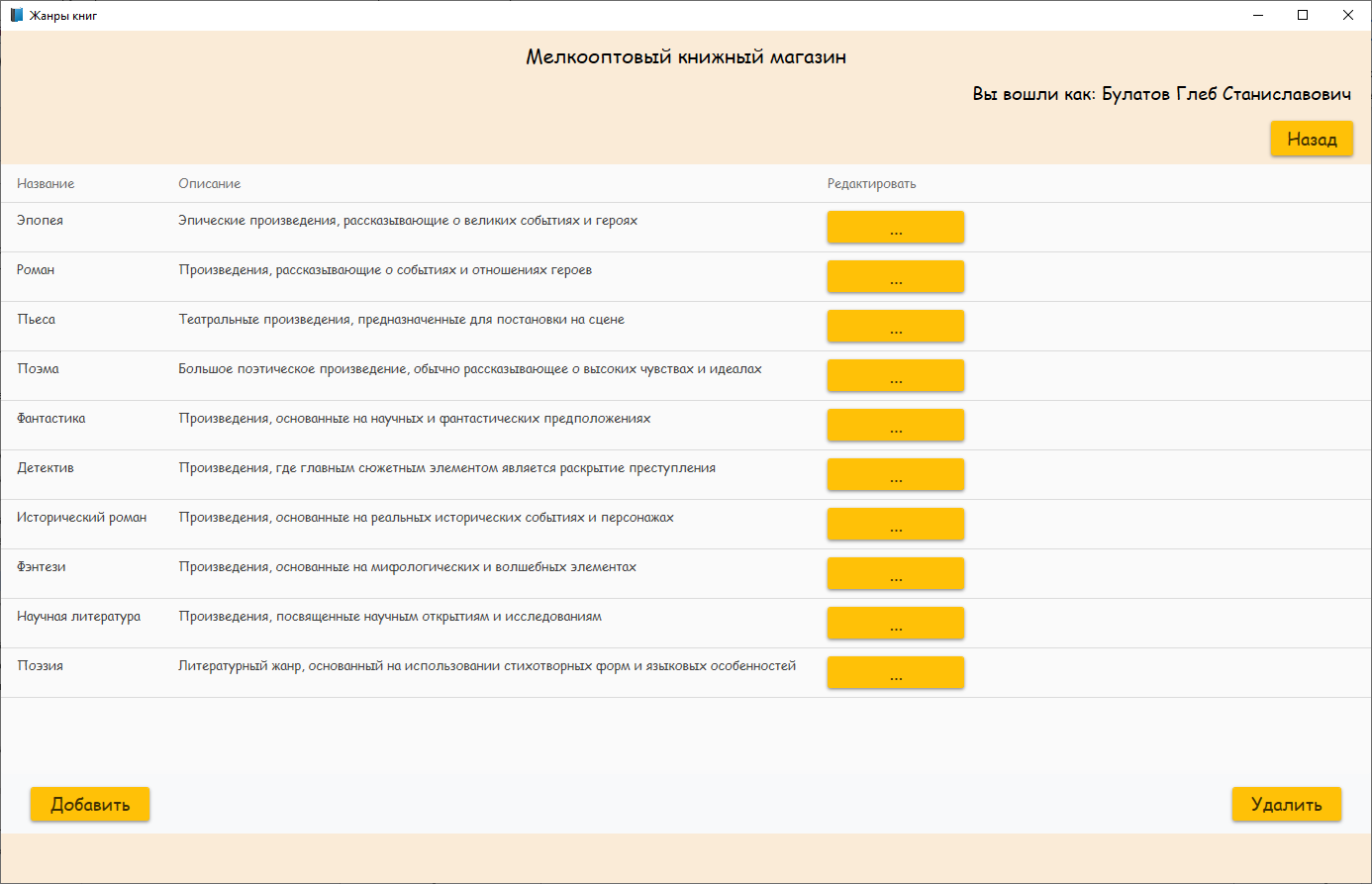
TextBox

Button

Рис. 29 Страница «Добавление и редактирование поставщика»

Страница «**Жанры книг»**. Интерфейс данной формы состоит из компонентов: DataGrid и трех кнопок: Назад, Добавить и Удалить (рис. 30).

Компонент DataGrid - таблица

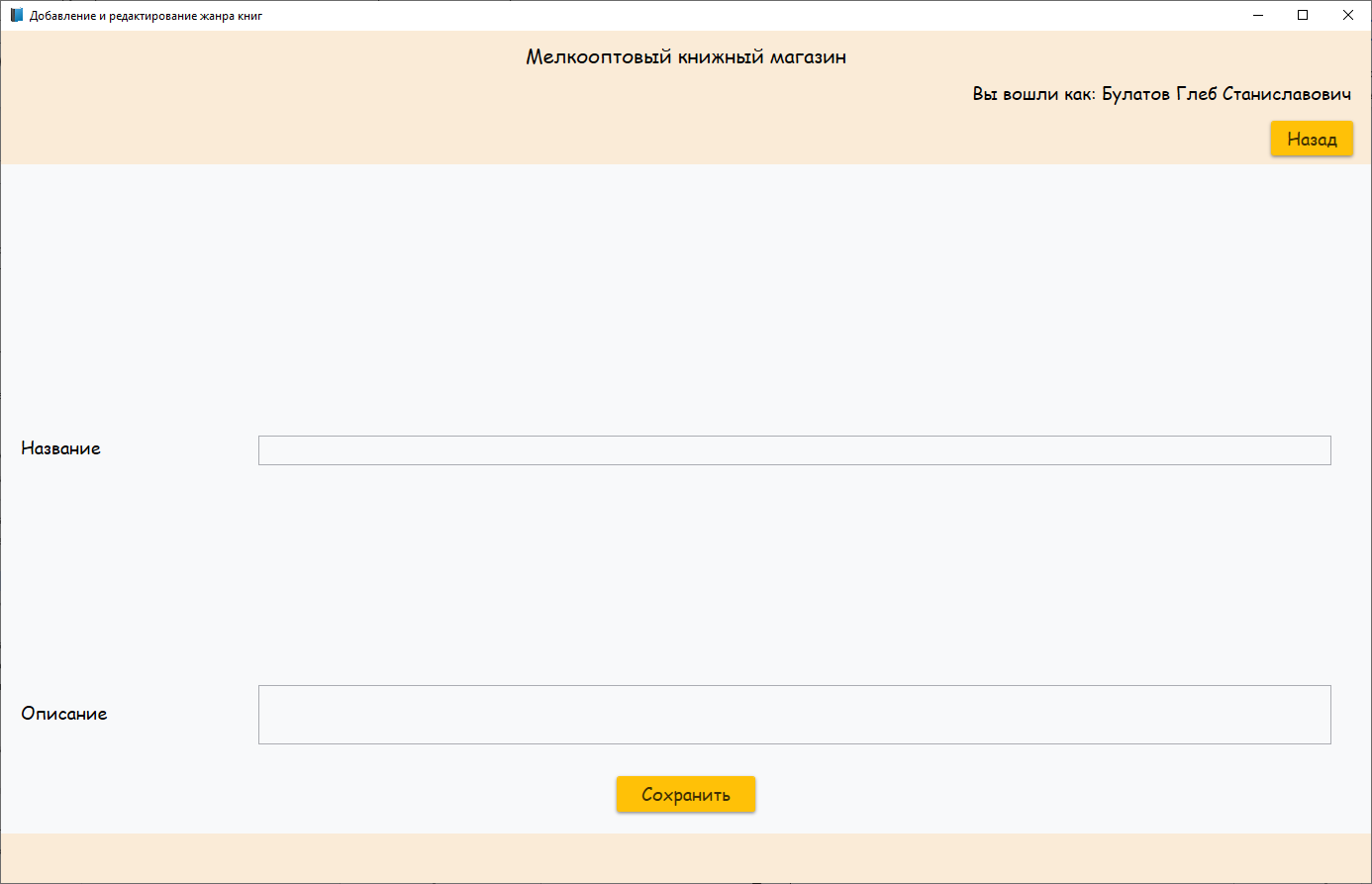


Button – кнопки Добавить и Удалить

Button – кнопка Назад

Рис. 30 Страница «Жанры книг»

Страница **«Добавление и редактирование жанра книг»**. Интерфейс данной формы состоит из компонентов: TextBox – поля ввода и Button – кнопок (рис. 31).



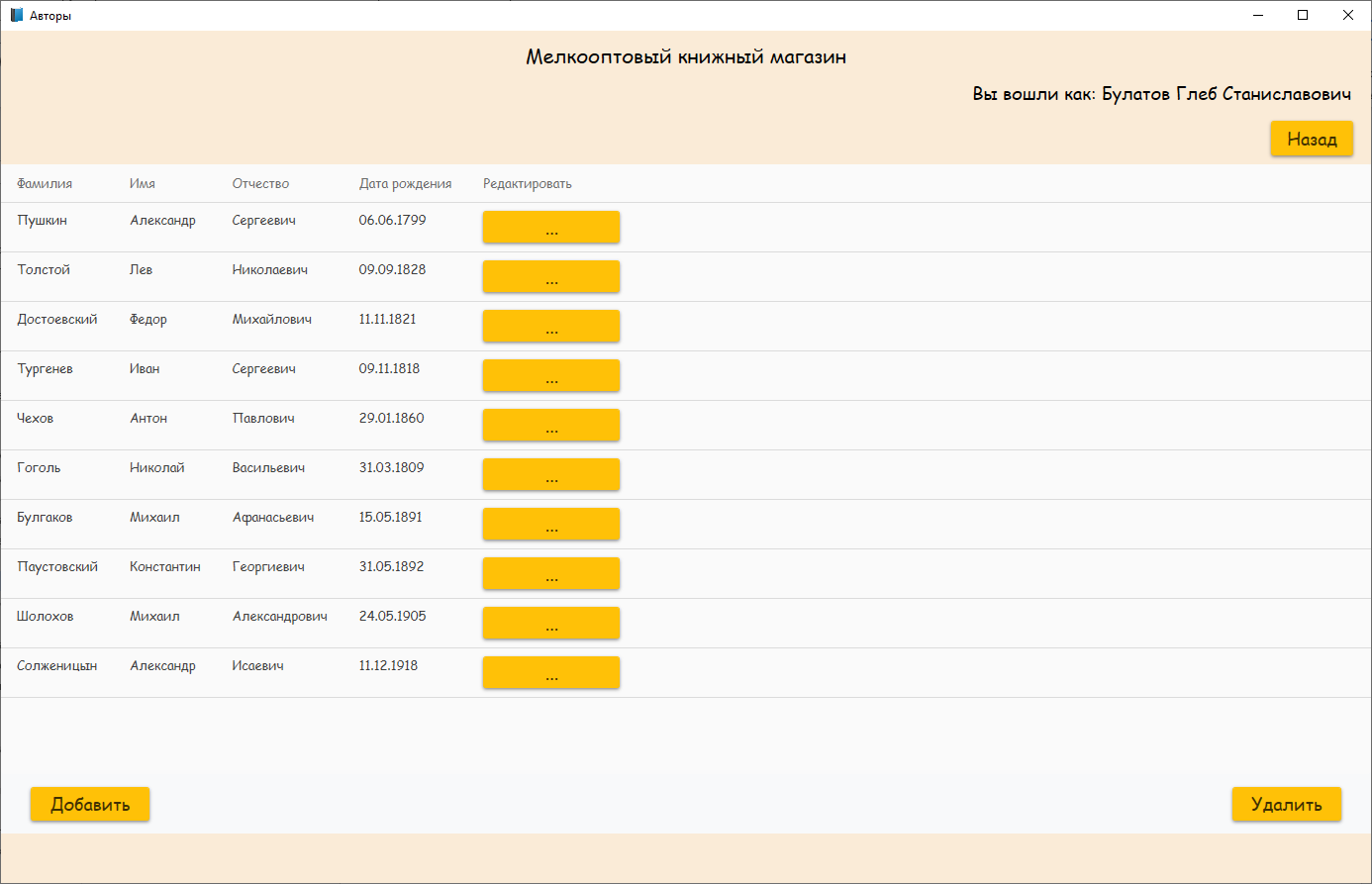
TextBox

Button

Рис. 31 Страница «Добавление и редактирование жанра книг»

Страница «**Авторы»**. Интерфейс данной формы состоит из компонентов: DataGrid и трех кнопок: Назад, Добавить и Удалить (рис. 32).

Компонент DataGrid - таблица

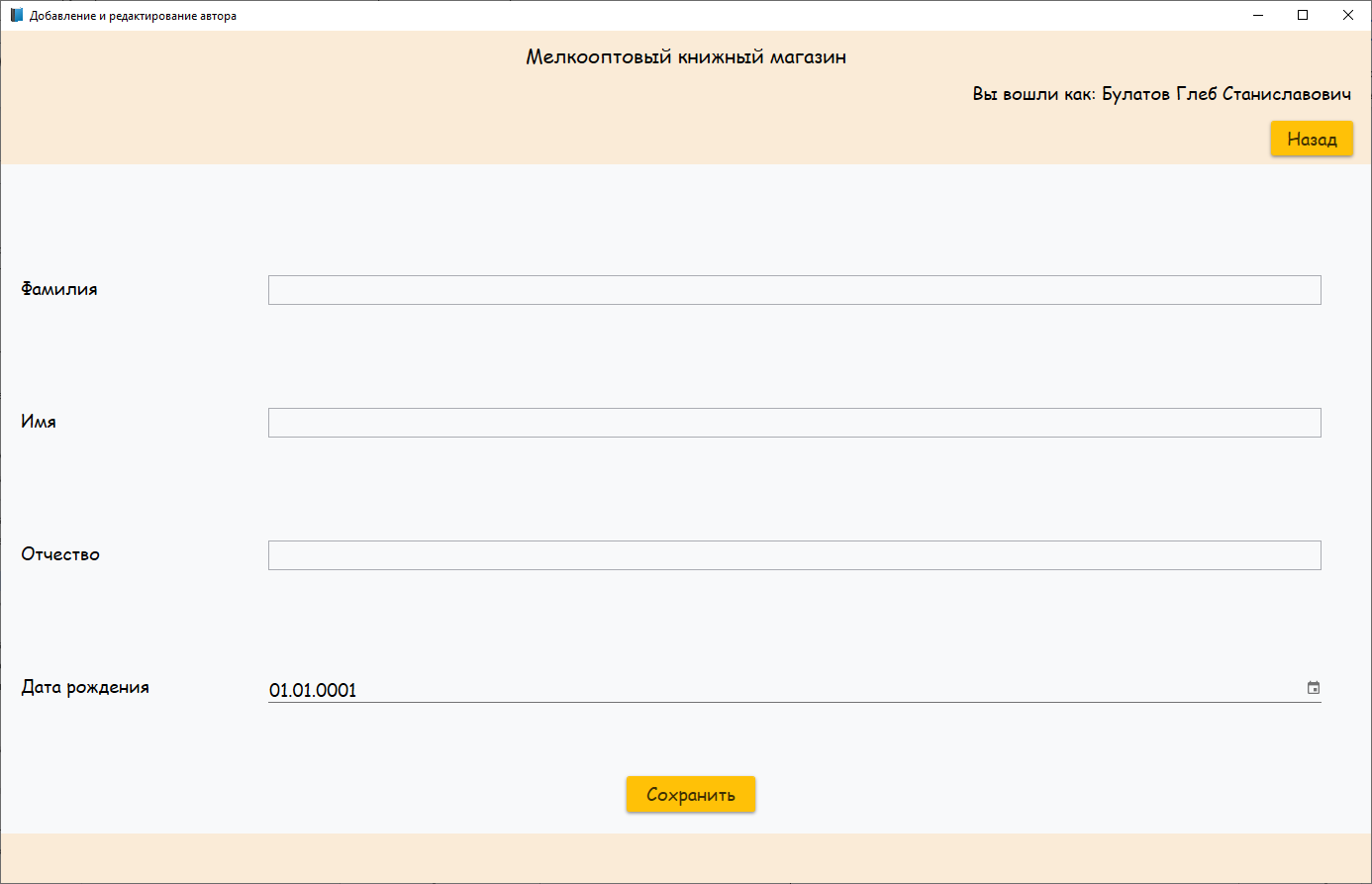


Button – кнопки Добавить и Удалить

Button – кнопка Назад

Рис. 32 Страница «Авторы»

Страница **«Добавление и редактирование автора»**. Интерфейс данной формы состоит из компонентов: TextBox – поля ввода и Button – кнопок (рис. 33).



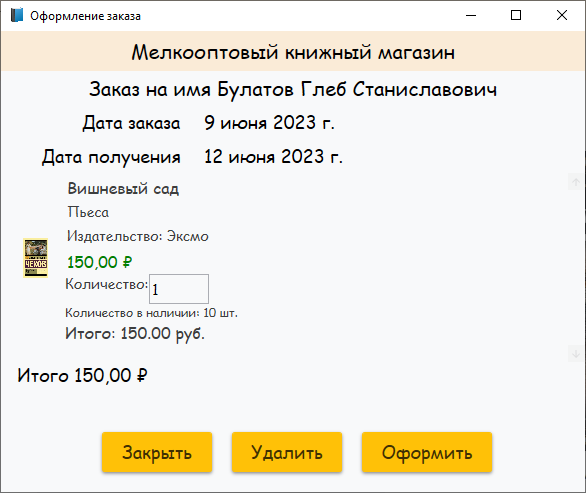
TextBox

Button

DatePicker

Рис. 33 Страница «Добавление и редактирование автора»

Страница «**Оформление заказа»**. Интерфейс данной формы состоит из компонентов: TextBlock – блок текста, TextBox – поле ввода, Button – кнопки (рис. 34).



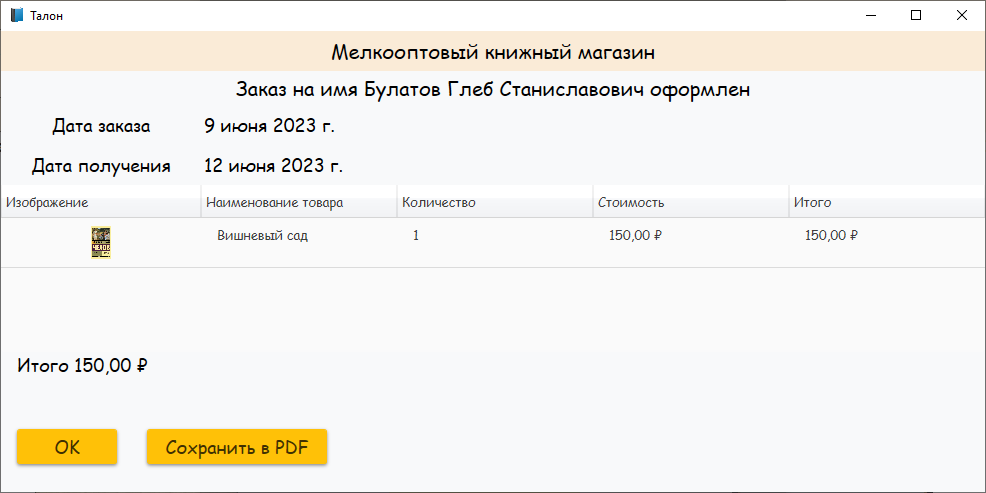
Button – кнопки

TextBox

TextBlock

Рис. 34 Страница «Оформление заказа»

Страница **«Талон»**. Интерфейс данной формы состоит из компонентов: DataGrid – таблица данных, TextBlock – блок текста, Button – кнопки (рис. 35).



Button – кнопки

DataGrid

TextBlock

Рис. 35 Страница «Талон»

**3.7 Описание структуры выходной информации**

Выходными данными являются список книг и список заказов, отфильтрованные по разным критериям.

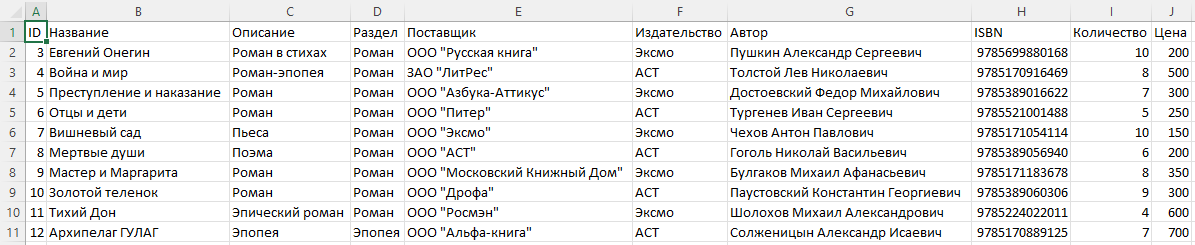


Рис. 36 Список книг

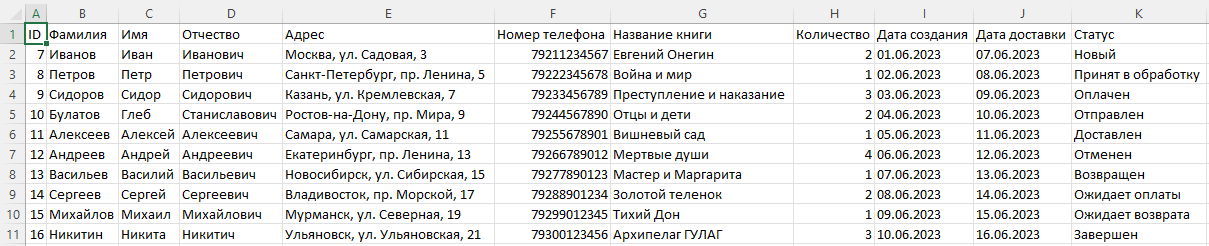


Рис. 37 Список заказов

**4 РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ**

**4.1 Назначение программного средства**

Назначением программного средства является автоматизация работы мелкооптового книжного магазина.

**4.2 Условия выполнения программного средства**

Условия, при соблюдении которых возможно выполнение работы с программным продуктом:

- Компьютер с установленной на нём «BookStoreApp».

Подготовительные действия:

- На иконке «BookStoreApp» рабочего стола произвести двойной щелчок левой кнопкой мыши.

**4.3 Эксплуатация программного средства**

Этот раздел содержит обзор приложений «BookStoreApp» и ее основных параметров.

Вводные сведения об интерфейсе пользователя

После запуска приложения отображается стартовое окно приложения (рис. 38). Главная форма представляет собой каталог товаров мелкооптового книжного магазина в виде плиточного интерфейса. Каждый товар отображается в виде карточки с краткой информацией: название, стоимость, количество. Пользователь имеет возможность отфильтровать список по жанру, по издательству, по автору, а также отсортировать по цене.

