

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ**

**ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**

**«ДОНСКОЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

**(ДГТУ)**

Факультет \_\_\_\_\_«Информатика и вычислительная техника» \_

(наименование факультета)

Кафедра «Программное обеспечение вычислительной техники и автоматизированных систем»

(наименование кафедры)

Зав. кафедрой «ПОВТиАС»

\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_В. В. Долгов

(подпись) (И.О.Ф)

«\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2022г.

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

к курсовой работе по дисциплине (модулю)\_\_\_Объектно-ориентированное\_\_\_\_\_\_\_\_ программирование­\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(наименование учебной дисциплины (модуля))

на тему: Система управления библиотекой

Автор проекта (работы) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ М.Л. Лапардин

(подпись) (И.О.Ф.)

Направление/специальность, профиль/специализация:

09.03.04 «Программная инженерия»

код направления наименование направления (специальности)

\_ ПОВТиАС

наименование профиля (специализации)

Обозначение курсовой работы КР.540000.000 Группа ВПР34

Руководитель проекта \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Б. В. Габрельян

подпись (И. О. Ф.)

Проект (работа) защищен(а) \_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_

дата оценка подпись

Ростов-на-Дону

2022



МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ**

**ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**

**«ДОНСКОЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

**(ДГТУ)**

Факультет «Информатика и вычислительная техника»

(наименование факультета)

Кафедра «Программное обеспечение вычислительной техники и автоматизированных систем»

(наименование кафедры)

Зав. кафедрой «ПОВТиАС»

\_\_\_\_\_\_\_\_\_\_\_\_\_\_ В. В. Долгов

(подпись) (И.О.Ф)

«\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2022г.

**ЗАДАНИЕ**

к курсовой работе по дисциплине (модулю) \_\_\_Объектно-ориентированное\_\_\_\_\_\_ программирование\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(наименование дисциплины (модуля))

Обучающийся Лапардин Максим Леонидович Группа  ВПР34

Тема: Система управления библиотекой

Срок представления проекта (работы) к защите «\_\_» \_\_\_\_\_\_\_\_\_ 2022 г.   
Исходные данные для курсовой работы

1. Задание на выполнение курсовой работы

2. Рихтер, Джеффри. CLR via C#. Программирование на платформе Microsoft .NET Framework на языке С# , 4-е изд. – «Питер», 2012. – 896с.

Содержание пояснительной записки

ВВЕДЕНИЕ:

В данном разделе рассказывается про важность библиотек в современном мире и использование информационных систем в работе библиотек.

Наименование и содержание разделов:

1. Теоретический раздел. В этом разделе даётся определение информационной системы библиотеки, и рассматриваются причины внедрения данной системы. Осуществляется анализ существующих систем управления библиотекой. Происходит постановка задачи.
2. Алгоритмическое конструирование. В данном разделе описаны алгоритмы добавления, изменения и удаление объектов в библиотеке. Подробно рассмотрены шаги алгоритмов, обеспечивающих функциональность системы.
3. Программное конструирование. В данном разделе обосновывается выбор языка программирования, среды разработки и дополнительных инструментов. Описываются классы программного средства, необходимые для регистрации, авторизации пользователей и управления библиотекой.
4. Тестирование программного средства. В данном разделе проводится тестирование программного средства на основе различных входных данных. Проводится проверка на корректное функционирование регистрации, авторизации, окон пользователя, администратора и кадровика для управления библиотекой. Также приводятся рисунки, которые подтверждают работу программного средства.

ЗАКЛЮЧЕНИЕ:

В курсовой работе, в рамках объектно-ориентированной парадигмы программирования были реализованы все поставленные задачи, разработана рабочая система управления библиотекой.

Руководитель проекта (работы) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Б. В. Габрельян

подпись, дата И.О.Ф.

Задание принял к исполнению \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ М.Л. Лапардин

подпись, дата И.О.Ф.

Содержание

Изм.

Лист

№ докум.

Подпись

Дата

Лист

4

КР.540000.000

Разраб.

Лапардин М.Л.

Провер.

Габрельян Б. В.

Система управления библиотекой

Лист

Листов

56

[Введение 6](#_Toc121782702)

[1 Теоретический раздел 8](#_Toc121782703)

[1.1 Обзор предметной области 8](#_Toc121782704)

[1.2 Анализ существующих систем управления библиотекой](#_Toc121782705) 11

[1.3 Интерфейсы существующих систем управления библиотекой 14](#_Toc121782706)

[1.4 Постановка задачи 16](#_Toc121782707)

[1.5 Вывод по главе 17](#_Toc121782708)

[2 Алгоритмическое конструирование 18](#_Toc121782709)

[2.1 Алгоритм добавления объекта 18](#_Toc121782710)

[2.2 Алгоритм изменения объекта 18](#_Toc121782711)

[2.3 Алгоритм удаления объекта 19](#_Toc121782712)

[2.4 Схемы алгоритмов 20](#_Toc121782716)

[2.5 Вывод по главе 23](#_Toc121782717)

[3 Программное конструирование 24](#_Toc121782718)