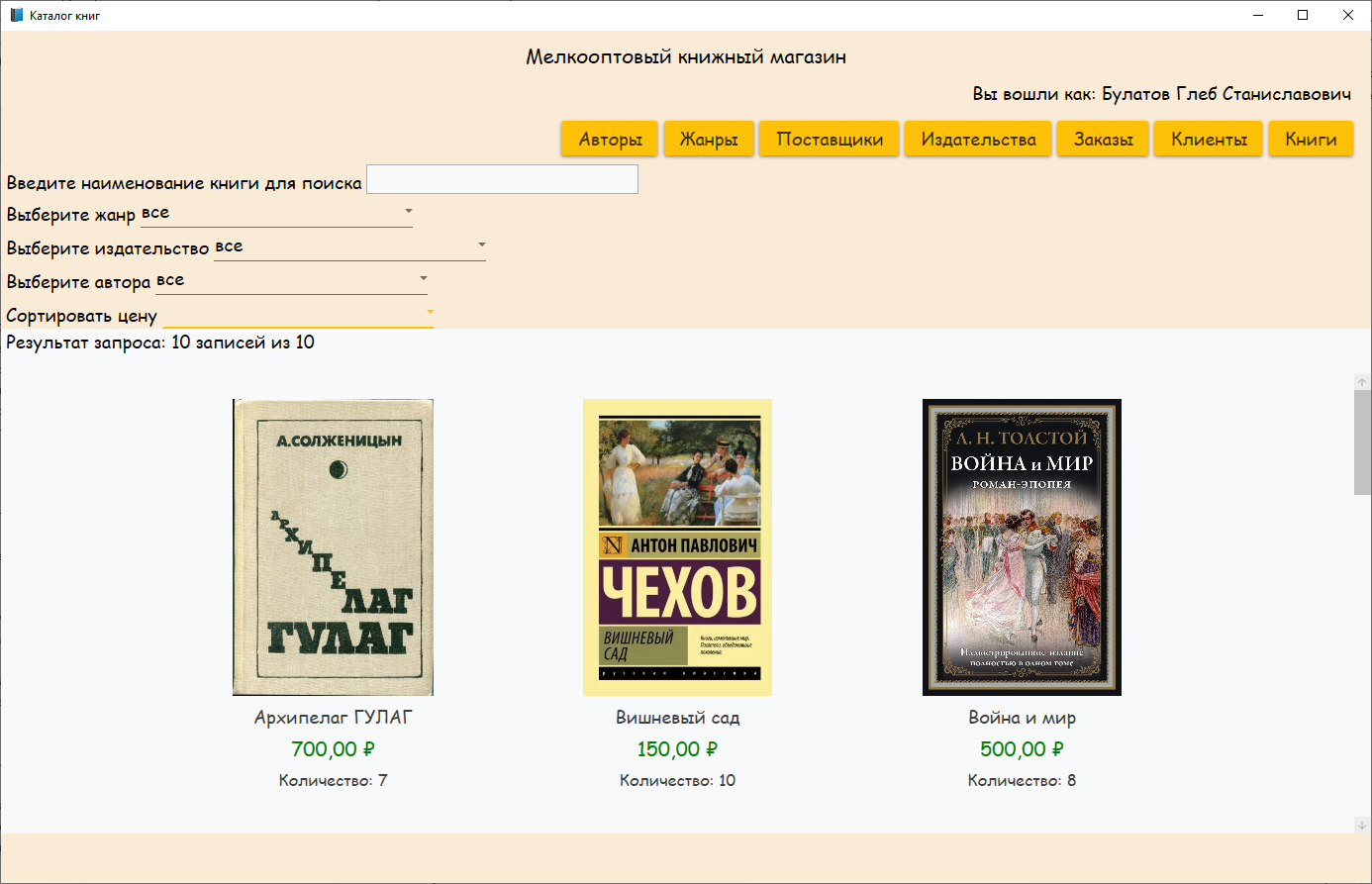


Рис. 38 Окно «Автоматизированная информационная система "Мелкооптовый книжный магазин"»

Кликнув по кнопке «Закрыть» программа потребует подтверждение (рис. 39).

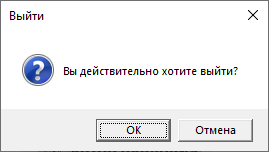


Рис. 39 Сообщение программы

Для входа в систему необходимо ввести логин и пароль пользователя и нажать OK (рис. 40).

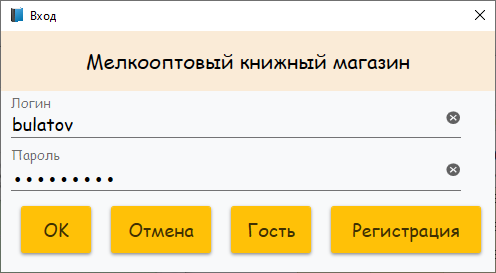
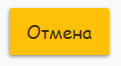


Рис. 40 Форма входа в систему

Для того, чтобы выйти из системы нажмите на кнопку , программа потребует подтверждения (рис. 41).

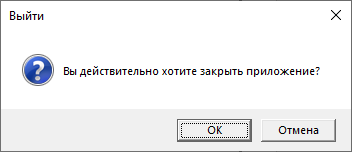


Рис. 41 Сообщение программы

«Книги»: эта страница предназначена для ввода, удаления и редактирования данных о книгах. Функционал страницы позволяет выгружать список книг в Excel. Для каждой книги можно открыть дополнительную страницу для просмотра или редактирования записи. Пользователь имеет возможность отфильтровать список по ID, по названию, а также отсортировать по фамилии автора (рис. 42).

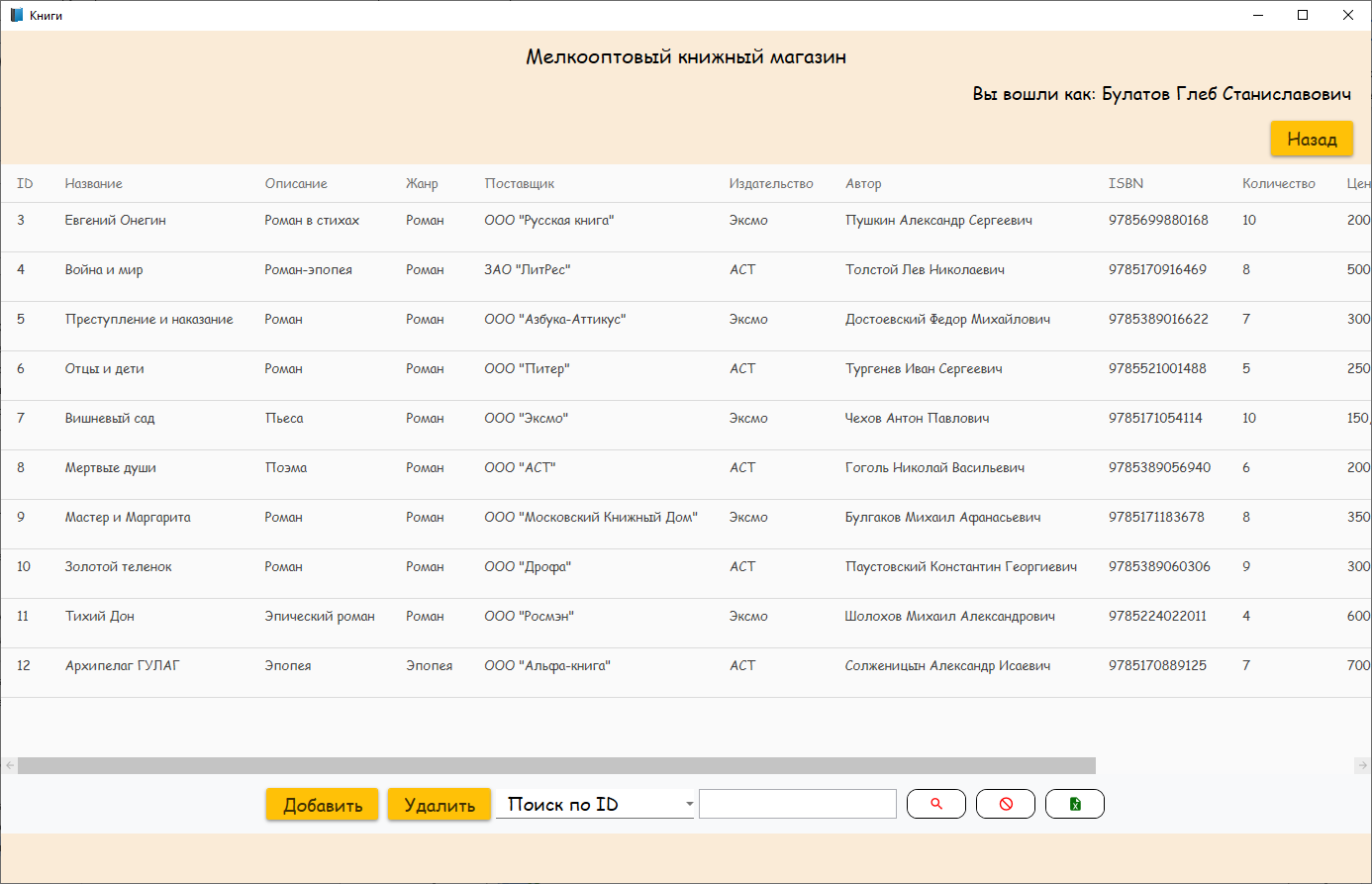


Рис. 42 Страница «Книги»

«Добавление и редактирование книги»: эта страница предназначена для ввода или редактирования данных о книге (рис. 43).

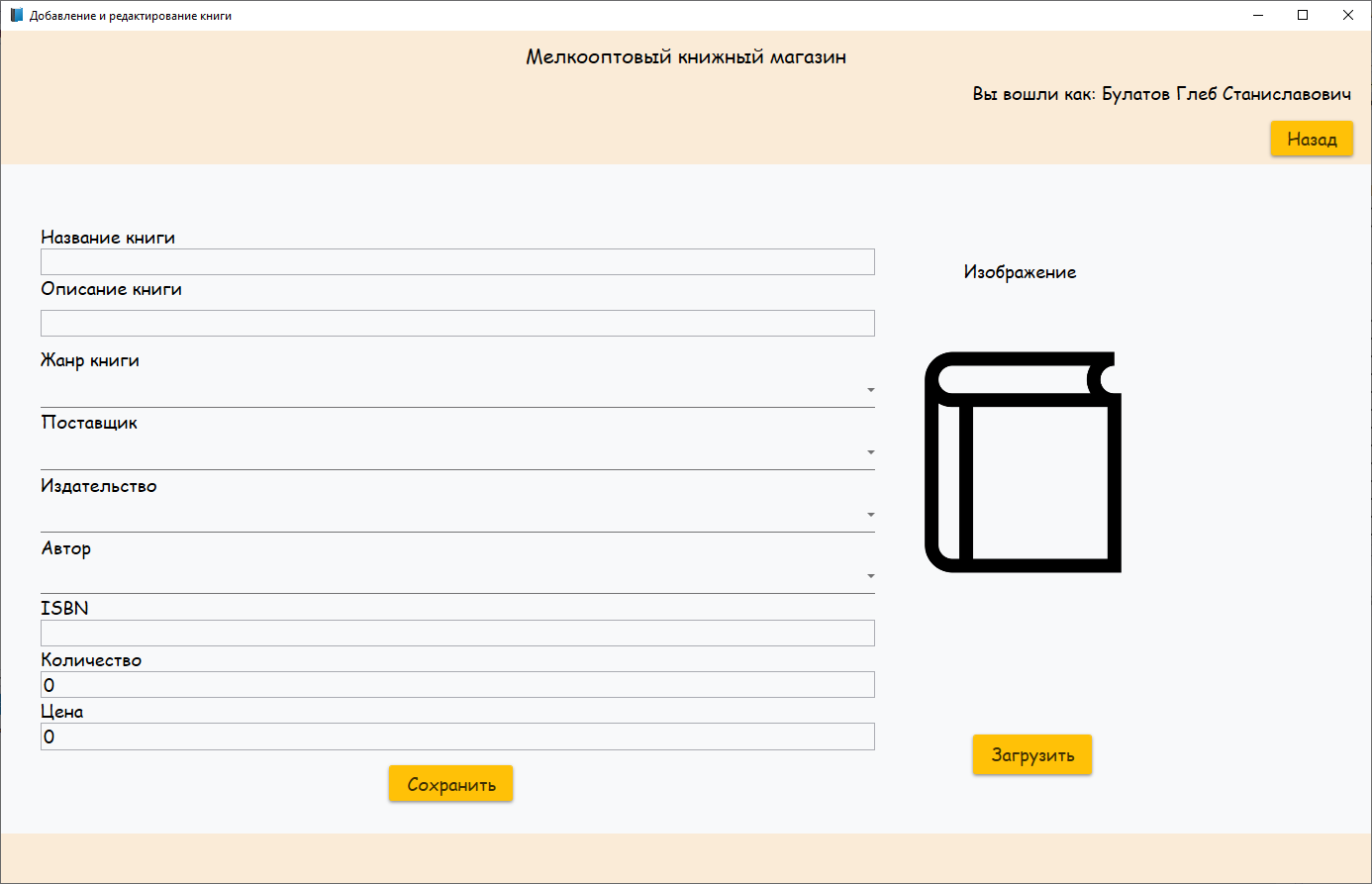


Рис. 43 Страница «Добавление и редактирование книги»

«Клиенты»: эта страница предназначена для ввода, удаления и редактирования данных о клиентах. Для каждого клиента можно открыть дополнительную страницу для просмотра или редактирования записи (рис. 44).

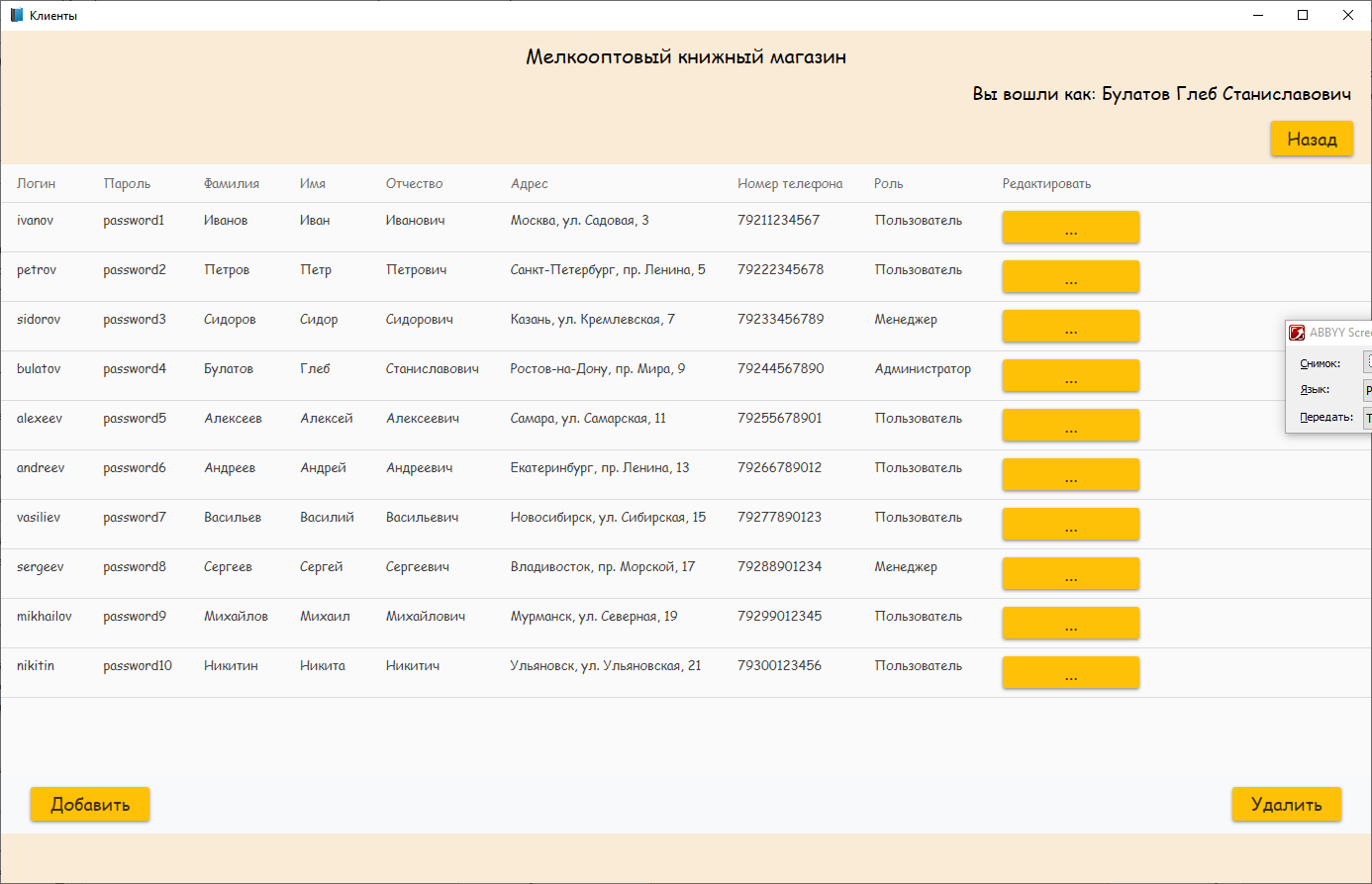


Рис. 44 Страница «Клиенты»

«Добавление и редактирование клиента»: эта страница предназначена для ввода или редактирования данных о клиенте (рис. 45).

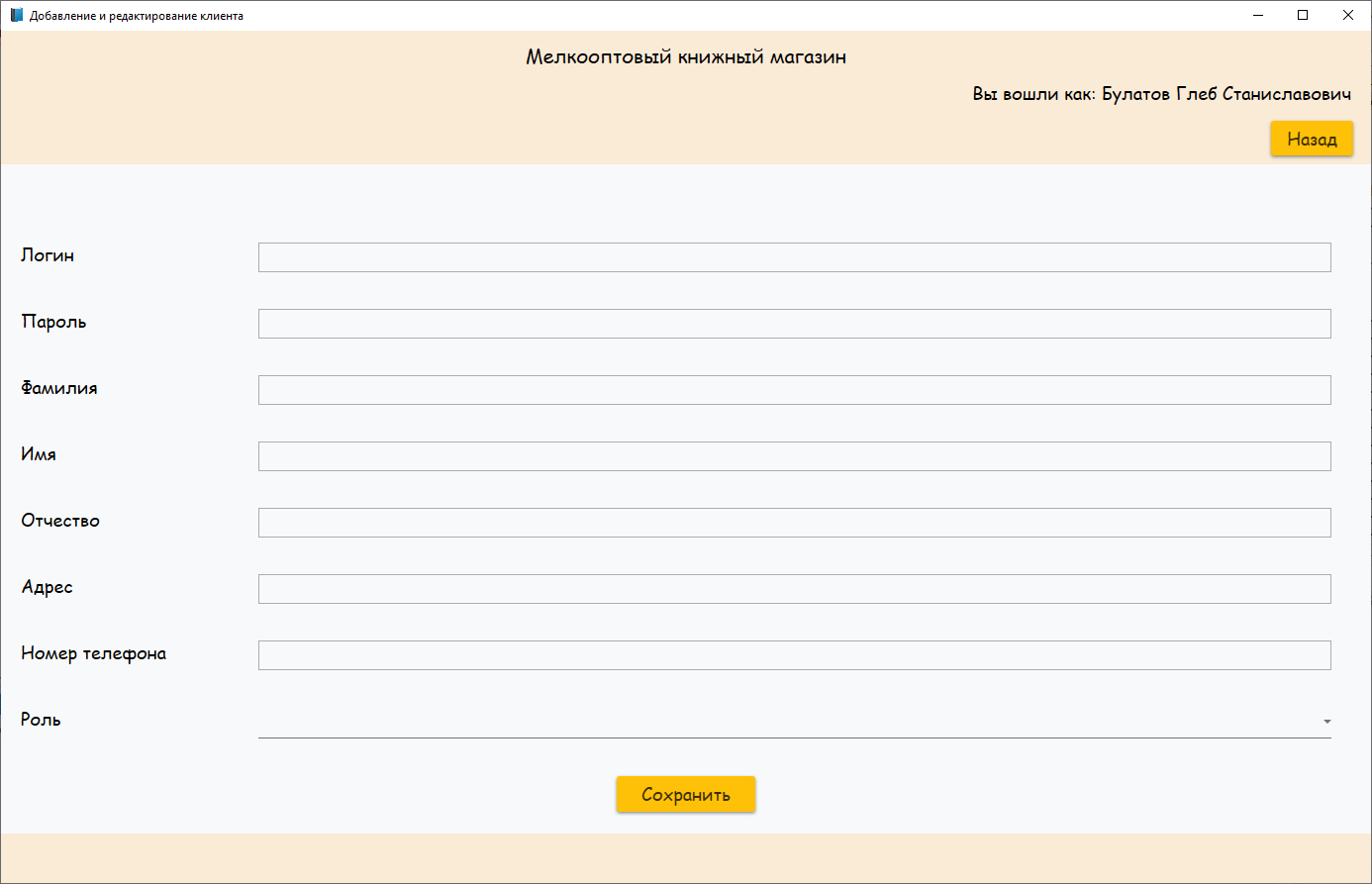


Рис. 45 Страница «Добавление и редактирование клиента»

«Заказы»: эта страница предназначена для ввода, удаления и редактирования данных о заказах. Функционал страницы позволяет выгружать список заказов в Excel. Для каждого заказа можно открыть дополнительную страницу для просмотра или редактирования записи. Пользователь имеет возможность отфильтровать список по ID, по фамилии клиента, по дате подачи заявки, а также отсортировать по номеру телефона (рис. 46).



Рис. 46 Страница «Заказы»

«Добавление и редактирование заказа»: эта страница предназначена для ввода или редактирования данных о заказе (рис. 47).

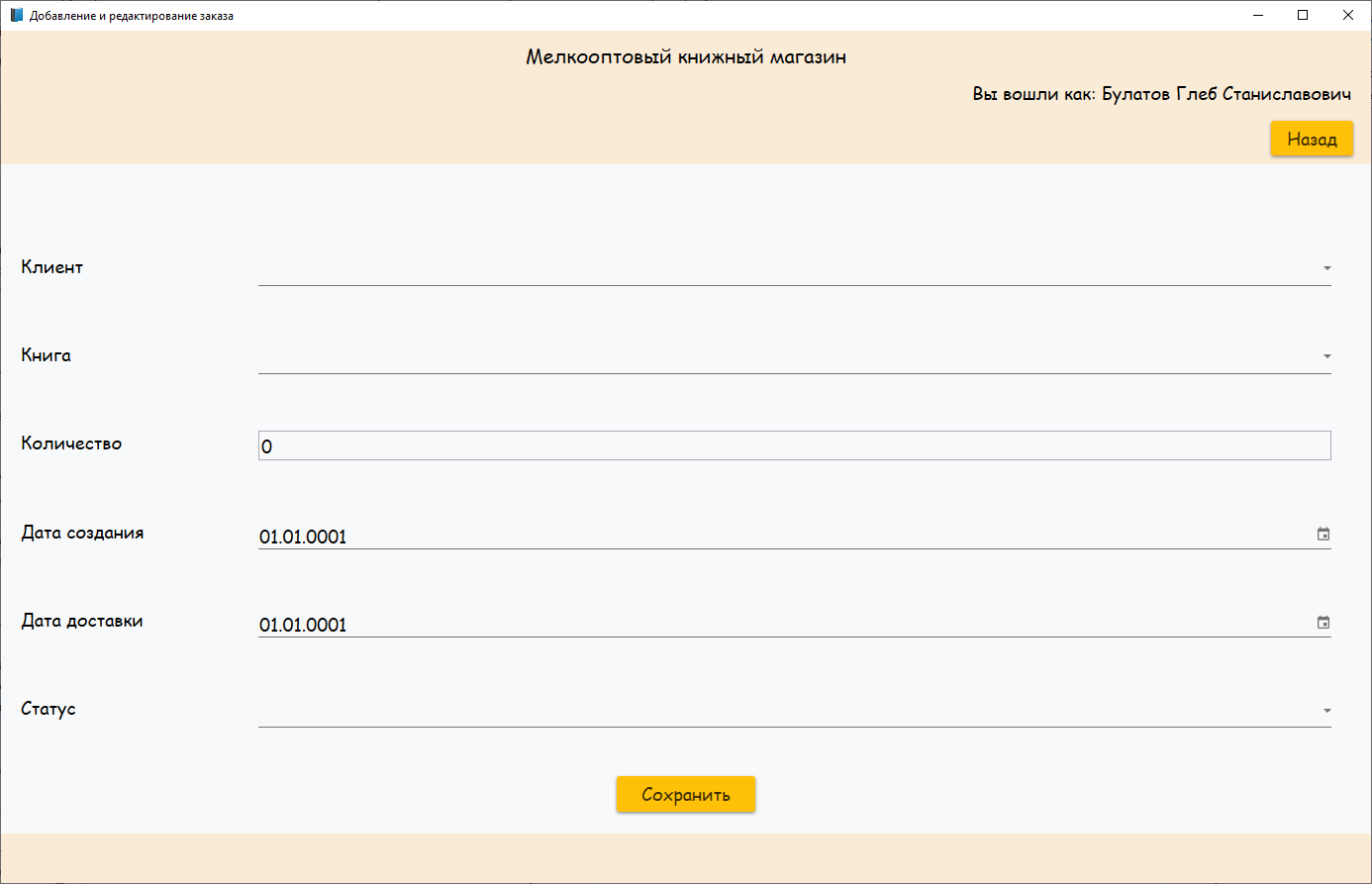


Рис. 47 Страница «Добавление и редактирование заказа»

«Издательства»: эта страница предназначена для ввода, удаления и редактирования данных об издательствах. Для каждого издательства можно открыть дополнительную страницу для просмотра или редактирования записи (рис. 48).

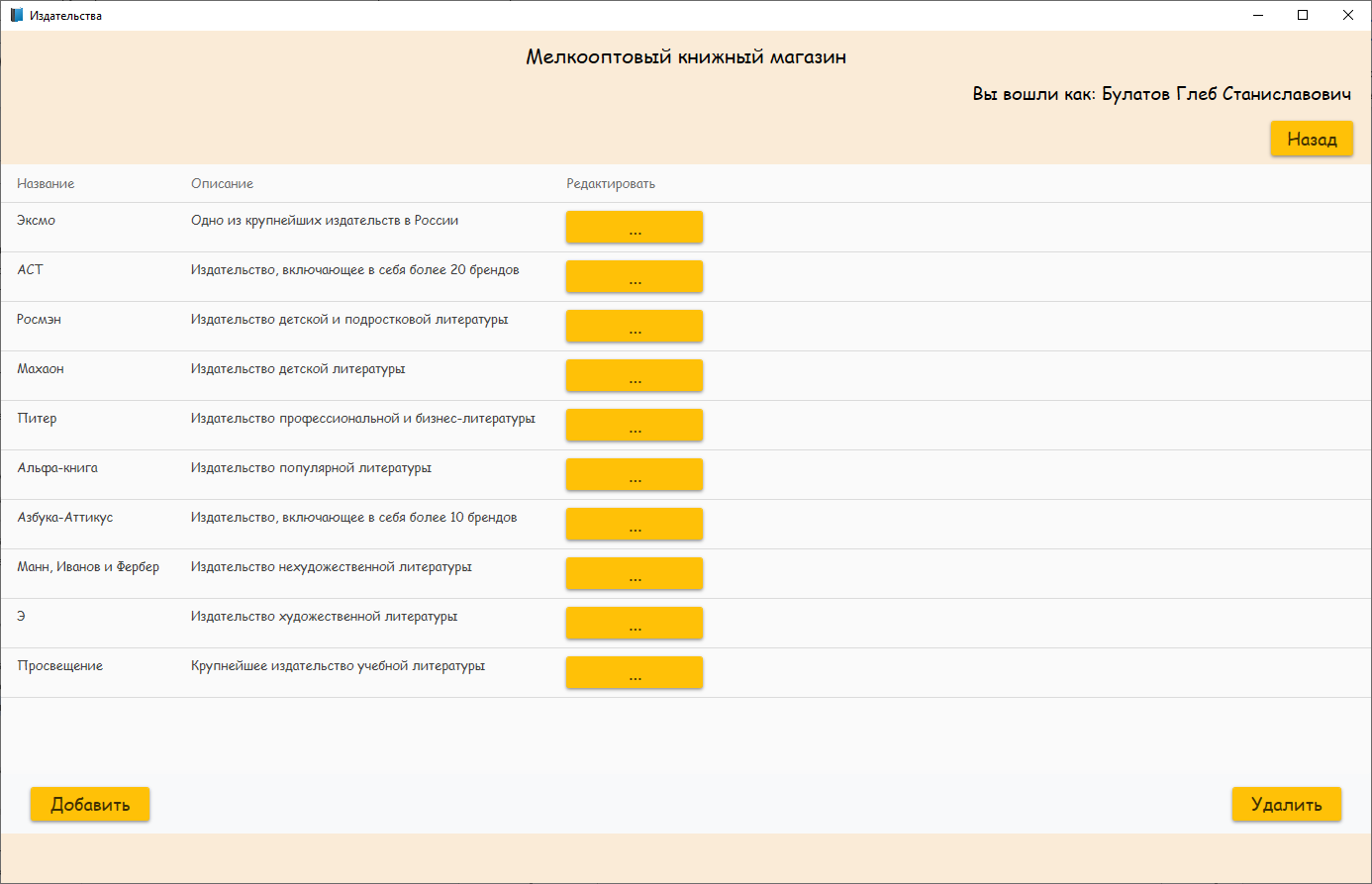


Рис. 48 Страница «Издательства»

«Добавление и редактирование издательства»: эта страница предназначена для ввода или редактирования данных об издательстве (рис. 49).

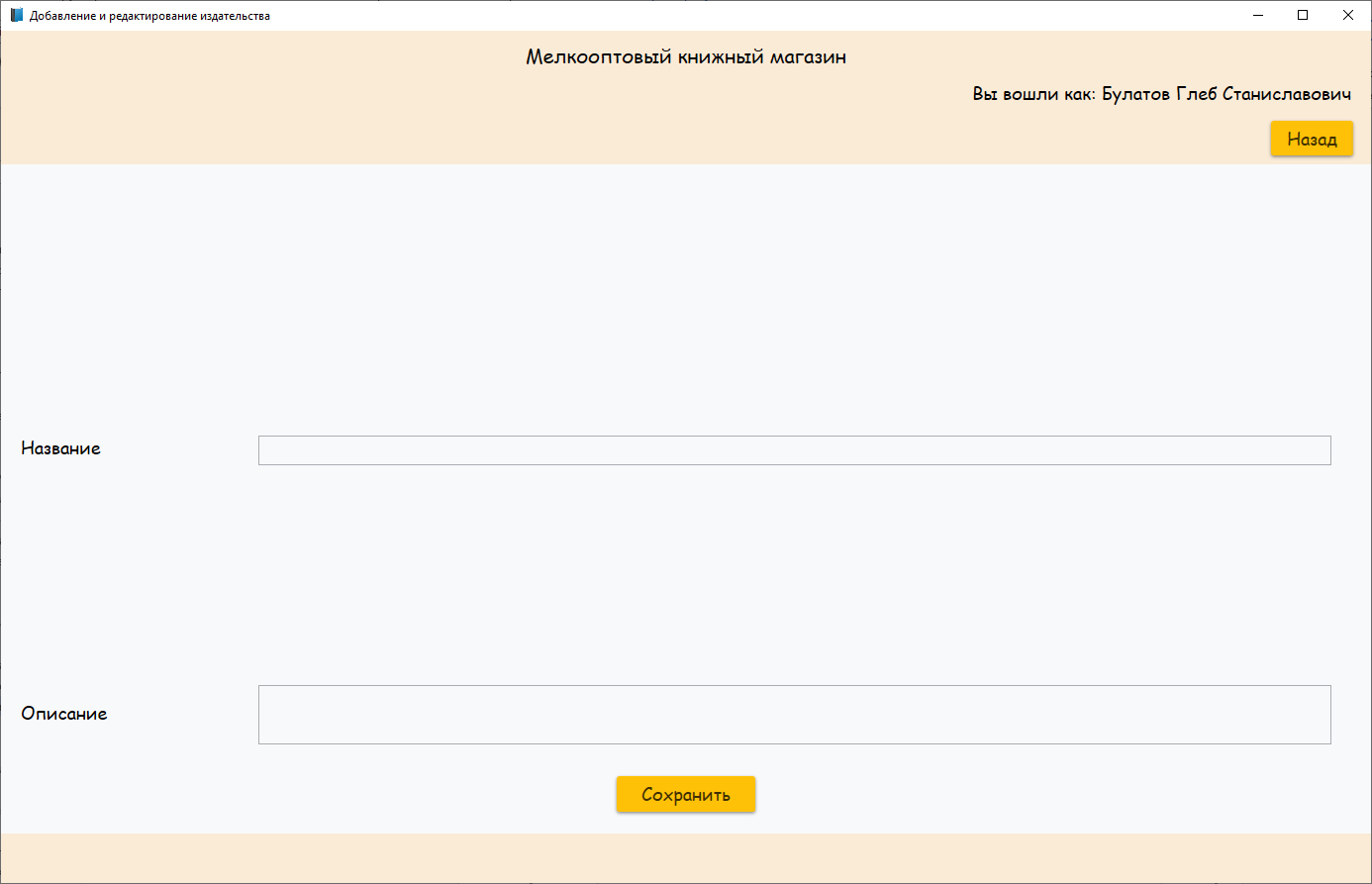


Рис. 49 Страница «Добавление и редактирование издательства»

«Поставщики»: эта страница предназначена для ввода, удаления и редактирования данных о поставщиках. Для каждого поставщика можно открыть дополнительную страницу для просмотра или редактирования записи (рис. 50).

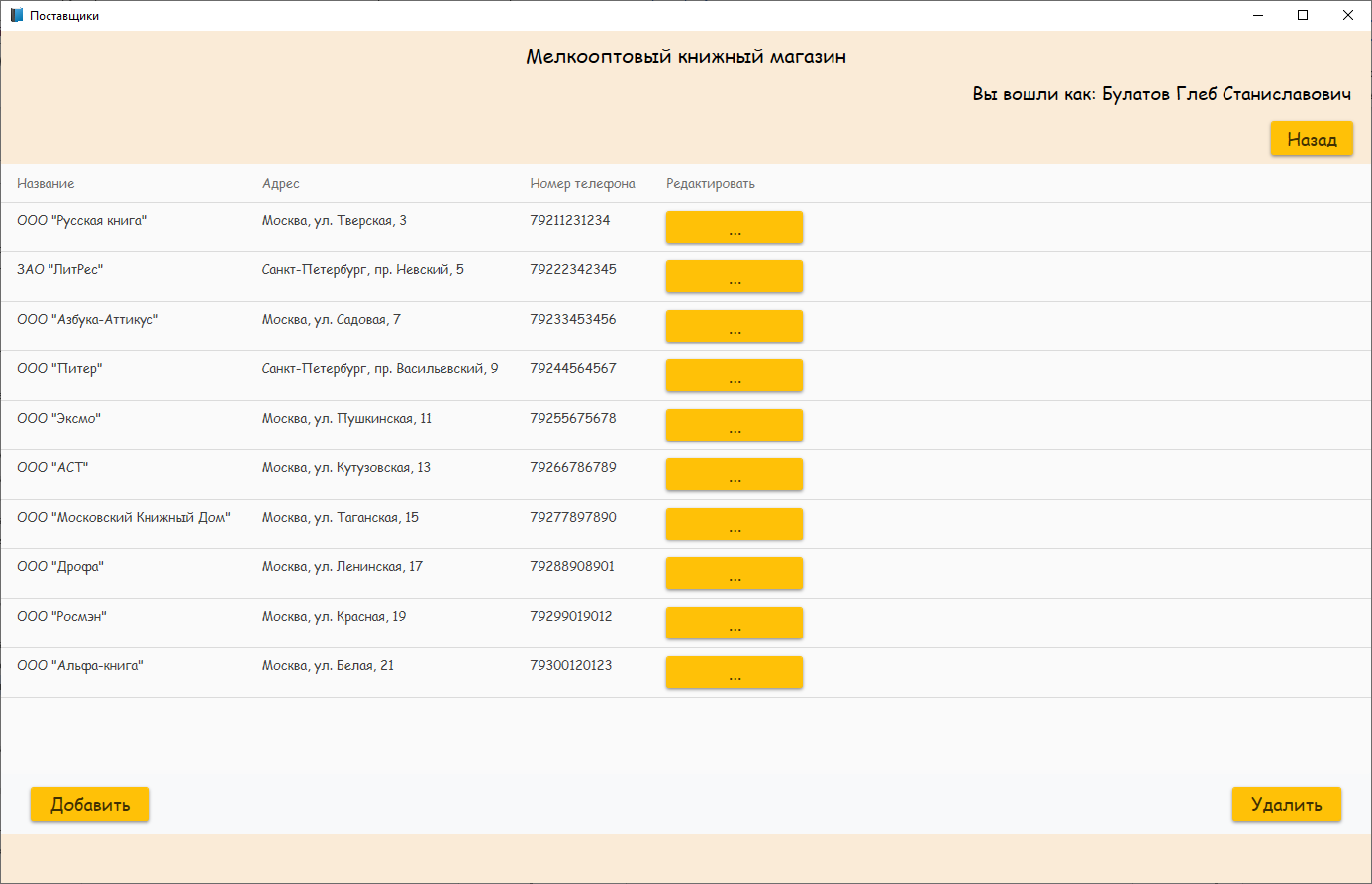


Рис. 50 Страница «Поставщики»

«Добавление и редактирование поставщика»: эта страница предназначена для ввода или редактирования данных о поставщике (рис. 51).

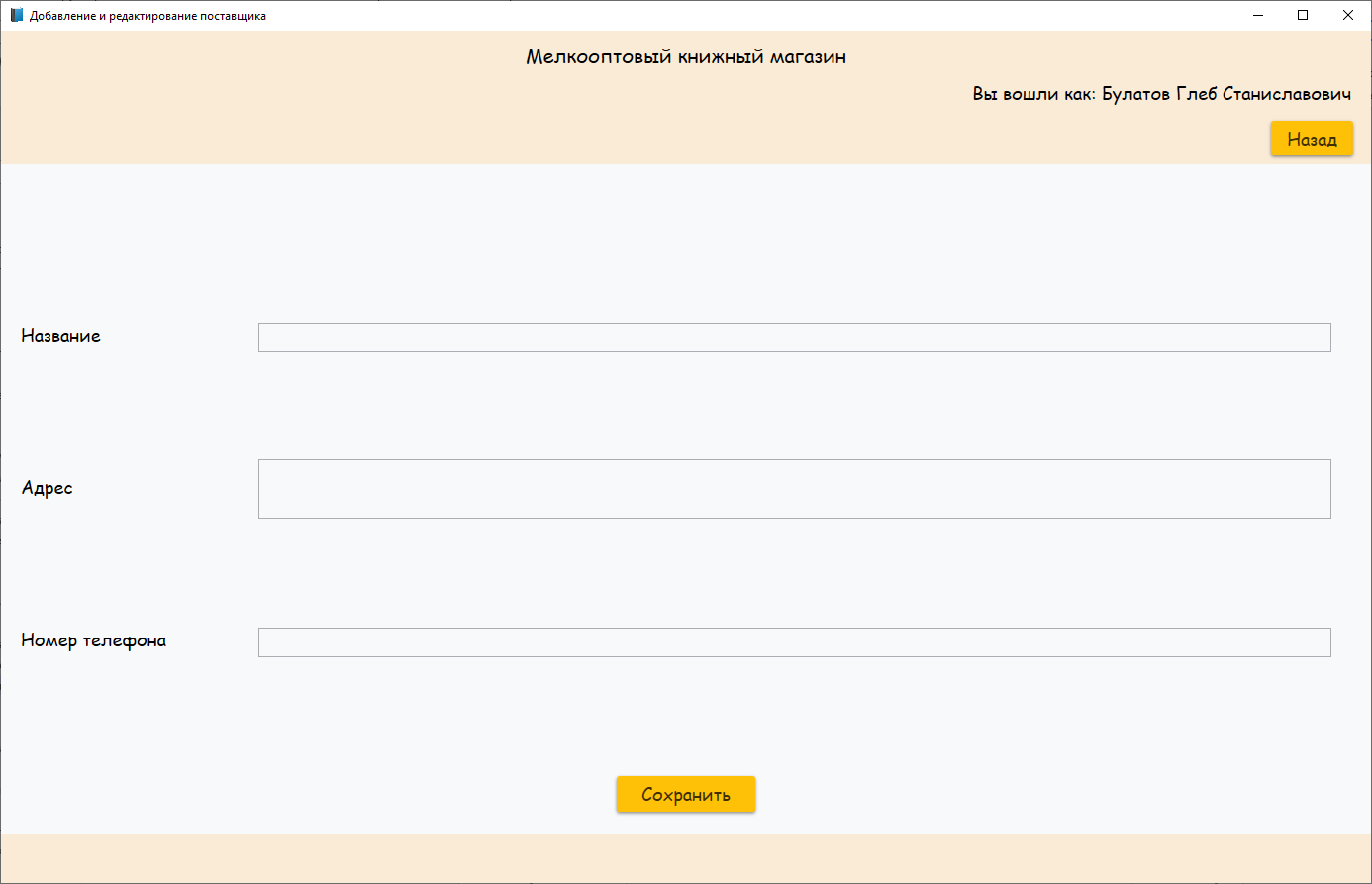


Рис. 51 Страница «Добавление и редактирование поставщика»

«Жанры книг»: эта страница предназначена для ввода, удаления и редактирования данных о жанре книг. Для каждого жанра книг можно открыть дополнительную страницу для просмотра или редактирования записи (рис. 52).

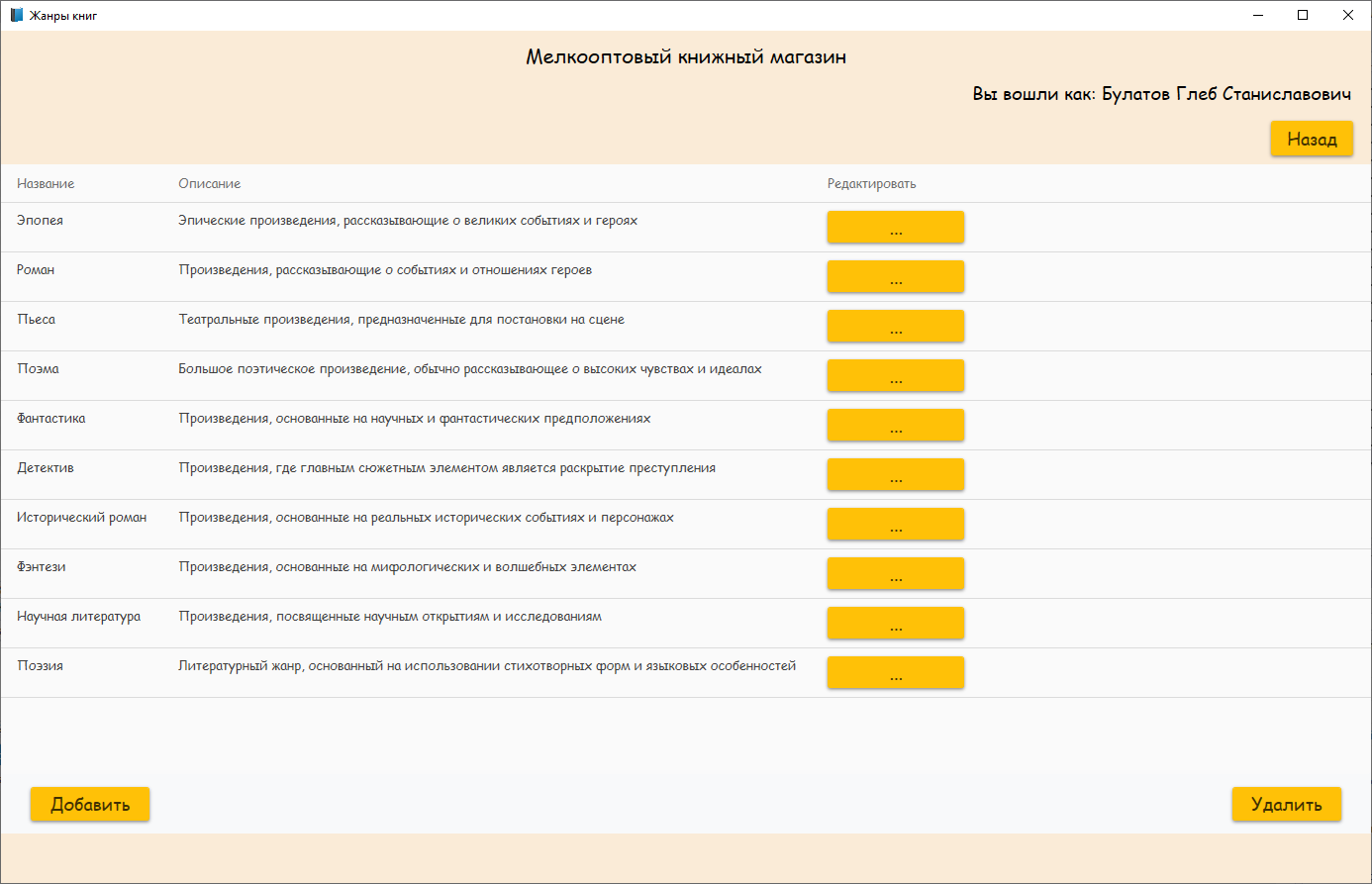


Рис. 52 Страница «Жанры книг»

«Добавление и редактирование жанра книг»: эта страница предназначена для ввода или редактирования данных о жанре книг (рис. 53).