[3.1 Выбор языка программирования 24](#_Toc121782719)

[3.2 Выбор среды разработки 25](#_Toc121782720)

[3.3 Выбор дополнительных инструментов 25](#_Toc121782721)

[3.4 Описание классов программного средства 26](#_Toc121782722)

[3.5 Даталогическое проектирование базы данных 29](#_Toc121782723)

[3.6 Вывод по главе 38](#_Toc121782724)

[4 Тестирование программного средства 39](#_Toc121782725)

[4.1 Описание процесса тестирования 39](#_Toc121782726)

[4.2 Вывод по главе 45](#_Toc121782727)

[Заключение 45](#_Toc121782728)

[Перечень использованных информационных ресурсов 47](#_Toc121782729)

[Приложение А 48](#_Toc121782730)

[UML-диаграмма классов из пакета «Models» 48](#_Toc121782731)

[Приложение Б 49](#_Toc121782736)

[Листинг основных классов программного средства 49](#_Toc121782737)

ДГТУ

Кафедра «ПОВТиАС»

Введение

Развитие библиотечной деятельности неразделимо с внедрением  
передовых технологий и процессов с целью обеспечения наиболее  
качественного обслуживания читателей. Внедрение автоматизированных  
информационных систем – это не только тенденция времени, но и хорошая  
база для увеличения производительности и качества труда сотрудников  
библиотеки, эффективный способ своевременного обеспечения читателей  
необходимыми им данными и информацией [1].

Библиотека (греч. bibliothēkē, от biblíon – книга и thēkē – хранилище), культурно-просветительное и научно-вспомогательное учреждение, организующее общественное пользование произведениями печати. Библиотеки систематически занимаются сбором, хранением, пропагандой и выдачей читателям произведений печати, а также информационно-библиографической работой. Это информационное, культурное, образовательное учреждение, располагающее организованным фондом документов и представляющее их во временное пользование абонентам, а также осуществляющее другие библиотечные услуги [2].

Деятельность библиотеки складывается из трех основных составляющих: хранение, поиск, и выдача на определенное время, имеющихся в наличие книг.

Актуальность данной темы заключается в том, что многие библиотеки до сих пор придерживаются, традиционных библиотечных систем и процессов. Необходим, переход от бумажной коммуникации к коммуникации безбумажной, чтоб сократить время поиска необходимой литературы, информации о читателе и т.д.

В последнее время информационные технологии стали неотъемлемой частью нашей жизни. Экономические информационные системы, связанные с предоставлением и обработкой информации для всех уровней управления экономическими объектами, приобретают особую важность в общественной жизни. На данный момент невозможно представить какую-либо организацию, не применяющую компьютерных технологий. Это обусловлено и тем, что государственные структуры требуют обязательных отчетов в электронном виде, следовательно, необходима систематизированная информация.

Следовательно, можно сделать вывод, что использование данных информационных систем может существенным образом увеличить эффективность работы библиотек и автоматизировать выполнение однообразных задач.

#### Теоретический раздел

В данном разделе проводится обзор предметной области. Проводится анализ существующих систем управления библиотекой. Делаются выводы по главе и происходит постановка задачи.

* 1. Обзор предметной области

Предметной областью данной системы являются библиотеки.

Для более эффективного управления учетом книг и библиотек была разработана реляционная система учета мониторинга контроля и счета книг, которая облегчает работу персонала в библиотеках и исключает человеческий фактор, весь учет берет на себя машина и не допускает погрешностей [3].

Библиотека – это учреждение культуры, организующее сбор, хранение и общественное пользование произведениями печати и другими документами. Библиотеки систематически занимаются сбором, хранением, пропагандой и выдачей читателям произведений печати, а также информационно-библиографической работой, являются общедоступным источником знаний и основной базой для самообразования.

Основными направлениями работы любой библиотеки являются: комплектование и организация книжного фонда, обслуживание читателей.

Комплектование фондов библиотеки состоит из систематического выявления (путем просмотра библиографических источников и литературы) нужных для данной библиотеки изданий и приобретения их. От своевременности и полноты комплектования библиотеки в значительной мере зависит уровень обслуживания читателей.

Организация книжного фонда включает вопросы учета, расстановки, хранения литературы и доставки ее читателю.

Учет библиотечного фонда – это прием поступивших в библиотеку изданий, запись прибывших и исключение по тем или иным причинам непригодных для читателей изданий в документах библиотеки, которые дают точные сведения о наличном составе фонда и помогают обеспечить его сохранность.

Правильная организация фонда облегчает читателю пользование литературой, библиотекарю - быстрое выполнение читательских требований, а также обеспечивает сохранность фондов как общественной собственности.

Обслуживание читателей библиотеки осуществляется различными путём выдачи литературы и помощи отдельным читателям в подборе необходимой им литературы.

Читатели, приходящие в библиотеку, обязаны иметь при себе читательский билет. При выбытии из вуза (отчисление, окончание обучения, увольнение) читатели обязаны вернуть числящиеся за ними издания и сдать читательские билеты.