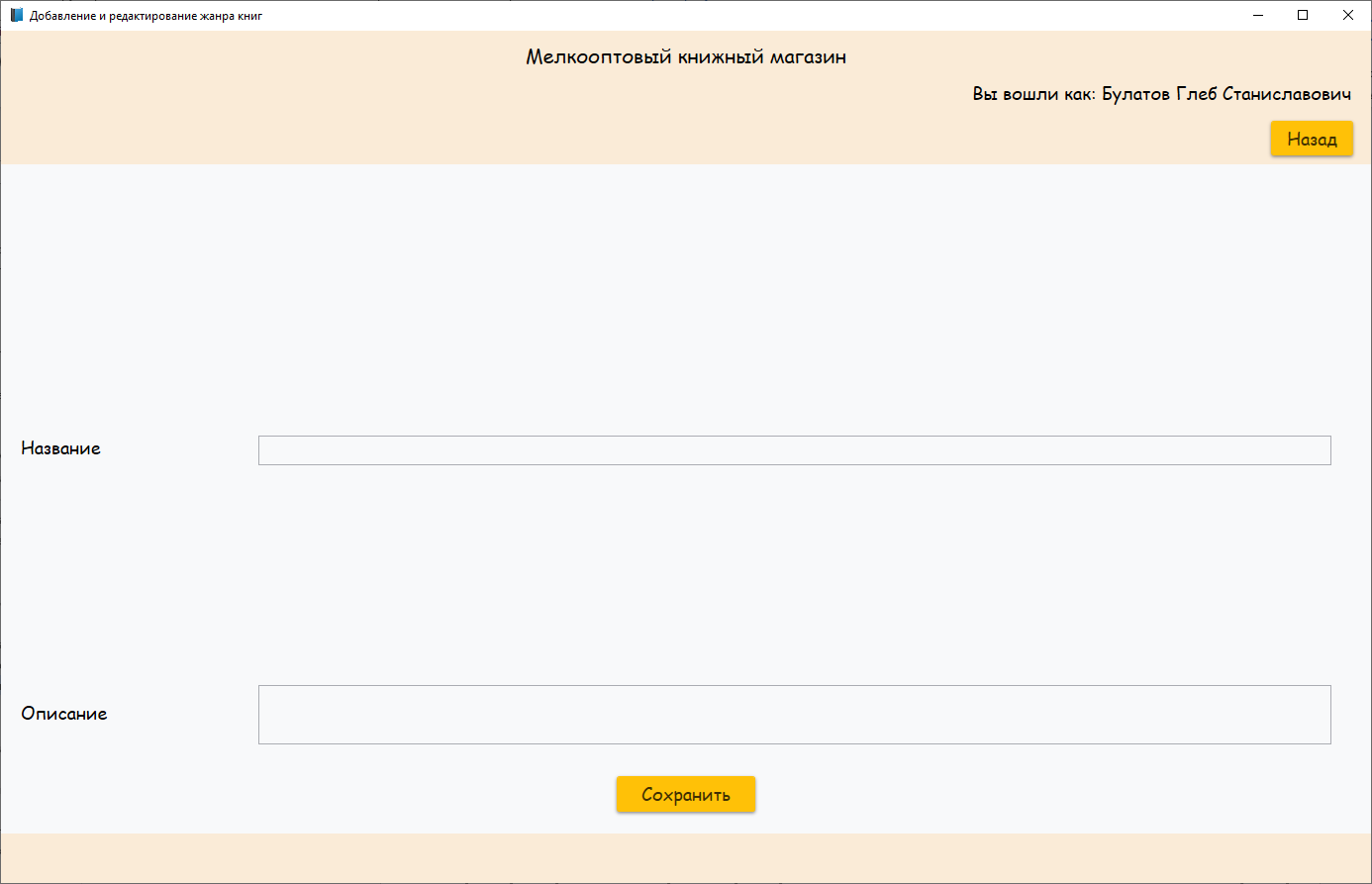


Рис. 53 Страница «Добавление и редактирование поставщика»

«Авторы»: эта страница предназначена для ввода, удаления и редактирования данных об авторе. Для каждого автора можно открыть дополнительную страницу для просмотра или редактирования записи (рис. 54).

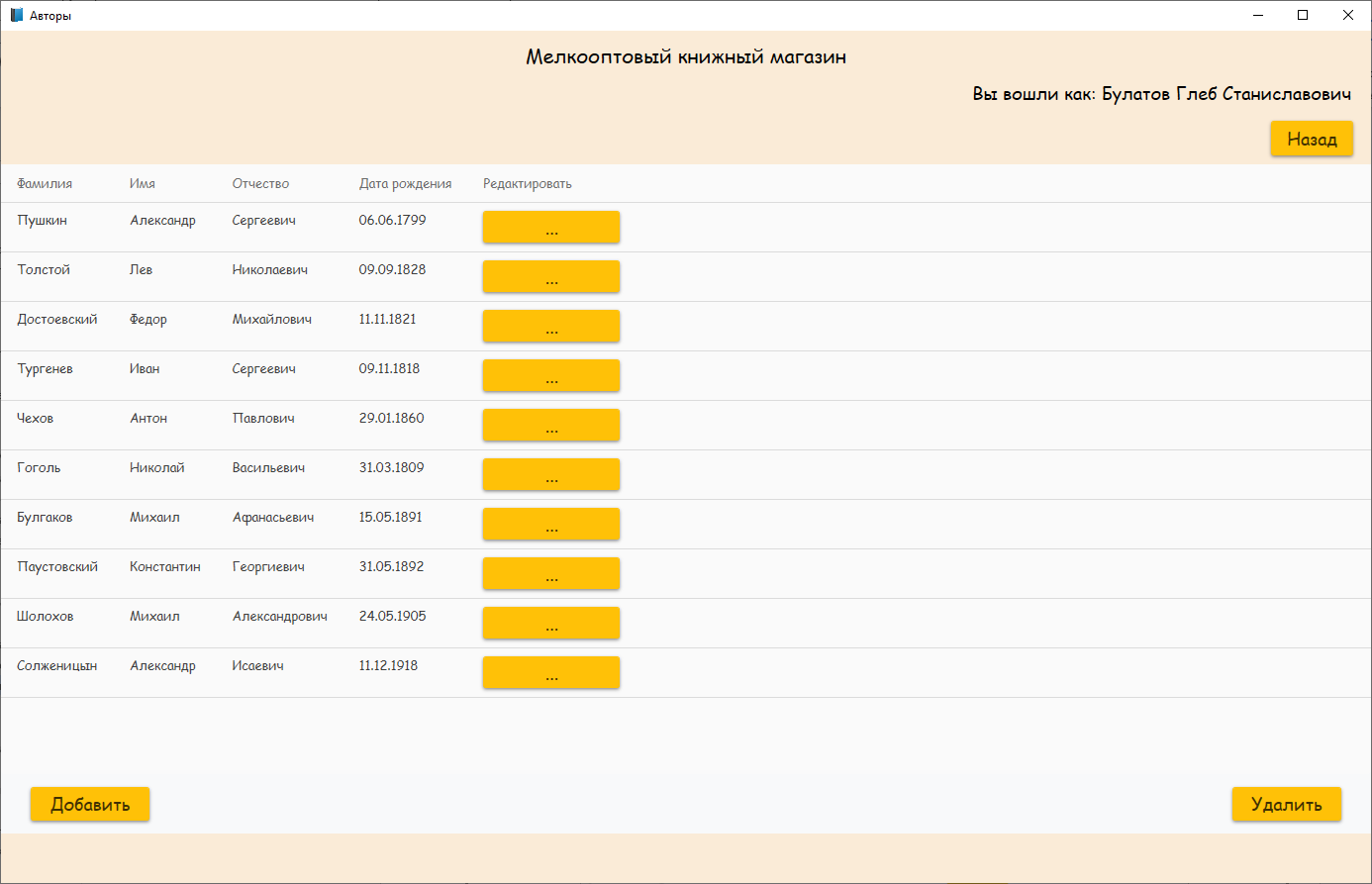


Рис. 54 Страница «Авторы»

«Добавление и редактирование автора»: эта страница предназначена для ввода или редактирования данных об авторе (рис. 55).

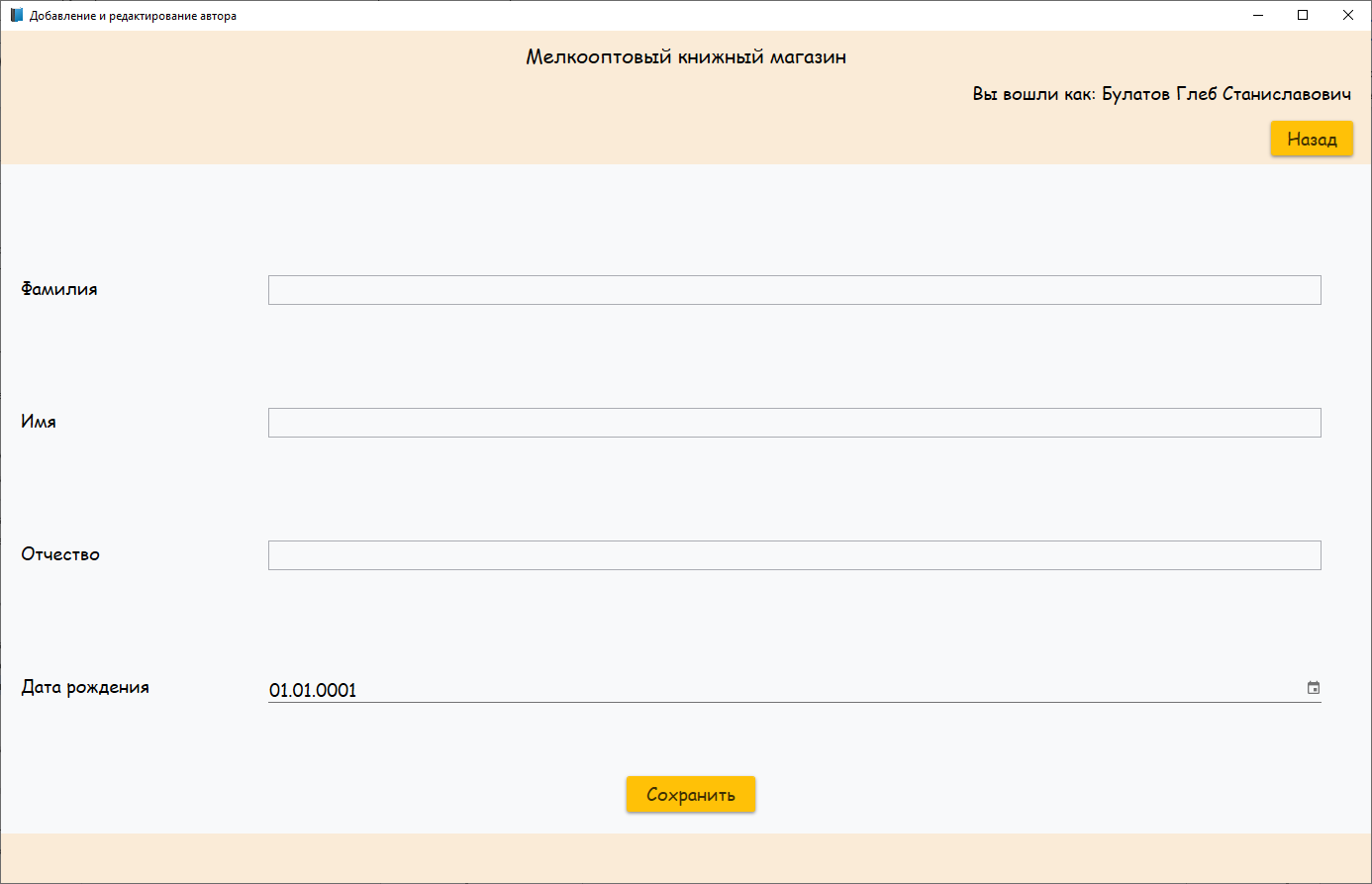


Рис. 55 Страница «Добавление и редактирование автора»

«Оформление заказа»: эта страница предназначена для оформления заказа с книгами (рис. 56).

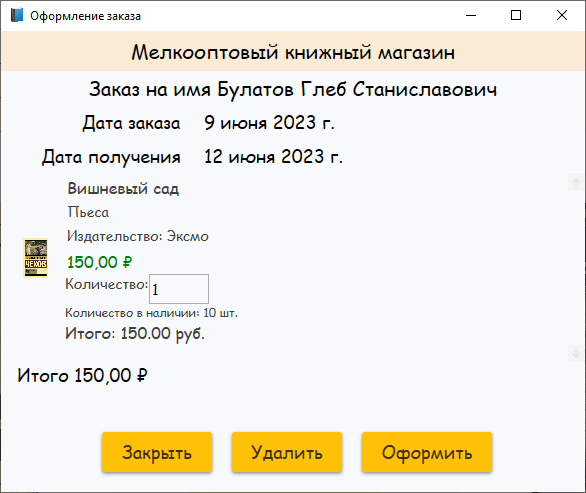


Рис. 56 Страница «Оформление заказа»

«Талон»: эта страница предназначена для вывода информации о созданном заказе (рис. 57).

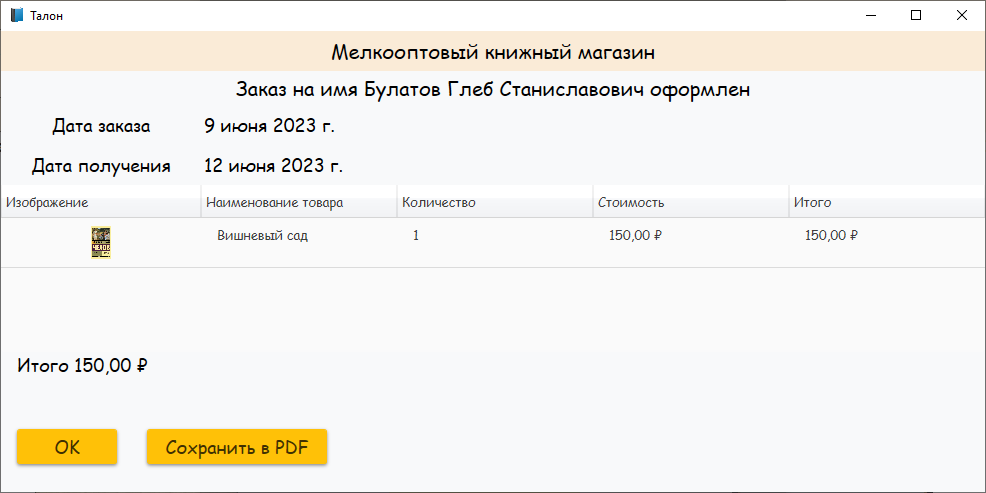


Рис. 57 Страница «Талон»

«Регистрация»: это окно предназначено для регистрации клиента (рис. 58).

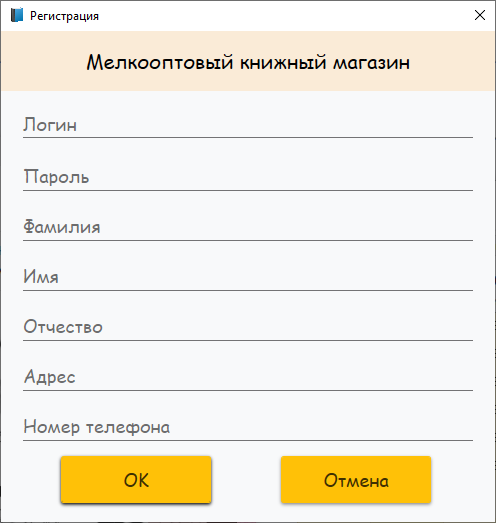


Рис. 58 Страница «Талон»

Работа с BookStoreApp

После авторизации в приложении Вы перейдете на главную форму программы – Каталог книг. На страницах Книги, Клиенты, Издательства, Поставщики, Жанры, Авторы заполняем все необходимые для работы пункты.

После этого кликаем по кнопке «Добавить запись».

Если Вы желаете удалить любую введенную ранее информацию, Вам необходимо кликнуть по нужной кнопке.

Для того чтобы изменить запись, введите в редактируемые поля данные и нажмите кнопку сохранить. Данные формы редактированию и управлению со стороны пользователя не подлежат.

После того, как Вы закончите сеанс работы с программой, если Вы хотите выйти из BookStoreApp кликните на соответствующей кнопке «Выход».

Основные кнопки BookStoreApp

Кнопки управления «Добавить», «Удалить запись», «Сохранить» расположены на каждой конкретной форме приложения.

Действия этих кнопок приведены в таблице 1:

Таблица 1 - Действие кнопок

|  |  |
| --- | --- |
| Добавить | После нажатия на эту кнопку выйдет вспомогательная форма для добавления записей. |
| Удалить | Кнопка «Удалить запись» позволяет удалить выделенную запись. |
| Сохранить | Кнопка «Сохранить данные» добавляет запись введенную пользователем в БД, либо обновляет существующую. |
| Изменить | Открывает форму в режиме редактирования информации. |
| Назад | Переход на предыдущую страницу. |
| Поиск | Поиск. |
| Очистить | Очистить поле ввода. |
| Excel | Выгрузить информацию в Excel. |

**4.4 Сообщения пользователю**

Для уверенной и комфортной работы пользователю важно понимать поведение программы, а программе, в свою очередь, поведение пользователя. Для информирования и уточнения намерений пользователей в программах используются сообщения.

При попытке удалить запись, пользователю выводится сообщение с запросом на подтверждение или отмену действия (рис. 59).

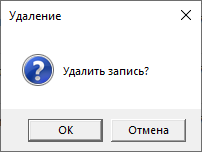


Рис. 59 Сообщение «Удаление»

После добавления или изменения записи, пользователю выводится сообщение (рис. 60).

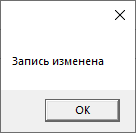


Рис. 60 Сообщение «Запись изменена»

При попытке удалить связанные записи, программа выводит пользователю сообщение о невозможности действия с указанием зависимой таблицы (рис. 61).

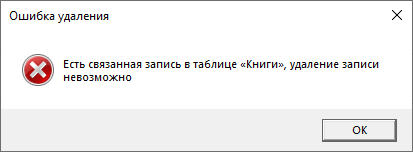


Рис. 61 Сообщение «Ошибка удаления»

**5 РАЗРАБОТКА ТЕСТОВОГО НАБОРА**

**5.1 Определение верхней границы количества тестов**

Для разработанного алгоритма был построен граф.

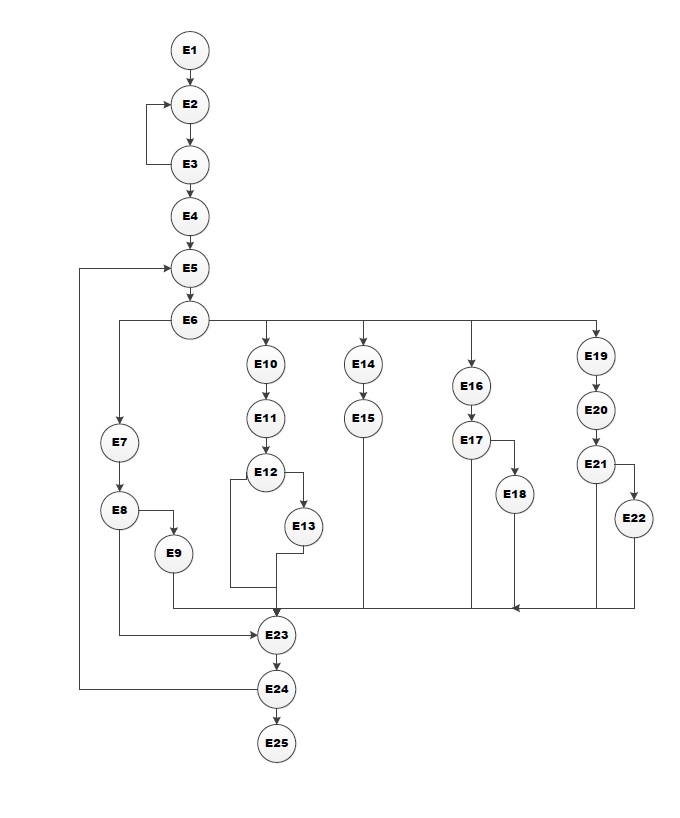
****

Рис. 62 Потоковый граф

Необходимое количество тестов определяется по формуле V(G) = E - N + 2, где E – это число ребер, а N – это число вершин данного графа. Согласно представленному графу E = 34, N = 25. Далее по формуле рассчитаем необходимое количество тестовых сценариев (тест-кейсов).

V(G) = 34 – 25 + 2 = 11.

Для данного программного средства необходимо разработать не менее 11 тестовых сценариев.

Чтобы определить количество тестовых прогонов для программного продукта BookStoreApp необходимо выделить требования. Из требований к приложению выделим поддержку 2-х операционных систем с одним основным языком локализации и выполнения 6-ти основных функций. Кроме того, приложение позволяет выполнять 5 функций по взаимодействию с системой (запуск приложения, выход из приложения, сохранение результатов в файл, работа с буфером и т.п.).

Таким образом, полное покрытие требований задаёт набор из 2 \* 1 \* (6 + 5) = 22 тестовых прогонов.

**5.2 Описание тестовых пакетов**

Таблица 2 – Тестирование корректного добавления информации в Книги

|  |  |  |  |
| --- | --- | --- | --- |
| **Название:** | Тестирование корректного добавления информации в таблицу Книги | | |
| **Функция:** | Циклы | | |
| **Действие** | | **Ожидаемый результат** | **Результат теста:**   * пройден * провален * заблокирован |
| **Предусловие:** | |  |  |
| Открываем страницу Книги | | Страница открыта |  |
| **Шаги теста:** | |  |  |
| Нажмите кнопку добавить  Заполните форму  Название = 1  Описание = 1  Жанр книги = Эпопея  Поставщик = ООО "АСТ"  Издательство = АСТ  Автор = Достоевский Федор Михайлович  ISBN = 1  Количество = 1  Цена = 1 | | Данные успешно введены |  |
| Нажмите кнопку «Сохранить» | | Данные успешно сохранены | **Тест пройден** |
| **Постусловие:** | |  |  |
| Страница Книги открыта | | Отображается страница Книги |  |

Таблица 3 – Тестирование некорректного добавления информации в таблицу Книги

|  |  |  |  |
| --- | --- | --- | --- |
| **Название:** | Тестирование некорректного добавления информации в таблицу Книги | | |
| **Функция:** | Циклы | | |
| **Действие** | | **Ожидаемый результат** | **Результат теста:**   * пройден * провален * заблокирован |
| **Предусловие:** | |  |  |
| Открываем страницу Книги | | Страница открыта |  |
| **Шаги теста:** | |  |  |
| Нажмите кнопку добавить  Заполните форму  Название =  Описание = 1  Жанр книги = Эпопея  Поставщик = ООО "АСТ"  Издательство = АСТ  Автор = Достоевский Федор Михайлович  ISBN = 1  Количество = 1  Цена = 1 | | Данные успешно введены |  |
| Нажмите кнопку «Сохранить» | | Выведено сообщение о пустом заполненном поле | **Тест пройден** |
| **Постусловие:** | |  |  |
| Страница Книги открыта | | Отображается страница Книги |  |

Таблица 4 – Тестирование корректного редактирования информации в таблице Книги

|  |  |  |  |
| --- | --- | --- | --- |
| **Название:** | Тестирование корректного редактирования информации в таблице Книги | | |
| **Функция:** | Циклы | | |
| **Действие** | | **Ожидаемый результат** | **Результат теста:**   * пройден * провален * заблокирован |
| **Предусловие:** | |  |  |
| Открываем страницу Книги | | Страница открыта |  |
| **Шаги теста:** | |  |  |
| Выберите запись для редактирования и измените данные  Название = 2  Описание = 1  Жанр книги = Эпопея  Поставщик = ООО "АСТ"  Издательство = АСТ  Автор = Достоевский Федор Михайлович  ISBN = 1  Количество = 1  Цена = 1 | | Данные успешно введены |  |
| Нажмите кнопку «Сохранить» | | Данные успешно сохранены | **Тест пройден** |
| **Постусловие:** | |  |  |
| Страница Книги открыта | | Отображается страница Книги |  |

Таблица 5 – Тестирование некорректного редактирования информации в таблице Книги

|  |  |  |  |
| --- | --- | --- | --- |
| **Название:** | Тестирование некорректного редактирования информации в таблице Книги | | |
| **Функция:** | Циклы | | |
| **Действие** | | **Ожидаемый результат** | **Результат теста:**   * пройден * провален * заблокирован |
| **Предусловие:** | |  |  |
| Открываем страницу Книги | | Страница открыта |  |
| **Шаги теста:** | |  |  |
| Выберите запись для редактирования и измените данные  Название = 1  Описание = 1  Жанр книги = Эпопея  Поставщик = ООО "АСТ"  Издательство = АСТ  Автор = Достоевский Федор Михайлович  ISBN = 1  Количество = 1  Цена = | | Данные успешно введены |  |
| Нажмите кнопку «Сохранить» | | Выведено сообщение о пустом заполненном поле | **Тест пройден** |
| **Постусловие:** | |  |  |
| Страница Книги открыта | | Отображается страница Книги |  |

Таблица 6 – Тестирование корректного добавления информации в Клиенты

|  |  |  |  |
| --- | --- | --- | --- |
| **Название:** | Тестирование корректного добавления информации в таблицу Клиенты | | |
| **Функция:** | Циклы | | |
| **Действие** | | **Ожидаемый результат** | **Результат теста:**   * пройден * провален * заблокирован |
| **Предусловие:** | |  |  |
| Открываем страницу Клиенты | | Страница открыта |  |

|  |  |  |
| --- | --- | --- |
| **Шаги теста:** |  |  |
| Нажмите кнопку добавить  Заполните форму  Логин = 4  Пароль = 4  Фамилия = 4  Имя = 4  Отчество = 4  Адрес = 4  Номер телефона = 4  Роль = Пользователь | Данные успешно введены |  |
| Нажмите кнопку «Сохранить» | Данные успешно сохранены | **Тест пройден** |
| **Постусловие:** |  |  |
| Страница Клиенты открыта | Отображается страница Клиенты |  |

Таблица 7 – Тестирование некорректного добавления информации в таблицу Клиенты

|  |  |  |  |
| --- | --- | --- | --- |
| **Название:** | Тестирование некорректного добавления информации в таблицу Клиенты | | |
| **Функция:** | Циклы | | |
| **Действие** | | **Ожидаемый результат** | **Результат теста:**   * пройден * провален * заблокирован |
| **Предусловие:** | |  |  |
| Открываем страницу Клиенты | | Страница открыта |  |
| **Шаги теста:** | |  |  |
| Нажмите кнопку добавить  Заполните форму  Логин =  Пароль = 4  Фамилия = 4  Имя = 4  Отчество = 4  Адрес = 4  Номер телефона = 4  Роль = Пользователь | | Данные успешно введены |  |
| Нажмите кнопку «Сохранить» | | Выведено сообщение о пустом заполненном поле | **Тест пройден** |
| **Постусловие:** | |  |  |
| Страница Клиенты открыта | | Отображается страница Клиенты |  |

Таблица 8 – Тестирование корректного редактирования информации в таблице Клиенты

|  |  |  |  |
| --- | --- | --- | --- |
| **Название:** | Тестирование корректного редактирования информации в таблице Клиенты | | |
| **Функция:** | Циклы | | |
| **Действие** | | **Ожидаемый результат** | **Результат теста:**   * пройден * провален * заблокирован |
| **Предусловие:** | |  |  |
| Открываем страницу Клиенты | | Страница открыта |  |
| **Шаги теста:** | |  |  |
| Выберите запись для редактирования и измените данные  Логин = 5  Пароль = 4  Фамилия = 4  Имя = 4  Отчество = 4  Адрес = 4  Номер телефона = 4  Роль = Пользователь | | Данные успешно введены |  |
| Нажмите кнопку «Сохранить» | | Данные успешно сохранены | **Тест пройден** |
| **Постусловие:** | |  |  |
| Страница Клиенты открыта | | Отображается страница Клиенты |  |

Таблица 9 – Тестирование некорректного редактирования информации в таблице Клиенты

|  |  |  |  |
| --- | --- | --- | --- |
| **Название:** | Тестирование некорректного редактирования информации в таблице Клиенты | | |
| **Функция:** | Циклы | | |
| **Действие** | | **Ожидаемый результат** | **Результат теста:**   * пройден * провален * заблокирован |
| **Предусловие:** | |  |  |
| Открываем страницу Клиенты | | Страница открыта |  |
| **Шаги теста:** | |  |  |
| Выберите запись для редактирования и измените данные  Логин = 5  Пароль =  Фамилия = 4  Имя = 4  Отчество = 4  Адрес = 4  Номер телефона = 4  Роль = Пользователь | | Данные успешно введены |  |
| Нажмите кнопку «Сохранить» | | Выведено сообщение о пустом заполненном поле | **Тест пройден** |
| **Постусловие:** | |  |  |
| Страница Клиенты открыта | | Отображается страница Клиенты |  |

Таблица 10 – Тестирование корректного добавления информации в Заказы

|  |  |  |  |
| --- | --- | --- | --- |
| **Название:** | Тестирование корректного добавления информации в таблицу Заказы | | |
| **Функция:** | Циклы | | |
| **Действие** | | **Ожидаемый результат** | **Результат теста:**   * пройден * провален * заблокирован |
| **Предусловие:** | |  |  |
| Открываем страницу Заказы | | Страница открыта |  |
| **Шаги теста:** | |  |  |
| Нажмите кнопку добавить  Заполните форму  Клиент = Иванов Иван Иванович  Книга = Война и мир  Количество = 1  Дата создания = 01.01.2001  Дата доставки = 01.01.2001  Статус = Новый | | Данные успешно введены |  |
| Нажмите кнопку «Сохранить» | | Данные успешно сохранены | **Тест пройден** |
| **Постусловие:** | |  |  |
| Страница Заказы открыта | | Отображается страница Заказы |  |

Таблица 11 – Тестирование некорректного добавления информации в таблицу Заказы

|  |  |  |  |
| --- | --- | --- | --- |
| **Название:** | Тестирование некорректного добавления информации в таблицу Заказы | | |
| **Функция:** | Циклы | | |
| **Действие** | | **Ожидаемый результат** | **Результат теста:**   * пройден * провален * заблокирован |
| **Предусловие:** | |  |  |
| Открываем страницу Заказы | | Страница открыта |  |
| **Шаги теста:** | |  |  |
| Нажмите кнопку добавить  Заполните форму  Клиент =  Книга = Война и мир  Количество = 1  Дата создания = 01.01.2001  Дата доставки = 01.01.2001  Статус = Новый | | Данные успешно введены |  |
| Нажмите кнопку «Сохранить» | | Выведено сообщение о пустом заполненном поле | **Тест пройден** |

|  |  |  |
| --- | --- | --- |
| **Постусловие:** |  |  |
| Страница Заказы открыта | Отображается страница Заказы |  |

Таблица 12 – Тестирование корректного редактирования информации в таблице Заказы

|  |  |  |  |
| --- | --- | --- | --- |
| **Название:** | Тестирование корректного редактирования информации в таблице Заказы | | |
| **Функция:** | Циклы | | |
| **Действие** | | **Ожидаемый результат** | **Результат теста:**   * пройден * провален * заблокирован |
| **Предусловие:** | |  |  |
| Открываем страницу Заказы | | Страница открыта |  |
| **Шаги теста:** | |  |  |
| Выберите запись для редактирования и измените данные  Клиент = Иванов Иван Иванович  Книга = Война и мир  Количество = 2  Дата создания = 01.01.2001  Дата доставки = 01.01.2001  Статус = Новый | | Данные успешно введены |  |
| Нажмите кнопку «Сохранить» | | Данные успешно сохранены | **Тест пройден** |
| **Постусловие:** | |  |  |
| Страница Заказы открыта | | Отображается страница Заказы |  |

Таблица 13 – Тестирование некорректного редактирования информации в таблице Заказы

|  |  |  |  |
| --- | --- | --- | --- |
| **Название:** | Тестирование некорректного редактирования информации в таблице Заказы | | |
| **Функция:** | Циклы | | |
| **Действие** | | **Ожидаемый результат** | **Результат теста:**   * пройден * провален * заблокирован |
| **Предусловие:** | |  |  |
| Открываем страницу Заказы | | Страница открыта |  |
| **Шаги теста:** | |  |  |
| Выберите запись для редактирования и измените данные  Клиент = Иванов Иван Иванович  Книга = Война и мир  Количество = 1  Дата создания = 01.01.2001  Дата доставки = 01.01.2001  Статус = | | Данные успешно введены |  |
| Нажмите кнопку «Сохранить» | | Выведено сообщение о пустом заполненном поле | **Тест пройден** |
| **Постусловие:** | |  |  |
| Страница Заказы открыта | | Отображается страница Заказы |  |

Таблица 14 – Тестирование корректного добавления информации в Издательства

|  |  |  |  |
| --- | --- | --- | --- |
| **Название:** | Тестирование корректного добавления информации в таблицу Издательства | | |
| **Функция:** | Циклы | | |
| **Действие** | | **Ожидаемый результат** | **Результат теста:**   * пройден * провален * заблокирован |
| **Предусловие:** | |  |  |
| Открываем страницу Издательства | | Страница открыта |  |
| **Шаги теста:** | |  |  |
| Нажмите кнопку добавить  Заполните форму  Название = 2  Описание = | | Данные успешно введены |  |
| Нажмите кнопку «Сохранить» | | Данные успешно сохранены | **Тест пройден** |
| **Постусловие:** | |  |  |
| Страница Издательства открыта | | Отображается страница Издательства |  |

Таблица 15 – Тестирование некорректного добавления информации в таблицу Издательства

|  |  |  |  |
| --- | --- | --- | --- |
| **Название:** | Тестирование некорректного добавления информации в таблицу Издательства | | |
| **Функция:** | Циклы | | |
| **Действие** | | **Ожидаемый результат** | **Результат теста:**   * пройден * провален * заблокирован |
| **Предусловие:** | |  |  |
| Открываем страницу Издательства | | Страница открыта |  |
| **Шаги теста:** | |  |  |
| Нажмите кнопку добавить  Заполните форму  Название =  Описание = | | Данные успешно введены |  |
| Нажмите кнопку «Сохранить» | | Выведено сообщение о пустом заполненном поле | **Тест пройден** |
| **Постусловие:** | |  |  |
| Страница Издательства открыта | | Отображается страница Издательства |  |

Таблица 16 – Тестирование корректного редактирования информации в таблице Издательства

|  |  |  |  |
| --- | --- | --- | --- |
| **Название:** | Тестирование корректного редактирования информации в таблице Издательства | | |
| **Функция:** | Циклы | | |
| **Действие** | | **Ожидаемый результат** | **Результат теста:**   * пройден * провален * заблокирован |
| **Предусловие:** | |  |  |
| Открываем страницу Издательства | | Страница открыта |  |
| **Шаги теста:** | |  |  |
| Выберите запись для редактирования и измените данные  Название = 3  Описание = | | Данные успешно введены |  |
| Нажмите кнопку «Сохранить» | | Данные успешно сохранены | **Тест пройден** |
| **Постусловие:** | |  |  |
| Страница Издательства открыта | | Отображается страница Издательства |  |

Таблица 17 – Тестирование некорректного редактирования информации в таблице Издательства

|  |  |  |  |
| --- | --- | --- | --- |
| **Название:** | Тестирование некорректного редактирования информации в таблице Издательства | | |
| **Функция:** | Циклы | | |
| **Действие** | | **Ожидаемый результат** | **Результат теста:**   * пройден * провален * заблокирован |
| **Предусловие:** | |  |  |
| Открываем страницу Издательства | | Страница открыта |  |
| **Шаги теста:** | |  |  |
| Выберите запись для редактирования и измените данные  Название =  Описание = | | Данные успешно введены |  |
| Нажмите кнопку «Сохранить» | | Выведено сообщение о пустом заполненном поле | **Тест пройден** |
| **Постусловие:** | |  |  |
| Страница Издательства открыта | | Отображается страница Издательства |  |

Таблица 18 – Тестирование корректного добавления информации в Поставщики

|  |  |  |  |
| --- | --- | --- | --- |
| **Название:** | Тестирование корректного добавления информации в таблицу Поставщики | | |
| **Функция:** | Циклы | | |
| **Действие** | | **Ожидаемый результат** | **Результат теста:**   * пройден * провален * заблокирован |
| **Предусловие:** | |  |  |
| Открываем страницу Поставщики | | Страница открыта |  |
| **Шаги теста:** | |  |  |
| Нажмите кнопку добавить  Заполните форму  Название = 2  Адрес = 2  Номер телефона = 2 | | Данные успешно введены |  |
| Нажмите кнопку «Сохранить» | | Данные успешно сохранены | **Тест пройден** |
| **Постусловие:** | |  |  |
| Страница Поставщики открыта | | Отображается страница Поставщики |  |

Таблица 19 – Тестирование некорректного добавления информации в таблицу Поставщики

|  |  |  |  |
| --- | --- | --- | --- |
| **Название:** | Тестирование некорректного добавления информации в таблицу Поставщики | | |
| **Функция:** | Циклы | | |
| **Действие** | | **Ожидаемый результат** | **Результат теста:**   * пройден * провален * заблокирован |
| **Предусловие:** | |  |  |
| Открываем страницу Поставщики | | Страница открыта |  |
| **Шаги теста:** | |  |  |
| Нажмите кнопку добавить  Заполните форму  Название =  Адрес = 2  Номер телефона = 2 | | Данные успешно введены |  |
| Нажмите кнопку «Сохранить» | | Выведено сообщение о пустом заполненном поле | **Тест пройден** |
| **Постусловие:** | |  |  |
| Страница Поставщики открыта | | Отображается страница Поставщики |  |

Таблица 20 – Тестирование корректного редактирования информации в таблице Поставщики

|  |  |  |  |
| --- | --- | --- | --- |
| **Название:** | Тестирование корректного редактирования информации в таблице Поставщики | | |
| **Функция:** | Циклы | | |
| **Действие** | | **Ожидаемый результат** | **Результат теста:**   * пройден * провален * заблокирован |
| **Предусловие:** | |  |  |
| Открываем страницу Поставщики | | Страница открыта |  |
| **Шаги теста:** | |  |  |
| Выберите запись для редактирования и измените данные  Название = 3  Адрес = 2  Номер телефона = 2 | | Данные успешно введены |  |
| Нажмите кнопку «Сохранить» | | Данные успешно сохранены | **Тест пройден** |
| **Постусловие:** | |  |  |
| Страница Поставщики открыта | | Отображается страница Поставщики |  |

Таблица 21 – Тестирование некорректного редактирования информации в таблице Поставщики

|  |  |  |  |
| --- | --- | --- | --- |
| **Название:** | Тестирование некорректного редактирования информации в таблице Поставщики | | |
| **Функция:** | Циклы | | |
| **Действие** | | **Ожидаемый результат** | **Результат теста:**   * пройден * провален * заблокирован |
| **Предусловие:** | |  |  |
| Открываем страницу Поставщики | | Страница открыта |  |
| **Шаги теста:** | |  |  |
| Выберите запись для редактирования и измените данные  Название =  Адрес = 2  Номер телефона = 2 | | Данные успешно введены |  |
| Нажмите кнопку «Сохранить» | | Выведено сообщение о пустом заполненном поле | **Тест пройден** |
| **Постусловие:** | |  |  |
| Страница Поставщики открыта | | Отображается страница Поставщики |  |

Таблица 22 – Тестирование корректного добавления информации в Жанры книг

|  |  |  |  |
| --- | --- | --- | --- |
| **Название:** | Тестирование корректного добавления информации в таблицу Жанры книг | | |
| **Функция:** | Циклы | | |
| **Действие** | | **Ожидаемый результат** | **Результат теста:**   * пройден * провален * заблокирован |
| **Предусловие:** | |  |  |
| Открываем страницу Жанры книг | | Страница открыта |  |
| **Шаги теста:** | |  |  |
| Нажмите кнопку добавить  Заполните форму  Название = 2  Описание = 2 | | Данные успешно введены |  |
| Нажмите кнопку «Сохранить» | | Данные успешно сохранены | **Тест пройден** |
| **Постусловие:** | |  |  |
| Страница Жанры книг открыта | | Отображается страница Жанры книг |  |

Таблица 23 – Тестирование некорректного добавления информации в таблицу Жанры книг

|  |  |  |  |
| --- | --- | --- | --- |
| **Название:** | Тестирование некорректного добавления информации в таблицу Жанры книг | | |
| **Функция:** | Циклы | | |
| **Действие** | | **Ожидаемый результат** | **Результат теста:**   * пройден * провален * заблокирован |
| **Предусловие:** | |  |  |
| Открываем страницу Жанры книг | | Страница открыта |  |
| **Шаги теста:** | |  |  |
| Нажмите кнопку добавить  Заполните форму  Название =  Описание = 2 | | Данные успешно введены |  |
| Нажмите кнопку «Сохранить» | | Выведено сообщение о пустом заполненном поле | **Тест пройден** |
| **Постусловие:** | |  |  |
| Страница Жанры книг открыта | | Отображается страница Жанры книг |  |

Таблица 24 – Тестирование корректного редактирования информации в таблице Жанры книг

|  |  |  |  |
| --- | --- | --- | --- |
| **Название:** | Тестирование корректного редактирования информации в таблице Жанры книг | | |
| **Функция:** | Циклы | | |
| **Действие** | | **Ожидаемый результат** | **Результат теста:**   * пройден * провален * заблокирован |
| **Предусловие:** | |  |  |
| Открываем страницу Жанры книг | | Страница открыта |  |
| **Шаги теста:** | |  |  |
| Выберите запись для редактирования и измените данные  Название = 3  Описание = 3 | | Данные успешно введены |  |
| Нажмите кнопку «Сохранить» | | Данные успешно сохранены | **Тест пройден** |
| **Постусловие:** | |  |  |
| Страница Жанры книг открыта | | Отображается страница Жанры книг |  |

Таблица 25 – Тестирование некорректного редактирования информации в таблице Жанры книг

|  |  |  |  |
| --- | --- | --- | --- |
| **Название:** | Тестирование некорректного редактирования информации в таблице Жанры книг | | |
| **Функция:** | Циклы | | |
| **Действие** | | **Ожидаемый результат** | **Результат теста:**   * пройден * провален * заблокирован |
| **Предусловие:** | |  |  |
| Открываем страницу Жанры книг | | Страница открыта |  |
| **Шаги теста:** | |  |  |
| Выберите запись для редактирования и измените данные  Название =  Описание = 3 | | Данные успешно введены |  |
| Нажмите кнопку «Сохранить» | | Выведено сообщение о пустом заполненном поле | **Тест пройден** |
| **Постусловие:** | |  |  |
| Страница Жанры книг открыта | | Отображается страница Жанры книг |  |

Таблица 26 – Тестирование корректного добавления информации в Авторы

|  |  |  |  |
| --- | --- | --- | --- |
| **Название:** | Тестирование корректного добавления информации в таблицу Авторы | | |
| **Функция:** | Циклы | | |
| **Действие** | | **Ожидаемый результат** | **Результат теста:**   * пройден * провален * заблокирован |
| **Предусловие:** | |  |  |
| Открываем страницу Авторы | | Страница открыта |  |
| **Шаги теста:** | |  |  |
| Нажмите кнопку добавить  Заполните форму  Фамилия = 6  Имя = 6  Отчество = 6  Дата рождения = 01.01.2006 | | Данные успешно введены |  |
| Нажмите кнопку «Сохранить» | | Данные успешно сохранены | **Тест пройден** |
| **Постусловие:** | |  |  |
| Страница Авторы открыта | | Отображается страница Авторы |  |

Таблица 27 – Тестирование некорректного добавления информации в таблицу Авторы

|  |  |  |  |
| --- | --- | --- | --- |
| **Название:** | Тестирование некорректного добавления информации в таблицу Авторы | | |
| **Функция:** | Циклы | | |
| **Действие** | | **Ожидаемый результат** | **Результат теста:**   * пройден * провален * заблокирован |
| **Предусловие:** | |  |  |
| Открываем страницу Авторы | | Страница открыта |  |
| **Шаги теста:** | |  |  |
| Нажмите кнопку добавить  Заполните форму  Фамилия =  Имя = 6  Отчество = 6  Дата рождения = 01.01.2006 | | Данные успешно введены |  |
| Нажмите кнопку «Сохранить» | | Выведено сообщение о пустом заполненном поле | **Тест пройден** |
| **Постусловие:** | |  |  |
| Страница Авторы открыта | | Отображается страница Авторы |  |

Таблица 28 – Тестирование корректного редактирования информации в таблице Авторы

|  |  |  |  |
| --- | --- | --- | --- |
| **Название:** | Тестирование корректного редактирования информации в таблице Авторы | | |
| **Функция:** | Циклы | | |
| **Действие** | | **Ожидаемый результат** | **Результат теста:**   * пройден * провален * заблокирован |
| **Предусловие:** | |  |  |
| Открываем страницу Авторы | | Страница открыта |  |
| **Шаги теста:** | |  |  |
| Выберите запись для редактирования и измените данные  Фамилия = 7  Имя = 7  Отчество = 7  Дата рождения = 01.01.2007 | | Данные успешно введены |  |
| Нажмите кнопку «Сохранить» | | Данные успешно сохранены | **Тест пройден** |
| **Постусловие:** | |  |  |
| Страница Авторы открыта | | Отображается страница Авторы |  |

Таблица 29 – Тестирование некорректного редактирования информации в таблице Авторы

|  |  |  |  |
| --- | --- | --- | --- |
| **Название:** | Тестирование некорректного редактирования информации в таблице Авторы | | |
| **Функция:** | Циклы | | |
| **Действие** | | **Ожидаемый результат** | **Результат теста:**   * пройден * провален * заблокирован |
| **Предусловие:** | |  |  |
| Открываем страницу Авторы | | Страница открыта |  |
| **Шаги теста:** | |  |  |
| Выберите запись для редактирования и измените данные  Фамилия =  Имя = 7  Отчество = 7  Дата рождения = 01.01.2007 | | Данные успешно введены |  |
| Нажмите кнопку «Сохранить» | | Выведено сообщение о пустом заполненном поле | **Тест пройден** |
| **Постусловие:** | |  |  |
| Страница Авторы открыта | | Отображается страница Авторы |  |

**5.3 Составление отчета о тестировании**

Для форм проекта были созданы и проведены тесты, результаты сведены в таблицу:

Таблица 30 – Результаты тестирования

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | Прохождение | | |
| Форма | Число ТП | пройдено | | не пройдено |
| Книги | 4 | 4 | | 0 |
| Клиенты | 4 | 4 | | 0 |
| Заказы | 4 | 4 | | 0 |
| Издательства | 4 | 4 | | 0 |
| Поставщики | 4 | 4 | | 0 |
| Жанры | 4 | 4 | | 0 |
| Авторы | 4 | 4 | | 0 |
| ИТОГО | 24 | 24 | 0 | |

**ЗАКЛЮЧЕНИЕ**

**Оценка качества программного средства с помощью метрик**

Оценка качества программного средства проводится исходя из стандарта оценки качества ISО 9126.

На компьютере следующей конфигурации:

Установленная операционная система – Windows 10 Professional.

Процессор: Intel Pentium G2030 3.0 GHz.

ОЗУ: 4 Gb

Таблица 31 – Требования к количественным характеристикам качества программного средства

|  |  |  |  |
| --- | --- | --- | --- |
| **Характеристики качества** | **Мера** | **Требуемое значение** | **Реальное значение** |
| **Надежность** |  | | |
| *Завершенность:*  *-* наработка на отказ при отсутствии рестарта; | Часы | 10 | 10 |
| *Устойчивость:*  - относительные ресурсы на обеспечение надежности и рестарта. | % | 0,5 | 0,5 |
| *Восстанавливаемость:*  - длительность восстановления. | Минуты | 2 | 2 |
| *Доступность-готовность:*  - относительное время работоспособного функционирования. | Вероятность | 0,998 | 0,998 |
| **Эффективность** |  | | |
| *Временная эффективность:*  *-* время отклика - получение результатов на типовое задание;  - пропускная способность число типовых заданий, исполняемых в единицу времени. | Секунды  Число в минуту | 3  60 | 3  60 |
| *Используемость ресурсов:*  *-* относительная величина использования ресурсов ЭВМ при нормальном функционировании программного средства. |  | 0,001 | 0,001 |

Таблица 32 – Требования к качественным характеристикам программного средства.

|  |  |  |  |
| --- | --- | --- | --- |
| **Характеристики качества** | **Мера** | **Требуемое значение** | **Реальное значение** |
| **Практичность** |  | | |
| *Простота использования:*  -среднее время ввода заданий;  - среднее время отклика на задание. | Секунды  Секунды | 10  3 | 10  3 |
| *Изучаемость:*  - трудоемкость изучения применения ПС;  - продолжительность изучения;  - объем эксплуатационной документации; | Чел. - часы  Часы  Страницы | 5  2  15 | 5  2  15 |
| **Сопровождаемость** |  | | |
| *Изменяемость:*  - трудоемкость подготовки изменений;  - длительность подготовки изменений. | Чел.-часы  Часы | 5  5 | 5  5 |
| *Тестируемость:*  - трудоемкость тестирования изменений;  - длительность тестирования изменений. | Чел. - часы  Часы | 3  3 | 3  3 |
| **Мобильность** |  | | |
| *Адаптируемость:*  - трудоемкость адаптации;  - длительность адаптации. | Чел. - часы  Часы | 0,2  1 | 0,2  1 |
| *Простота установки:*  - трудоемкость инсталляции;  - длительность инсталляции. | Чел. - часы  Часы | 0,2  1 | 0,2  1 |
| *Замещаемость:*  - трудоемкость замены компонентов;  - длительность замены компонентов. | Чел. - часы  Часы | 0,2  0,2 | 0,2  0,2 |

Таблица 33 – Сложность

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Тип объекта | Количество | Вес | | | Итого |
|  |  | Простой | Средний | Сложный |  |
| Экран | 17 | x1 | x2 | **x3** | = 51 |
| Отчет | 3 | x2 | x5 | **x8** | = 24 |

Таблица 34 – Сложность экрана

|  |  |  |  |
| --- | --- | --- | --- |
| Экраны | Количество клиентских (клт) таблиц данных | | |
| Количество представлений | Всего < 4 | Всего < 8 | **Всего > 8** |
|  | (< 2 срв, < 3 клт) | (2 - 3 срв, 3 - 5 клт) | (> 3 срв, > 5 клт) |
| < 3 | Простой | Простой | Средний |
| 3 - 7 | Простой | Средний | Сложный |
| **> 8** | Средний | Сложный | **Сложный** |

Таблица 35 – Сложность отчета

|  |  |  |  |
| --- | --- | --- | --- |
| Отчеты | Количество Тип клиентаских (клт) таблиц данных | | |
| Количество представлений | Всего < 4 | Всего < 8 | Всего > 8 |
|  | (< 2 срв, < 3 клт) | (2 - 3 срв, 3 - 5 клт) | **(> 3 срв, > 5 клт)** |
| 0 или 1 | Простой | Простой | Средний |
| **2 или 3** | Простой | Средний | **Сложный** |
| > 4 | Средний | Сложный | Сложный |

Производительность = 5000 тысяч строк кода / 1 месяца = 5000 тысяч строк кода / месяц

Качество = 25 ошибок / 5 тысяч строк кода = 5 ошибок / тысяч строк кода

**СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

1. Г. Шилдт C#: Учебный курс. - Спб.: Питер, 2020. - 512с

2. Либерти, Д. Программирование на Си Шарп. - М.: Символ-плюс, 2018. - 684 с.

3. Ватсон К. Си Шарп/К.Ватсон.-М.:Лори, 2020.-862 с

4. Балена Ф., Димауро Д. Современная практика программирования на Visual Basic и Visual C#.-М.:Русская редакция,2020,604с.

5. Енин А., Енин Н. Локальная СУБД своими руками. Учимся на примерах.-М.:СОЛОН-ПРЕСС,2020,464с.,СД

6. Лабор В.В. Си Шарп: СоТип клиента приложений для Windows/-Мн.: Харвест, 2018-384с.

7. Фролов А.В., Фролов Г.В. Визуальное проектирование приложений C#.-М.:КУДИЦ-ОБРАЗ, 2020-512с.

8. ГОСТ 19.201-78.ЕСПД. Техническое задание. Требования к содержанию и оформлению

9. ГОСТ 19.102-77.ЕСПД. Стадии разработки

10. ГОСТ 19.505-79.ЕСПД. Руководство оператора. Требования к содержанию и оформлению

11. ГОСТ 19.002-80.ЕСПД. Схемы алгоритмов и программ. Правила выполнения

12. ГОСТ 19.404-79.ЕСПД. Пояснительная записка. Требования к содержанию и оформлению

13. Владимир Биллинг. Основы программирования на С#. 2018 г.

[Электронный ресурс http://mexalib.com/view/42671]

14. Никита Культин. Основы программирования в MicrosoftVisualC# 2020. 2020 г.

[Электронный ресурс http://www.litres.ru/nikita-kultin/osnovy-programmirovaniya-v-microsoft-visual-c-2010-2]

15. Фролов А.В., Фролов Г.В. Визуальное проектирование приложений C#.-М.:КУДИЦ-ОБРАЗ, 2020.

[Электронный ресурс http://www.twirpx.com/file/27852]

16. Йен Гриффитс. Программирование на С#. 2021 г.

[Электронный ресурс http://rutracker.org/forum/viewtopic.php?t=4772853]

**Приложение А**

Автоматизированная информационная система «Мелкооптовый книжный магазин»

Текст кода программного средства

460.ЗМК.00004-01 12 01

**Текст кода программного средства**

**App.config**

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

<configuration>

<configSections>

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

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

</configSections>

<startup>

<supportedRuntime version="v4.0" sku=".NETFramework,Version=v4.7.2" />

</startup>

<connectionStrings>

<add name="BookStoreEntities" connectionString="metadata=res://\*/Models.BookStoreModel.csdl|res://\*/Models.BookStoreModel.ssdl|res://\*/Models.BookStoreModel.msl;provider=System.Data.SqlClient;provider connection string=&quot;data source=(LocalDB)\MSSQLLocalDB;attachdbfilename=|DataDirectory|\BookStore.mdf;integrated security=True;connect timeout=30;MultipleActiveResultSets=True;App=EntityFramework&quot;" providerName="System.Data.EntityClient" />

</connectionStrings>

<entityFramework>

<defaultConnectionFactory type="System.Data.Entity.Infrastructure.LocalDbConnectionFactory, EntityFramework">

<parameters>

<parameter value="mssqllocaldb" />

</parameters>

</defaultConnectionFactory>

<providers>

<provider invariantName="System.Data.SqlClient" type="System.Data.Entity.SqlServer.SqlProviderServices, EntityFramework.SqlServer" />

</providers>

</entityFramework>

</configuration>

**App.xaml**

<Application x:Class="BookStoreApp.App"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:local="clr-namespace:BookStoreApp"

StartupUri="Windows/LoginWindow.xaml">

<Application.Resources>

<ResourceDictionary>

<ResourceDictionary.MergedDictionaries>

<ResourceDictionary Source="pack://application:,,,/MaterialDesignThemes.Wpf;component/Themes/MaterialDesignTheme.Button.xaml" />

<ResourceDictionary Source="pack://application:,,,/MaterialDesignThemes.Wpf;component/Themes/MaterialDesignTheme.PopupBox.xaml" />

<ResourceDictionary Source="pack://application:,,,/MaterialDesignThemes.Wpf;component/Themes/MaterialDesignTheme.Light.xaml" />

<ResourceDictionary Source="pack://application:,,,/MaterialDesignThemes.Wpf;component/Themes/MaterialDesignTheme.Defaults.xaml" />

<ResourceDictionary Source="pack://application:,,,/MaterialDesignColors;component/Themes/Recommended/Primary/MaterialDesignColor.Amber.xaml" />

<ResourceDictionary Source="pack://application:,,,/MaterialDesignColors;component/Themes/Recommended/Accent/MaterialDesignColor.Cyan.xaml" />

<ResourceDictionary Source="Resources/Dictionary.xaml"/>

</ResourceDictionary.MergedDictionaries>

</ResourceDictionary>

</Application.Resources>

</Application>

**LoginWindow.xaml**

<Window x:Class="BookStoreApp.Windows.LoginWindow"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:local="clr-namespace:BookStoreApp.Windows"

xmlns:materialDesign="http://materialdesigninxaml.net/winfx/xaml/themes"

mc:Ignorable="d"

Title="Вход" Style="{StaticResource base\_window}" Height="230"

Width="510" MinHeight="280" MinWidth="510"

WindowStartupLocation="CenterScreen"

Closing="Window\_Closing"

ResizeMode="NoResize">

<Grid>

<Grid.ColumnDefinitions>

<ColumnDefinition />

</Grid.ColumnDefinitions>

<Grid.RowDefinitions>

<RowDefinition Height="60"/>

<RowDefinition Height="50"/>

<RowDefinition Height="50"/>

<RowDefinition Height="60"/>

</Grid.RowDefinitions>

<Grid Background="{StaticResource additional}" />

<TextBlock Grid.Row="0" Text="Мелкооптовый книжный магазин"

HorizontalAlignment="Center"

VerticalAlignment="Center"

Background="{StaticResource additional}"

FontSize="20"

Foreground="Black"/>

<StackPanel Orientation="Horizontal" Grid.Row="1" Margin="10,0,20,0">

<TextBox x:Name="TbLogin"

Width="450"

VerticalAlignment="Center" HorizontalAlignment="Center" HorizontalContentAlignment="Left"

materialDesign:HintAssist.Hint="Логин" Style="{StaticResource MaterialDesignFloatingHintTextBox}"

materialDesign:TextFieldAssist.HasClearButton="True"

Margin="0,2,0,3" Padding="0,0,0,5" />

</StackPanel>

<StackPanel Orientation="Horizontal" Grid.Row="2" Margin="10,0,0,0">

<PasswordBox x:Name="TbPass" Width="450"

materialDesign:HintAssist.Hint="Пароль"

Style="{StaticResource MaterialDesignFloatingHintPasswordBox}"

materialDesign:TextFieldAssist.HasClearButton="True"/>

</StackPanel >

<StackPanel Orientation="Horizontal" Grid.Row="3" HorizontalAlignment="Center" Margin="10,5,10,-12" Width="474">

<Button Width="70" Height="Auto"

Content="OK"

x:Name="BtnOk"

FontSize="18"

Click="BtnOk\_Click" Margin="10,10,20,10"

Style="{StaticResource MaterialDesignRaisedButton}" />

<Button Width="100"

Height="Auto"

Content="Отмена"

x:Name="BtnCancel"

FontSize="18"

Click="BtnCancel\_Click" Margin="0,10,20,10"

Style="{StaticResource MaterialDesignRaisedButton}" />

<Button Width="80"

Height="Auto"

Content="Гость"

x:Name="BtnGuestMode"

FontSize="18"

Click="BtnGuestMode\_Click" Margin="0,10,20,10"

Style="{StaticResource MaterialDesignRaisedButton}" />

<Button Width="150"

Height="Auto"

Content="Регистрация"

x:Name="BtnRegister"

FontSize="18"

Click="BtnRegister\_Click" Margin="0,10,0,10"

Style="{StaticResource MaterialDesignRaisedButton}" />

</StackPanel>

</Grid>

</Window>

**LoginWindow.xaml.cs**

using BookStoreApp.Models;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Windows;

namespace BookStoreApp.Windows

{

/// <summary>

/// Логика взаимодействия для LoginWindow.xaml

/// </summary>

public partial class LoginWindow : Window

{

public LoginWindow()

{

InitializeComponent();

TbLogin.Text = "bulatov";

TbPass.Password = "password4";

}

private void BtnOk\_Click(object sender, RoutedEventArgs e)

{

try

{

List<Client> clients = BookStoreEntities.GetContext().Clients.ToList();

Client client = clients.FirstOrDefault(p => p.ClientPassword == TbPass.Password && p.ClientLogin == TbLogin.Text);

if (client != null)

{

Manager.CurrentUser = client;

MainWindow mainWindow = new MainWindow();

mainWindow.Owner = this;

this.Hide(); // скрываем форму авторизации

mainWindow.Show();

}

else

{

MessageBox.Show("Неверный логин или пароль");

}

}

catch (Exception ex)

{

MessageBox.Show(ex.Message.ToString());

}

}

// код кнопки Cancel

private void BtnCancel\_Click(object sender, RoutedEventArgs e)

{

this.Close();

}

// попытка закрыть приложение

private void Window\_Closing(object sender, System.ComponentModel.CancelEventArgs e)

{

// на экране отображается форма с двумя кнопками

MessageBoxResult x = MessageBox.Show("Вы действительно хотите закрыть приложение?",

"Выйти", MessageBoxButton.OKCancel, MessageBoxImage.Question);

if (x == MessageBoxResult.Cancel)

e.Cancel = true;

}

private void BtnGuestMode\_Click(object sender, RoutedEventArgs e)

{

// вход в режиме гостя

Manager.CurrentUser = null;

MainWindow mainWindow = new MainWindow();

mainWindow.Owner = this;

this.Hide();

mainWindow.Show();

}

private void BtnRegister\_Click(object sender, RoutedEventArgs e)

{

// Создание нового экземпляра окна регистрации

RegisterWindow registerWindow = new RegisterWindow();

// Открытие окна регистрации

registerWindow.Show();

// Скрытие текущего окна входа

this.Hide();

}

}

}

**NewOrderWindow.xaml**

<Window x:Class="BookStoreApp.Windows.NewOrderWindow"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:local="clr-namespace:BookStoreApp.Windows"

mc:Ignorable="d"

Title="Оформление заказа" Style="{StaticResource base\_window}" Height="500"

Width="600" MinHeight="400" MinWidth="400"

WindowStartupLocation="CenterScreen">

<Grid>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="200"/>

<ColumnDefinition />

</Grid.ColumnDefinitions>

<Grid.RowDefinitions>

<RowDefinition Height="40"/>

<RowDefinition Height="34"/>

<RowDefinition Height="34"/>

<RowDefinition Height="34"/>

<RowDefinition/>

<RowDefinition Height="50"/>

<RowDefinition Height="80"/>

</Grid.RowDefinitions>

<Grid Background="{StaticResource additional}" Grid.ColumnSpan="2" />

<TextBlock Grid.Row="0" Text="Мелкооптовый книжный магазин" Grid.ColumnSpan="2"

HorizontalAlignment="Center" VerticalAlignment="Center" Background="{StaticResource additional}"

FontSize="20" Foreground="black"/>

<TextBlock Text="Заказ" Grid.Row="1" Grid.Column="0"

Grid.ColumnSpan="2" x:Name="TextBlockOrderNumber" HorizontalAlignment="Center"

VerticalAlignment="Center" FontSize="20"/>

<TextBlock Text="Дата заказа" Grid.Row="2" Grid.Column="0"

HorizontalAlignment="Left" Margin="70,2,0,2"

VerticalAlignment="Center"

Style="{StaticResource item\_textblock}" Width="120"/>

<TextBlock x:Name="TextBlockOrderCreateDate"

Grid.Row="2" Grid.Column="1"

VerticalAlignment="Center"

HorizontalAlignment="Left"

Style="{StaticResource item\_textblock}"/>

<TextBlock Text="Дата получения" Grid.Row="3" Grid.Column="0"

HorizontalAlignment="Left" Margin="30,2,0,2"

VerticalAlignment="Center"

Style="{StaticResource item\_textblock}" Width="160" />

<TextBlock x:Name="TextBlockOrderDeliveryDate" Grid.Row="3"

Grid.Column="1" VerticalAlignment="Center"

HorizontalAlignment="Left" Style="{StaticResource item\_textblock}"/>

<StackPanel Orientation="Horizontal" Grid.Row="6"/>

<ListBox x:Name="ListBoxOrderProducts"

ScrollViewer.VerticalScrollBarVisibility="Visible" SelectedValuePath="Key" Grid.Row="4"

Grid.Column="0" Grid.ColumnSpan="2"

ItemContainerStyle="{StaticResource good\_item}" >

<ListBox.ItemTemplate>

<DataTemplate>

<Grid HorizontalAlignment="Stretch" UseLayoutRounding="True">

<Grid.ColumnDefinitions>

<ColumnDefinition Width="60"/>

<ColumnDefinition Width="\*"/>

<ColumnDefinition Width="80"/>

</Grid.ColumnDefinitions>

<Image Grid.Column="0" Width="60" Height="40"

Stretch="Uniform"

Source="{Binding Path=Key.GetPhoto}"

Margin="5 0 5 0"/>

<StackPanel Grid.Column="1" >

<TextBlock Margin="2" FontSize="16"

Text="{Binding Path=Key.BookName}"

Width="Auto" HorizontalAlignment="Left" />

<TextBlock Margin="2" FontSize="14"

Text="{Binding Path=Key.BookDescription}"

Width="Auto" HorizontalAlignment="Left" />

<TextBlock Margin="2" FontSize="14"

Text="{Binding Path=Key.PublishingHouse.PublishingHouseName, StringFormat={}Издательство: {0}}"

Width="Auto" HorizontalAlignment="Left" />

<StackPanel Orientation="Horizontal">

<TextBlock Margin="2" FontSize="16"

Text="{Binding Path=Key.BookPrice, StringFormat='c', ConverterCulture='ru-RU'}"

Width="Auto" HorizontalAlignment="Left"

Foreground="Green"/>

</StackPanel>

<StackPanel Orientation="Horizontal">

<TextBlock FontSize="14" Text="Количество:"

HorizontalAlignment="Left" />

<TextBox FontSize="14" Text="{Binding Path=Value.Count}"

Width="60" HorizontalAlignment="Left"

Tag="{Binding Path=Key}"

PreviewKeyDown="TextBox\_PreviewKeyDown"

PreviewTextInput="TextBox\_PreviewTextInput" />

</StackPanel>

<TextBlock FontSize="12"

Text="{Binding Path=Key.BookCount, StringFormat={}Количество в наличии: {0} шт.}"

HorizontalAlignment="Left" />

<TextBlock FontSize="16"

Text="{Binding Path=Value.Total, StringFormat={}Итого: {0:F2} руб.}"

Width="300" HorizontalAlignment="Left" />

</StackPanel>

</Grid>

</DataTemplate>

</ListBox.ItemTemplate>

</ListBox>

<TextBlock x:Name="TextBlockTotalCost" Grid.Row="5" Grid.ColumnSpan="2"/>

<StackPanel Orientation="Horizontal" Grid.Row="6"

Grid.ColumnSpan="2" HorizontalAlignment="Center" Width="592">

<Button Margin="105,0,0,0" Width="110" Content="Закрыть" x:Name="BtnOk" Click="BtnOkClick" Height="40" Style="{StaticResource MaterialDesignRaisedButton}" FontSize="18"/>

<Button Margin="20,0,0,0" x:Name="BtnDelete" Content="Удалить" Click="BtnDeleteClick" Width="110" Height="40" Style="{StaticResource MaterialDesignRaisedButton}" FontSize="18"/>

<Button Margin="20,20,0,20" x:Name="BtnBuyItem" Content="Оформить" Width="130" Click="BtnBuyItem\_Click" Height="40" Style="{StaticResource MaterialDesignRaisedButton}" FontSize="18"/>

</StackPanel>

</Grid>

</Window>

**NewOrderWindow.xaml.cs**

using BookStoreApp.Models;

using System;

using System.Linq;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Input;

namespace BookStoreApp.Windows

{

/// <summary>

/// Логика взаимодействия для NewOrderWindow.xaml

/// </summary>

public partial class NewOrderWindow : Window

{

Client \_currentClient;

Order \_currentOrder;

public NewOrderWindow()

{

InitializeComponent();

LoadDataAndInit();

}

/// <summary>

/// Загрузка и инициализация полей

/// </summary>

void LoadDataAndInit()

{

// источник данных корзина

ListBoxOrderProducts.ItemsSource = Basket.GetBasket;

// текущий пользователь

\_currentClient = Manager.CurrentUser;

if (\_currentClient != null)

{

TextBlockOrderNumber.Text = $"Заказ на имя " +

$"{ \_currentClient.ClientSurname} {\_currentClient.ClientName} {\_currentClient.ClientPatronymic}";

}

else

{

TextBlockOrderNumber.Text = $"Заказ";

}

TextBlockTotalCost.Text = $"Итого {Basket.GetTotalCost:C}";

TextBlockOrderCreateDate.Text = DateTime.Today.ToLongDateString();

TextBlockOrderDeliveryDate.Text = DateTime.Today.AddDays(3).ToLongDateString();

TextBlockTotalCost.Text = $" Итого {Basket.GetTotalCost:C}";

}

private void BtnOkClick(object sender, RoutedEventArgs e)

{

this.Close();

}

private void BtnDeleteClick(object sender, RoutedEventArgs e)

{

// вывод сообщения с вопросом Удалить запись?

MessageBoxResult messageBoxResult = MessageBox.Show($"Удалить товар из корзины???",

"Удаление", MessageBoxButton.OKCancel, MessageBoxImage.Question);

//если пользователь нажал ОК, пытаемся удалить запись

if (messageBoxResult == MessageBoxResult.OK)

{

if (ListBoxOrderProducts.SelectedIndex >= 0)

{

var x = (ListBoxOrderProducts.SelectedValue as Book);

Basket.DeleteProductFromBasket(x);

ListBoxOrderProducts.Items.Refresh();

TextBlockTotalCost.Text = $"Итого: {Basket.GetTotalCost:C}";

}

}

}

// проверка данных в поле количество

private void TextBox\_PreviewKeyDown(object sender, KeyEventArgs e)

{

if (e.Key == Key.Return) // при нажатии на кнопку Enter

{

TextBox textBox = sender as TextBox;

int k = ListBoxOrderProducts.SelectedIndex;

Book book = textBox.Tag as Book;

if (!string.IsNullOrWhiteSpace(textBox.Text))

{

int x = 0;

if (!int.TryParse(textBox.Text, out x))

{

MessageBox.Show("Количество только число");

return;

}

x = Convert.ToInt32(textBox.Text);

if (x < 0)

{

MessageBox.Show("Количество не может быть отрицательным");

}

else if (x > book.BookCount)

{

MessageBox.Show("Количество не может быть больше, чем количество товаров на складе");

}

else if (x == 0)

{

MessageBoxResult messageBoxResult = MessageBox.Show($"Удалить {book.BookName} из корзины???",

"Удаление", MessageBoxButton.OKCancel, MessageBoxImage.Question);

if (messageBoxResult == MessageBoxResult.OK)

{

Basket.DeleteProductFromBasket(book);

ListBoxOrderProducts.Items.Refresh();

ListBoxOrderProducts.SelectedIndex = k;

}

}

else

{

Basket.SetCount(book, x);

ListBoxOrderProducts.Items.Refresh();

ListBoxOrderProducts.SelectedIndex = k;

}

}

}

if (e.Key == Key.Escape) // клавиша ESC

{

int k = ListBoxOrderProducts.SelectedIndex;

ListBoxOrderProducts.Items.Refresh();

ListBoxOrderProducts.SelectedIndex = k;

}

TextBlockTotalCost.Text = $" Итого {Basket.GetTotalCost:C}";

}

// в поле количество можно вводить только цифры

private void TextBox\_PreviewTextInput(object sender, TextCompositionEventArgs e)

{

e.Handled = !char.IsDigit(e.Text, 0);

}

// кнопка оформить покупку

private void BtnBuyItem\_Click(object sender, RoutedEventArgs e)

{

try

{

if (\_currentClient == null) { throw new Exception("В гостевом режиме нельзя сделать заказ"); }

MessageBoxResult messageBoxResult = MessageBox.Show($"Оформить покупку?",

"Оформление", MessageBoxButton.OKCancel, MessageBoxImage.Question);

if (messageBoxResult == MessageBoxResult.OK)

{

// формируем данные в таблицу Order (товары заказа)

foreach (var item in Basket.GetBasket)

{

Order order = new Order();

order.ClientID = \_currentClient.ClientID;

order.BookID = item.Key.BookID;

order.OrderCount = item.Value.Count;

order.OrderCreateDate = DateTime.Now;

order.OrderDeliveryDate = DateTime.Now.AddDays(3);

order.OrderStatusID = 1;

Book book = BookStoreEntities.GetContext().Books.FirstOrDefault(p => p.BookID == item.Key.BookID);

if (item.Value.Count >= book.BookCount)

book.BookCount = 0;

else book.BookCount -= item.Value.Count;

BookStoreEntities.GetContext().Orders.Add(order);

\_currentOrder = order;

}

BookStoreEntities.GetContext().SaveChanges(); // сохраняем изменения в БД

// показываем талон заказа в новом окне

OrderTicketWindow orderTicketWindow = new OrderTicketWindow(\_currentOrder);

orderTicketWindow.ShowDialog();

// очищаем корзину

Basket.ClearBasket();

this.Close();

}

}

catch (Exception ex) { MessageBox.Show(ex.Message); }

}

}

}

**CatalogPage.xaml**

<Page x:Class="BookStoreApp.Pages.CatalogPage"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:local="clr-namespace:BookStoreApp.Pages"

mc:Ignorable="d"

d:DesignHeight="450" d:DesignWidth="800"

Title="Каталог книг"

IsVisibleChanged="Page\_IsVisibleChanged"

Style="{StaticResource base\_page}">

<Grid>

<Grid.RowDefinitions>

<RowDefinition Height="auto"/>

<RowDefinition Height="45"/>

<RowDefinition Height="\*"/>

</Grid.RowDefinitions>

<Grid.ColumnDefinitions>

<ColumnDefinition/>

</Grid.ColumnDefinitions>

<StackPanel Grid.Row="0" VerticalAlignment="Stretch" Background="{StaticResource additional}">

<StackPanel Orientation="Horizontal">

<TextBlock Text="Введите наименование книги для поиска"

Style="{StaticResource base\_textblock}"/>

<TextBox Width="275" x:Name="TBoxSearch"

TextChanged="TBoxSearch\_TextChanged"/>

</StackPanel>

<StackPanel Orientation="Horizontal">

<TextBlock Text="Выберите жанр" Style="{StaticResource base\_textblock}"/>

<ComboBox Width="275" x:Name="ComboBoxBookGenre"

SelectionChanged="ComboBoxBookGenre\_SelectionChanged"

DisplayMemberPath="BookGenreName"/>

</StackPanel>

<StackPanel Orientation="Horizontal">

<TextBlock Text="Выберите издательство" Style="{StaticResource base\_textblock}"/>

<ComboBox Width="275" x:Name="ComboBoxPublishingHouse"

SelectionChanged="ComboBoxPublishingHouse\_SelectionChanged"

DisplayMemberPath="PublishingHouseName"/>

</StackPanel>

<StackPanel Orientation="Horizontal">

<TextBlock Text="Выберите автора" Style="{StaticResource base\_textblock}"/>

<ComboBox Width="275" x:Name="ComboBoxAuthor" SelectionChanged="ComboBoxAuthor\_SelectionChanged">

<ComboBox.ItemTemplate>

<DataTemplate>

<TextBlock>

<TextBlock.Text>

<MultiBinding StringFormat="{}{0} {1} {2}">

<Binding Path="AuthorSurname"/>

<Binding Path="AuthorName"/>

<Binding Path="AuthorPatronymic"/>

</MultiBinding>

</TextBlock.Text>

</TextBlock>

</DataTemplate>

</ComboBox.ItemTemplate>

</ComboBox>

</StackPanel>

<StackPanel Orientation="Horizontal">

<TextBlock Text="Сортировать цену"

Style="{StaticResource base\_textblock}" />

<ComboBox Width="275" x:Name="ComboSort" SelectionChanged="ComboSort\_SelectionChanged" >

<ComboBoxItem Content="по возрастанию"/>

<ComboBoxItem Content="по убыванию"/>

</ComboBox>

</StackPanel>

</StackPanel>

<TextBlock Grid.Row="1" x:Name="TextBlockCount" Width="auto" HorizontalAlignment="Left" />

<TextBlock Grid.Row="1" x:Name="TextBlockBasketInfo" Width="auto" Height="auto" HorizontalAlignment="Right" Margin="0 0 140 0" />

<Button Grid.Row="1" Width="120" HorizontalAlignment="Right" Content="Корзина" x:Name="BtnBasket" Click="BtnBasket\_Click" Height="Auto" Margin="0,5,5,5" Style="{StaticResource MaterialDesignRaisedButton}" FontSize="18"/>

<ListBox x:Name="ListBoxBooks"

ScrollViewer.VerticalScrollBarVisibility="Visible"

ScrollViewer.HorizontalScrollBarVisibility="Disabled"

SelectedValuePath="BookID" Grid.Row="2" Grid.Column="0"

ItemContainerStyle="{StaticResource good\_item}"

SelectionChanged="ListBoxProducts\_SelectionChanged">

<ListBox.ItemsPanel>

<ItemsPanelTemplate>

<WrapPanel Orientation="Horizontal" HorizontalAlignment="Center"/>

</ItemsPanelTemplate>

</ListBox.ItemsPanel>

<ListBox.ItemTemplate>

<DataTemplate>

<Grid Margin="20" Width="300">

<Grid.RowDefinitions>

<RowDefinition Height="310"/>

<RowDefinition Height="auto"/>

<RowDefinition Height="20"/>

</Grid.RowDefinitions>

<Image Grid.Row="0" Width="240" Stretch="Uniform"

HorizontalAlignment="Center" Margin="5"

Source="{Binding Path=GetPhoto}"/>

<StackPanel Grid.Row="1" Height="100">

<TextBlock Padding="3" Height="Auto"

Style="{StaticResource item\_textblock}" Text="{Binding BookName}"/>

<TextBlock Text="{Binding BookPrice, StringFormat='c',

ConverterCulture='ru-RU'}" Height="Auto"

Style="{StaticResource item\_textblock}"

VerticalAlignment="Center"

Foreground="Green" FontSize="20"/>

<TextBlock Padding="3" Height="Auto" FontSize="16"

Style="{StaticResource item\_textblock}" Text="{Binding Path=BookCount, StringFormat={}Количество: {0}}"/>

</StackPanel>

</Grid>

</DataTemplate>

</ListBox.ItemTemplate>

<ListBox.ContextMenu>

<ContextMenu>

<MenuItem Header="Добавить в заказ" Click="MenuItem\_Click"></MenuItem>

</ContextMenu>

</ListBox.ContextMenu>

</ListBox>

</Grid>

</Page>

**RegisterWindow.xaml**

<Window x:Class="BookStoreApp.Windows.RegisterWindow"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:local="clr-namespace:BookStoreApp.Windows"

xmlns:materialDesign="http://materialdesigninxaml.net/winfx/xaml/themes"

mc:Ignorable="d"

Title="Регистрация" Style="{StaticResource base\_window}" Height="530"

Width="510" MinHeight="530" MinWidth="510"

WindowStartupLocation="CenterScreen"

Closing="Window\_Closing"

ResizeMode="NoResize">

<Grid>

<Grid.ColumnDefinitions>

<ColumnDefinition />

</Grid.ColumnDefinitions>

<Grid.RowDefinitions>

<RowDefinition Height="60"/>

<RowDefinition Height="50"/>

<RowDefinition Height="50"/>

<RowDefinition Height="50"/>

<RowDefinition Height="50"/>

<RowDefinition Height="50"/>

<RowDefinition Height="50"/>

<RowDefinition Height="50"/>

<RowDefinition Height="60"/>

</Grid.RowDefinitions>

<Grid Background="{StaticResource additional}" />

<TextBlock Grid.Row="0" Text="Мелкооптовый книжный магазин"

HorizontalAlignment="Center"

VerticalAlignment="Center"

Background="{StaticResource additional}"

FontSize="20"

Foreground="Black"/>

<TextBox x:Name="TbLogin"

Grid.Row="1"

Width="450"

VerticalAlignment="Center"

HorizontalAlignment="Center"

HorizontalContentAlignment="Left"

materialDesign:HintAssist.Hint="Логин"

Style="{StaticResource MaterialDesignFloatingHintTextBox}"

materialDesign:TextFieldAssist.HasClearButton="True"

Margin="0,2,0,3"

Padding="0,0,0,5" />

<PasswordBox x:Name="TbPass"

Grid.Row="2"

Width="450"

materialDesign:HintAssist.Hint="Пароль"

Style="{StaticResource MaterialDesignFloatingHintPasswordBox}"

materialDesign:TextFieldAssist.HasClearButton="True"/>

<TextBox x:Name="TextBoxClientSurname"

Grid.Row="3"

Width="450"

materialDesign:HintAssist.Hint="Фамилия"

Style="{StaticResource MaterialDesignFloatingHintTextBox}"

materialDesign:TextFieldAssist.HasClearButton="True"/>

<TextBox x:Name="TextBoxClientName"

Grid.Row="4"

Width="450"

materialDesign:HintAssist.Hint="Имя"

Style="{StaticResource MaterialDesignFloatingHintTextBox}"

materialDesign:TextFieldAssist.HasClearButton="True"/>

<TextBox x:Name="TextBoxClientPatronymic"

Grid.Row="5"

Width="450"

materialDesign:HintAssist.Hint="Отчество"

Style="{StaticResource MaterialDesignFloatingHintTextBox}"

materialDesign:TextFieldAssist.HasClearButton="True"/>

<TextBox x:Name="TextBoxClientAddress"

Grid.Row="6"

Width="450"

materialDesign:HintAssist.Hint="Адрес"

Style="{StaticResource MaterialDesignFloatingHintTextBox}"

materialDesign:TextFieldAssist.HasClearButton="True"/>

<TextBox x:Name="TextBoxClientPhoneNumber"

Grid.Row="7"

Width="450"

materialDesign:HintAssist.Hint="Номер телефона"

Style="{StaticResource MaterialDesignFloatingHintTextBox}"

materialDesign:TextFieldAssist.HasClearButton="True"/>

<StackPanel Orientation="Horizontal" Grid.Row="8" HorizontalAlignment="Center" Margin="10,5,10,-12" Width="474">

<Button Width="150" Height="Auto"

Content="OK"

x:Name="BtnRegister"

FontSize="18"

Click="BtnRegister\_Click" Margin="50,10,20,10"

Style="{StaticResource MaterialDesignRaisedButton}" />

<Button Width="150"

Height="Auto"

Content="Отмена"

x:Name="BtnCancel"

FontSize="18"

Click="BtnCancel\_Click" Margin="50,10,0,10"

Style="{StaticResource MaterialDesignRaisedButton}" />

</StackPanel>

</Grid>

</Window>

**RegisterWindow.xaml.cs**

using BookStoreApp.Models;

using System;

using System.Linq;

using System.Text;

using System.Windows;

namespace BookStoreApp.Windows

{

public partial class RegisterWindow : Window

{

private BookStoreEntities \_context;

public RegisterWindow()

{

InitializeComponent();

\_context = BookStoreEntities.GetContext();

}

private void Window\_Closing(object sender, System.ComponentModel.CancelEventArgs e)

{

// Создание нового экземпляра окна входа

LoginWindow loginWindow = new LoginWindow();

// Открытие окна входа

loginWindow.Show();

}

private void BtnRegister\_Click(object sender, RoutedEventArgs e)

{

try

{

// Проверка полей на содержимое

StringBuilder errors = CheckFields();

if (errors.Length > 0)

{

MessageBox.Show(errors.ToString());

return;

}

// Проверка на уже существующий логин

if (\_context.Clients.Any(c => c.ClientLogin == TbLogin.Text))

{

MessageBox.Show("Пользователь с таким логином уже существует.");

return;

}

// Создание нового клиента

Client newClient = new Client

{

ClientLogin = TbLogin.Text,

ClientPassword = TbPass.Password,

ClientSurname = TextBoxClientSurname.Text,

ClientName = TextBoxClientName.Text,

ClientPatronymic = TextBoxClientPatronymic.Text,

ClientAddress = TextBoxClientAddress.Text,

ClientPhoneNumber = TextBoxClientPhoneNumber.Text,

RoleID = 3

};

// Добавление нового клиента в базу данных

\_context.Clients.Add(newClient);

// Сохранение изменений

\_context.SaveChanges();

MessageBox.Show("Регистрация успешна");

this.Close();

}

catch (Exception ex) { MessageBox.Show(ex.Message.ToString()); }

}

private StringBuilder CheckFields()

{

StringBuilder s = new StringBuilder();

// Проверка полей на содержимое

if (string.IsNullOrWhiteSpace(TbLogin.Text))

s.AppendLine("Заполните «Логин»");

if (string.IsNullOrWhiteSpace(TbPass.Password))

s.AppendLine("Заполните «Пароль»");

if (string.IsNullOrWhiteSpace(TextBoxClientSurname.Text))

s.AppendLine("Заполните «Фамилию»");

if (string.IsNullOrWhiteSpace(TextBoxClientName.Text))

s.AppendLine("Заполните «Имя»");

if (string.IsNullOrWhiteSpace(TextBoxClientPatronymic.Text))

s.AppendLine("Заполните «Отчество»");

if (string.IsNullOrWhiteSpace(TextBoxClientAddress.Text))

s.AppendLine("Заполните «Адрес»");

if (string.IsNullOrWhiteSpace(TextBoxClientPhoneNumber.Text))

s.AppendLine("Заполните «Телефон»");

return s;

}

private void BtnCancel\_Click(object sender, RoutedEventArgs e)

{

// Закрытие окна регистрации

this.Close();

}

}

}

**CatalogPage.xaml.cs**

using BookStoreApp.Models;

using BookStoreApp.Windows;

using System.Linq;

using System.Windows;

using System.Windows.Controls;

namespace BookStoreApp.Pages

{

/// <summary>

/// Логика взаимодействия для CatalogPage.xaml

/// </summary>

public partial class CatalogPage : Page

{

int \_itemcount = 0; //

Book \_selectedBook = null;

public CatalogPage()

{

InitializeComponent();

LoadAndInitData();

}

void LoadAndInitData()

{

// загрузка данных в ComboBoxBookGenre + добавление дополнительной строки

var bookGenres = BookStoreEntities.GetContext().BookGenres.OrderBy(p => p.BookGenreName).ToList();

bookGenres.Insert(0, new BookGenre

{

BookGenreName = "все"

}

);

ComboBoxBookGenre.ItemsSource = bookGenres;

ComboBoxBookGenre.SelectedIndex = 0;

// загрузка данных в ComboBoxPublishingHouse + добавление дополнительной строки

var publishingHouses = BookStoreEntities.GetContext().PublishingHouses.OrderBy(p => p.PublishingHouseName).ToList();

publishingHouses.Insert(0, new PublishingHouse

{

PublishingHouseName = "все"

}

);

ComboBoxPublishingHouse.ItemsSource = publishingHouses;

ComboBoxPublishingHouse.SelectedIndex = 0;

// загрузка данных в ComboBoxAuthor + добавление дополнительной строки

var authors = BookStoreEntities.GetContext().Authors.OrderBy(p => p.AuthorSurname).ToList();

authors.Insert(0, new Author

{

AuthorSurname = "все"

}

);

ComboBoxAuthor.ItemsSource = authors;

ComboBoxAuthor.SelectedIndex = 0;

// загрузка данных в ListBoxBooks сортируем по названию

ListBoxBooks.ItemsSource = BookStoreEntities.GetContext().Books.OrderBy(p => p.BookName).ToList();

\_itemcount = ListBoxBooks.Items.Count;

// скрываем кнопки корзины

BtnBasket.Visibility = Visibility.Collapsed;

TextBlockBasketInfo.Visibility = Visibility.Collapsed;

// отображение количества записей

TextBlockCount.Text = $" Результат запроса: {\_itemcount} записей из {\_itemcount}";

Basket.ClearBasket();

}

/// <summary>

/// Метод для фильтрации и сортировки данных

/// </summary>

private void UpdateData()

{

// получаем текущие данные из бд

var currentGoods = BookStoreEntities.GetContext().Books.OrderBy(p => p.BookName).ToList();

// выбор тех товаров, в названии которых есть поисковая строка

currentGoods = currentGoods.Where(p => p.BookName.ToLower().Contains(TBoxSearch.Text.ToLower())).ToList();

// сортировка

if (ComboSort.SelectedIndex >= 0)

{

// сортировка по возрастанию цены

if (ComboSort.SelectedIndex == 0)

currentGoods = currentGoods.OrderBy(p => p.BookPrice).ToList();

// сортировка по убыванию цены

if (ComboSort.SelectedIndex == 1)

currentGoods = currentGoods.OrderByDescending(p => p.BookPrice).ToList();

}

// выбор только тех книг, которые принадлежат данному разделу книг

if (ComboBoxBookGenre.SelectedIndex > 0)

currentGoods = currentGoods.Where(p => p.BookGenreID == (ComboBoxBookGenre.SelectedItem as BookGenre).BookGenreID).ToList();

// выбор только тех книг, которые принадлежат данному издательству книг

if (ComboBoxPublishingHouse.SelectedIndex > 0)

currentGoods = currentGoods.Where(p => p.PublishingHouseID == (ComboBoxPublishingHouse.SelectedItem as PublishingHouse).PublishingHouseID).ToList();

// выбор только тех книг, которые принадлежат данному автору книг

if (ComboBoxAuthor.SelectedIndex > 0)

currentGoods = currentGoods.Where(p => p.AuthorID == (ComboBoxAuthor.SelectedItem as Author).AuthorID).ToList();

// в качестве источника данных присваиваем список данных

ListBoxBooks.ItemsSource = currentGoods;

// отображение количества записей

TextBlockCount.Text = $" Результат запроса: {currentGoods.Count} записей из {\_itemcount}";

}

private string GetCorrectForm(int count)

{

int remainder100 = count % 100;

int remainder10 = count % 10;

if (remainder100 >= 11 && remainder100 <= 19)

return "товаров";

if (remainder10 == 1)

return "товар";

if (remainder10 >= 2 && remainder10 <= 4)

return "товара";

return "товаров";

}

private void MenuItem\_Click(object sender, RoutedEventArgs e)

{

// контекстное меню по нажатию правой кнопки мыши

// если товар не выбран, завершаем работу

if (\_selectedBook == null)

return;

// добавляем товар в корзину

Basket.AddProductInBasket(\_selectedBook);

// отображаем кнопку и текстовое поле

if (Basket.GetCount > 0)

{

BtnBasket.Visibility = Visibility.Visible;

TextBlockBasketInfo.Visibility = Visibility.Visible;

TextBlockBasketInfo.Text = $"В корзине {Basket.GetCount} {GetCorrectForm(Basket.GetCount)}";

}

}

private void ListBoxProducts\_SelectionChanged(object sender, SelectionChangedEventArgs e)

{

// получаем все выделенные товары

\_selectedBook = null;

var selected = ListBoxBooks.SelectedItems.Cast<Book>().ToList();

if (selected.Count == 0) return;

\_selectedBook = selected[0];

}

private void BtnBasket\_Click(object sender, RoutedEventArgs e)

{

// кнопка Корзина

NewOrderWindow newOrderWindow = new NewOrderWindow();

newOrderWindow.ShowDialog();

if (Basket.GetCount > 0)

{

BtnBasket.Visibility = Visibility.Visible;

TextBlockBasketInfo.Visibility = Visibility.Visible;

}

else

{

BtnBasket.Visibility = Visibility.Collapsed;

TextBlockBasketInfo.Visibility = Visibility.Collapsed;

}

}

private void ComboBoxBookGenre\_SelectionChanged(object sender, SelectionChangedEventArgs e)

{

UpdateData();

}

private void ComboBoxPublishingHouse\_SelectionChanged(object sender, SelectionChangedEventArgs e)

{

UpdateData();

}

private void Page\_IsVisibleChanged(object sender, DependencyPropertyChangedEventArgs e)

{

// обновление данных после каждой активации окна

if (Visibility == Visibility.Visible)

{

BookStoreEntities.GetContext().ChangeTracker.Entries().ToList().ForEach(p => p.Reload());

ListBoxBooks.ItemsSource = BookStoreEntities.GetContext().Books.OrderBy(p => p.BookName).ToList();

}

}

// поиск товаров, которые содержат данную поисковую строку

private void TBoxSearch\_TextChanged(object sender, TextChangedEventArgs e)

{

UpdateData();

}

// сортировка товаров

private void ComboSort\_SelectionChanged(object sender, SelectionChangedEventArgs e)

{

UpdateData();

}

private void ComboBoxAuthor\_SelectionChanged(object sender, SelectionChangedEventArgs e)

{

UpdateData();

}

}

}

**ClientsPage.xaml**

<Page x:Class="BookStoreApp.Pages.ClientsPage"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:local="clr-namespace:BookStoreApp.Pages"

mc:Ignorable="d"

d:DesignHeight="550" d:DesignWidth="740"

Title="Клиенты" IsVisibleChanged="Page\_IsVisibleChanged" Style="{StaticResource base\_page}">

<Grid>

<Grid.RowDefinitions>

<RowDefinition Height="\*"/>

<RowDefinition Height="60"/>

</Grid.RowDefinitions>

<DataGrid x:Name="DataGridClients" Grid.Row="0"

AutoGenerateColumns="False"

IsReadOnly="True"

RowHeight="50"

SelectionMode="Single"

LoadingRow="DataGridClients\_LoadingRow">

<DataGrid.Columns>

<DataGridTextColumn Binding="{Binding ClientLogin}" Header="Логин" Width="auto"/>

<DataGridTextColumn Binding="{Binding ClientPassword}" Header="Пароль" Width="auto"/>

<DataGridTextColumn Binding="{Binding ClientSurname}" Header="Фамилия" Width="auto"/>

<DataGridTextColumn Binding="{Binding ClientName}" Header="Имя" Width="auto"/>

<DataGridTextColumn Binding="{Binding ClientPatronymic}" Header="Отчество" Width="auto"/>

<DataGridTextColumn Binding="{Binding ClientAddress}" Header="Адрес" Width="auto"/>

<DataGridTextColumn Binding="{Binding ClientPhoneNumber}" Header="Номер телефона" Width="auto"/>

<DataGridTextColumn Binding="{Binding Role.RoleName}" Header="Роль" Width="auto"/>

<DataGridTemplateColumn Width="170" Header="Редактировать">

<DataGridTemplateColumn.CellTemplate>

<DataTemplate>

<Button x:Name="BtnEdit" Content="..." Click="BtnEdit\_Click" Margin="0" Style="{StaticResource MaterialDesignRaisedButton}" FontSize="18"/>

</DataTemplate>

</DataGridTemplateColumn.CellTemplate>

</DataGridTemplateColumn>

</DataGrid.Columns>

</DataGrid>

<Button x:Name="BtnAdd" Grid.Row="1" Width="120" HorizontalAlignment="Left" VerticalAlignment="Center"

Content="Добавить" Click="BtnAdd\_Click" Height="34" Margin="30,13,0,13" Style="{StaticResource MaterialDesignRaisedButton}" FontSize="18"/>

<Button x:Name="BtnDelete" Grid.Row="1" Width="110" HorizontalAlignment="Right" VerticalAlignment="Center"

Content="Удалить" Click="BtnDelete\_Click" Height="34" Margin="0,13,30,13" Style="{StaticResource MaterialDesignRaisedButton}" FontSize="18"/>

</Grid>

</Page>

**ClientsPage.xaml.cs**

using BookStoreApp.Models;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Windows;

using System.Windows.Controls;

namespace BookStoreApp.Pages

{

/// <summary>

/// Логика взаимодействия для ClientsPage.xaml

/// </summary>

public partial class ClientsPage : Page

{

public ClientsPage()

{

InitializeComponent();

}

// отображение номеров строк в DataGrid

private void DataGridClients\_LoadingRow(object sender, DataGridRowEventArgs e)

{

e.Row.Header = (e.Row.GetIndex() + 1).ToString();

}

private void BtnAdd\_Click(object sender, RoutedEventArgs e)

{

// открытие AddClientPage для добавления новой записи

Manager.MainFrame.Navigate(new AddClientPage(null));

}

private void BtnDelete\_Click(object sender, RoutedEventArgs e)

{

// удаление выбранного клиента из таблицы

// получаем всех выделенных клиентов

var selectedClients = DataGridClients.SelectedItems.Cast<Client>().ToList();

// вывод сообщения с вопросом Удалить запись?

MessageBoxResult messageBoxResult = MessageBox.Show($"Удалить запись?",

"Удаление", MessageBoxButton.OKCancel, MessageBoxImage.Question);

// если пользователь нажал ОК, пытаемся удалить запись

if (messageBoxResult == MessageBoxResult.OK)

{

try

{

// берем из списка удаляемых клиентов один элемент

Client x = selectedClients[0];

// проверка, есть ли у выбранного клиента в таблице «Заказы» связанные записи

// если да, то выбрасывается исключение и удаление прерывается

if (x.Orders.Count > 0)

throw new Exception("Есть связанная запись в таблице «Заказы», удаление записи невозможно");

BookStoreEntities.GetContext().Clients.Remove(x);

// сохраняем изменения

BookStoreEntities.GetContext().SaveChanges();

MessageBox.Show("Записи удалены");

List<Client> clients = BookStoreEntities.GetContext().Clients.OrderBy(p => p.ClientID).ToList();

DataGridClients.ItemsSource = null;

DataGridClients.ItemsSource = clients;

}

catch (Exception ex)

{

MessageBox.Show(ex.Message.ToString(), "Ошибка удаления", MessageBoxButton.OK, MessageBoxImage.Error);

}

}

}

private void Page\_IsVisibleChanged(object sender, DependencyPropertyChangedEventArgs e)

{

// событие отображения данного Page

// обновляем данные каждый раз, когда активируется этот Page

if (Visibility == Visibility.Visible)

{

DataGridClients.ItemsSource = null;

// загрузка обновленных данных

BookStoreEntities.GetContext().ChangeTracker.Entries().ToList().ForEach(p => p.Reload());

List<Client> clients = BookStoreEntities.GetContext().Clients.OrderBy(p => p.ClientID).ToList();

DataGridClients.ItemsSource = clients;

}

}

private void BtnEdit\_Click(object sender, RoutedEventArgs e)

{

// открытие редактирования клиента

// передача выбранного клиента в AddClientPage

Manager.MainFrame.Navigate(new AddClientPage((sender as Button).DataContext as Client));

}

}

}

**OrdersPage.xaml**

<Page x:Class="BookStoreApp.Pages.OrdersPage"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:local="clr-namespace:BookStoreApp.Pages"

mc:Ignorable="d"

xmlns:materialDesign="http://materialdesigninxaml.net/winfx/xaml/themes"

d:DesignHeight="550" d:DesignWidth="1200"

Title="Заказы" IsVisibleChanged="Page\_IsVisibleChanged" Style="{StaticResource base\_page}">

<Grid>

<Grid.RowDefinitions>

<RowDefinition Height="\*"/>

<RowDefinition Height="60"/>

</Grid.RowDefinitions>

<DataGrid x:Name="DataGridOrders" Grid.Row="0"

AutoGenerateColumns="False"

IsReadOnly="True"

RowHeight="50"

SelectionMode="Single"

LoadingRow="DataGridOrders\_LoadingRow">

<DataGrid.Columns>

<DataGridTextColumn Binding="{Binding OrderID}" Header="ID" Width="Auto"/>

<DataGridTextColumn Binding="{Binding Client.ClientSurname}" Header="Фамилия" Width="Auto"/>

<DataGridTextColumn Binding="{Binding Client.ClientName}" Header="Имя" Width="Auto"/>

<DataGridTextColumn Binding="{Binding Client.ClientPatronymic}" Header="Отчество" Width="Auto"/>

<DataGridTextColumn Binding="{Binding Client.ClientAddress}" Header="Адрес" Width="Auto"/>

<DataGridTextColumn Binding="{Binding Client.ClientPhoneNumber}" Header="Номер телефона" Width="Auto"/>

<DataGridTextColumn Binding="{Binding Book.BookName}" Header="Название книги" Width="Auto"/>

<DataGridTextColumn Binding="{Binding OrderCount}" Header="Количество" Width="Auto"/>

<DataGridTextColumn Binding="{Binding OrderCreateDate, StringFormat=dd.MM.yyyy}" Header="Дата создания" Width="Auto"/>

<DataGridTextColumn Binding="{Binding OrderDeliveryDate, StringFormat=dd.MM.yyyy}" Header="Дата доставки" Width="Auto"/>

<DataGridTextColumn Binding="{Binding OrderStatu.OrderStatusName}" Header="Статус" Width="Auto"/>

<DataGridTemplateColumn Width="170" Header="Редактировать">

<DataGridTemplateColumn.CellTemplate>

<DataTemplate>

<Button x:Name="BtnEdit" Content="..." Click="BtnEdit\_Click" Margin="0" Style="{StaticResource MaterialDesignRaisedButton}"/>

</DataTemplate>

</DataGridTemplateColumn.CellTemplate>

</DataGridTemplateColumn>

</DataGrid.Columns>

</DataGrid>

<StackPanel Grid.Row="1" Orientation="Horizontal" VerticalAlignment="Center" HorizontalAlignment="Center">

<Button x:Name="BtnAdd" Content="Добавить" Click="BtnAdd\_Click" Margin="5" Style="{StaticResource MaterialDesignRaisedButton}" FontSize="18"/>

<Button x:Name="BtnDelete" Content="Удалить" Click="BtnDelete\_Click" Margin="5" Style="{StaticResource MaterialDesignRaisedButton}" FontSize="18"/>

<StackPanel Orientation="Horizontal">

<ComboBox x:Name="CmbSearchType" Height="30" Background="White" VerticalAlignment="Stretch" VerticalContentAlignment="Center" Width="200" SelectedIndex="0" Padding="10,0,0,0">

<ComboBoxItem Content="Поиск по ID"/>

<ComboBoxItem Content="Поиск по фамилии клиента"/>

<ComboBoxItem Content="Поиск по дате подачи заявки"/>

<ComboBoxItem Content="Поиск по номеру телефона"/>

</ComboBox>

<TextBox x:Name="TbSearchID" Width="200" BorderThickness="1" Background="White" Margin="5"/>

<Button x:Name="BtnSearch" Width="60" Click="BtnSearch\_Click" Background="White" BorderBrush="Black" Margin="5">

<materialDesign:PackIcon Kind="Search" Foreground="#FF0000" VerticalAlignment="Center"/>

</Button>

<Button x:Name="BtnClearSearch" Width="60" Click="BtnClearSearch\_Click" Background="White" BorderBrush="Black" Margin="5">

<materialDesign:PackIcon Kind="Cancel" Foreground="#FF0000" VerticalAlignment="Center"/>

</Button>

<Button x:Name="BtnExcel" Width="60" Background="White" BorderBrush="Black" Click="BtnExcel\_Click" Margin="5">

<materialDesign:PackIcon Kind="FileExcel" Foreground="#FF087008" VerticalAlignment="Center"/>

</Button>

</StackPanel>

</StackPanel>

</Grid>

</Page>

**OrdersPage.xaml.cs**

using BookStoreApp.Models;

using System;

using System.Collections.Generic;

using System.Diagnostics;

using System.Linq;

using System.Windows;

using System.Windows.Controls;

using Excel = Microsoft.Office.Interop.Excel;

namespace BookStoreApp.Pages

{

public partial class OrdersPage : Page

{

private List<Order> orders; // Хранит список заказов для использования в сортировке и поиске

public OrdersPage()

{

InitializeComponent();

LoadOrders(); // Загружаем список заказов при инициализации страницы

}

// Метод загрузки заказов и привязки к DataGrid

private void LoadOrders()

{

orders = BookStoreEntities.GetContext().Orders.OrderBy(p => p.OrderID).ToList();

DataGridOrders.ItemsSource = orders;

}

// Отображение номеров строк в DataGrid

private void DataGridOrders\_LoadingRow(object sender, DataGridRowEventArgs e)

{

e.Row.Header = (e.Row.GetIndex() + 1).ToString();

}

private void BtnAdd\_Click(object sender, RoutedEventArgs e)

{

// Открытие AddOrderPage для добавления новой записи

Manager.MainFrame.Navigate(new AddOrderPage(null));

}

private void BtnDelete\_Click(object sender, RoutedEventArgs e)

{

// Удаление выбранного заказа из таблицы

// Получаем все выделенные заказы

var selectedOrders = DataGridOrders.SelectedItems.Cast<Order>().ToList();

// Вывод сообщения с вопросом "Удалить запись?"

MessageBoxResult messageBoxResult = MessageBox.Show("Удалить запись?",

"Удаление", MessageBoxButton.OKCancel, MessageBoxImage.Question);

// Если пользователь нажал "ОК", пытаемся удалить запись

if (messageBoxResult == MessageBoxResult.OK)

{

try

{

// Берем из списка удаляемых заказов один элемент

Order x = selectedOrders[0];

BookStoreEntities.GetContext().Orders.Remove(x);

// Сохраняем изменения

BookStoreEntities.GetContext().SaveChanges();

MessageBox.Show("Запись удалена");

LoadOrders(); // Перезагружаем список заказов после удаления

}

catch (Exception ex)

{

MessageBox.Show(ex.Message.ToString(), "Ошибка удаления", MessageBoxButton.OK, MessageBoxImage.Error);

}

}

}

private void Page\_IsVisibleChanged(object sender, DependencyPropertyChangedEventArgs e)

{

// Событие отображения данной страницы

// Обновляем данные каждый раз, когда активируется эта страница

if (Visibility == Visibility.Visible)

{

LoadOrders(); // Перезагружаем список заказов

}

}

private void BtnEdit\_Click(object sender, RoutedEventArgs e)

{

// Открытие редактирования заказа

// Передача выбранного заказа в AddOrderPage

Manager.MainFrame.Navigate(new AddOrderPage((sender as Button).DataContext as Order));

}

private void BtnSearch\_Click(object sender, RoutedEventArgs e)

{

string searchTerm = TbSearchID.Text.Trim();

List<Order> searchResults = new List<Order>();

if (string.IsNullOrEmpty(searchTerm))

{

MessageBox.Show("Введите значение для поиска.");

return;

}

string searchType = (CmbSearchType.SelectedItem as ComboBoxItem)?.Content.ToString();

switch (searchType)

{

case "Поиск по ID":

if (int.TryParse(searchTerm, out int searchID))

{

searchResults = orders.Where(o => o.OrderID == searchID).ToList();

}

break;

case "Поиск по фамилии клиента":

searchResults = orders.Where(o => o.Client.ClientSurname.ToLower().Contains(searchTerm.ToLower())).ToList();

break;

case "Поиск по дате подачи заявки":

if (DateTime.TryParse(searchTerm, out DateTime searchDate))

{

searchResults = orders.Where(o => o.OrderCreateDate.Date == searchDate.Date).ToList();

}

break;

case "Поиск по номеру телефона":

searchResults = orders.Where(o => o.Client.ClientPhoneNumber.ToLower().Contains(searchTerm.ToLower())).ToList();

break;

default:

MessageBox.Show("Выберите тип поиска.");

return;

}

DataGridOrders.ItemsSource = searchResults;

}

private void BtnClearSearch\_Click(object sender, RoutedEventArgs e)

{

// Обработчик события нажатия кнопки "Очистить поиск"

TbSearchID.Clear();

CmbSearchType.SelectedIndex = 0;

LoadOrders(); // Восстанавливаем исходный список заказов

}

private void BtnExcel\_Click(object sender, RoutedEventArgs e)

{

Excel.Application xlApp = new Excel.Application();

Excel.Workbook xlWorkbook = null;

Excel.Worksheet xlSheet = null;

try

{

string fileName = AppDomain.CurrentDomain.BaseDirectory + "Orders.xlsx";

xlWorkbook = xlApp.Workbooks.Add(Type.Missing);

xlSheet = (Excel.Worksheet)xlWorkbook.Sheets[1];

xlSheet.Name = "Список заявок";

int row = 1;

int i = 0;

// Заголовки столбцов

xlSheet.Cells[row, 1] = "ID";

xlSheet.Cells[row, 2] = "Фамилия";

xlSheet.Cells[row, 3] = "Имя";

xlSheet.Cells[row, 4] = "Отчество";

xlSheet.Cells[row, 5] = "Адрес";

xlSheet.Cells[row, 6] = "Номер телефона";

xlSheet.Cells[row, 7] = "Название книги";

xlSheet.Cells[row, 8] = "Количество";

xlSheet.Cells[row, 9] = "Дата создания";

xlSheet.Cells[row, 10] = "Дата доставки";

xlSheet.Cells[row, 11] = "Статус";

row++;

if (DataGridOrders.Items.Count > 0)

{

for (i = 0; i < DataGridOrders.Items.Count; i++)

{

Order order = DataGridOrders.Items[i] as Order;

xlSheet.Cells[row, 1] = order.OrderID.ToString();

xlSheet.Cells[row, 2] = order.Client.ClientSurname;

xlSheet.Cells[row, 3] = order.Client.ClientName;

xlSheet.Cells[row, 4] = order.Client.ClientPatronymic;

xlSheet.Cells[row, 5] = order.Client.ClientAddress;

xlSheet.Cells[row, 6] = order.Client.ClientPhoneNumber;

xlSheet.Cells[row, 7] = order.Book.BookName;

xlSheet.Cells[row, 8] = order.OrderCount.ToString();

xlSheet.Cells[row, 9] = order.OrderCreateDate.ToString("dd.MM.yyyy");

xlSheet.Cells[row, 10] = order.OrderDeliveryDate.ToString("dd.MM.yyyy");

xlSheet.Cells[row, 11] = order.OrderStatu.OrderStatusName;

row++;

}

}

xlSheet.Columns.AutoFit();

xlSheet.Rows.AutoFit();

xlWorkbook.SaveAs(fileName);

xlWorkbook.Close();

MessageBox.Show("Данные успешно экспортированы в Excel.");

// Открытие файла после успешного экспорта

Process.Start(fileName);

}

catch (Exception ex)

{

MessageBox.Show("Ошибка экспорта данных в Excel: " + ex.Message);

}

finally

{

// Блок finally, который гарантирует выполнение кода при выходе из блока try-catch-finally.

if (xlSheet != null)

System.Runtime.InteropServices.Marshal.ReleaseComObject(xlSheet);

// Проверка, что переменная xlSheet не является пустой, а затем освобождение ресурсов, связанных с xlSheet.

if (xlWorkbook != null)

System.Runtime.InteropServices.Marshal.ReleaseComObject(xlWorkbook);

// Проверка, что переменная xlWorkbook не является пустой, а затем освобождение ресурсов, связанных с xlWorkbook.

if (xlApp != null)

{

xlApp.Quit();

System.Runtime.InteropServices.Marshal.ReleaseComObject(xlApp);

}

// Проверка, что переменная xlApp не является пустой, а затем вызов метода Quit() для закрытия приложения, связанного с xlApp.

// Затем освобождение ресурсов, связанных с xlApp.

}

}

}

}

**Приложение Б**

Автоматизированная информационная система «Мелкооптовый книжный магазин»

Спецификация

460.ЗМК.00004-01 01

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| |  | | --- | | **Обозначение** | | |  | | --- | | **Наименование** | | |  | | --- | | **Примечание** | |
|  | |  | | --- | | Документация | |  |
| 460.ЗМК.00004-01 12 01 | Автоматизированная информационная система «Мелкооптовый книжный магазин» |  |
|  | Текст программы |  |
| 460.ЗМК.00004-01 13 01 | Автоматизированная информационная система «Мелкооптовый книжный магазин» |  |
|  | Описание программы |  |
| 460.ЗМК.00004-01 34 01 | Автоматизированная информационная система «Мелкооптовый книжный магазин» |  |
|  | Руководство пользователя |  |
| 460.ЗМК.00004-01 81 01 | Автоматизированная информационная система «Мелкооптовый книжный магазин» |  |
|  | Пояснительная записка |  |

**Приложение B**

**Диск с программным средством**