Библиотечная деятельность связана с учетом большого количества операций, множество книг и читателей серьезно замедляют работу библиотекарей. Сложность поиска нужной книги в каталоге занимает длительное время, и целиком опирается на компетентность работников библиотеки.

Для ведения библиотечных каталогов, организации поиска требуемых изданий и библиотечной статистики в базе должны храниться сведения, большая часть которых размещаются в аннотированных каталожных карточках

В библиотеку приходит много книг из различных издательств. Каждая книга, хранящаяся в библиотеке, имеет следующие параметры:

* Название книги;
* Количество страниц;
* Год издания;
* Библиотека(название), в которой находится книга;
* Категория;
* Автор;
* Издательство;
* Комментарий;
* Количество экземпляров.

Каждая книга может присутствовать в нескольких экземплярах.

В библиотеке ведется картотека студентов. О каждом студенте заносятся следующие сведения:

* Имя;
* Фамилия;
* Группа;
* Номер семестра.

В библиотеке также ведется картотека учителей. О каждом учителе заносятся следующие сведения:

* Имя;
* Фамилия;
* Кафедра.

Каждому читателю присваивается номер читательского билета. В случае выдачи экземпляра книги в карточке указывается дата выдачи, дата предполагаемого возврата и номер читательского билета.

Данный программный продукт может быть использован в любой библиотеке. Система реализована в виде клиентского приложения, поэтому она может быть размещена на любом компьютере и открыта для свободного доступа для одновременного использования многими библиотеками

* Поиск информации должен быть эффективным, т.е. по запросу пользователя должно выдаваться как можно больше релевантных результатов;
* Система должна иметь высокую скорость загрузки и обработки информации для оперативной работы;
* Система должна иметь средства для поддержания обратной связи с пользователем: информирования пользователя о его заказах, задолженностях и т.п.
  1. Анализ существующих систем управления библиотекой

АБИС «ОРАС-Global» – это система управления библиотекой, которая предназначена для автоматизации и упрощения управления библиотекой. Интерфейс системы «ОРАС-Global» представлен на рисунке 1.1.

Главной отличительной особенностью АБИС «ОРАС-Global» является то, что все функциональные задачи выполняются на центральном сервере, а пользователи работают исключительно в стандартных веб-браузерах на ПК или мобильных устройствах. Эта особенность позволяет легко администрировать и непрерывно развивать систему для всей библиотечной сети области или региона через модернизацию центрального сервера. Это также облегчает техническую поддержку системы через Интернет. Ориентированная на веб-среду, система непрерывно развивается вместе с мировой паутиной.

«OPAC-Global» имеет следующие функциональные возможности:

* комплектование;
* каталогизация;
* ведение авторитетных файлов;
* поиск;
* циркуляция;
* запись пользователей/читателей;
* регистрация посещений;
* отчеты;
* администрирование и настройка системы.

АИС «Фундаментальная библиотека» – это автоматизированная библиотечно-информационная система, которая сокращает временные затраты на административную работу и упрощает поиск книг. Интерфейс системы «Фундаментальная библиотека» показан на рисунке 1.2 [1].

«Фундаментальная библиотека» имеет следующие функциональные возможности:

* единая строка поиска по каталогам с расширенными возможностями;
* поддержка полного цикла библиотечных процессов;
* реализованы основные виды печатных документов для обеспечения бумажного учета;
* веб-интерфейс для поиска и заказа литературы, адаптированный под все виды устройств;
* простота выдачи и приема литературы, полная информация о читателях;
* помимо штрих-кодирования поддерживает бесконтактную технологию учета и инвентаризации книг.

АИБС «Фолиант» – это система для комплексной автоматизации библиотечных процессов от создания библиографии до учета читателей и книговыдачи. На рисунке 1.3 представлен интерфейс системы «Фолиант».

«Фолиант» имеет следующие функциональные возможности:

* Система автоматизации всех процессов работы библиотеки на основе взаимосвязанного функционирования различных модулей (АРМ) системы;
* возможность использования в библиотеках любого типа (от национальных до школьных) и в корпоративных проектах;
* широкие возможности для адаптации к условиям работы конкретной библиотеки;
* настраиваемый, комфортный, дружественный пользовательский интерфейс;
* модульная организация системы - возможность как одновременного, так и постепенного внедрения системы и отдельных АРМ;
* безопасность - контроль доступа к данным по категориям пользователей;
* управляемость - настройка, в том числе удаленная, основных параметров системы;
* поддержка технологии корпоративной каталогизации.
* создание и ведение электронных библиотек полнотекстовых документов.

1С: Библиотека. Редакция 2.0 – это автоматизированная система библиотечных процессов. На рисунке 1.4 представлен интерфейс системы 1С: Библиотека.

«1С: Библиотека» имеет следующие функциональные возможности:

* комплектование;
* каталогизация;
* учет, актуализация и хранение фонда;
* обслуживание читателей;
* виртуальный кабинет читателя;
* формирование отчетных и учетных документов;
* поддержка печатных форм;
* интеграция с бухгалтерией.
  1. Интерфейсы существующих систем управления библиотекой

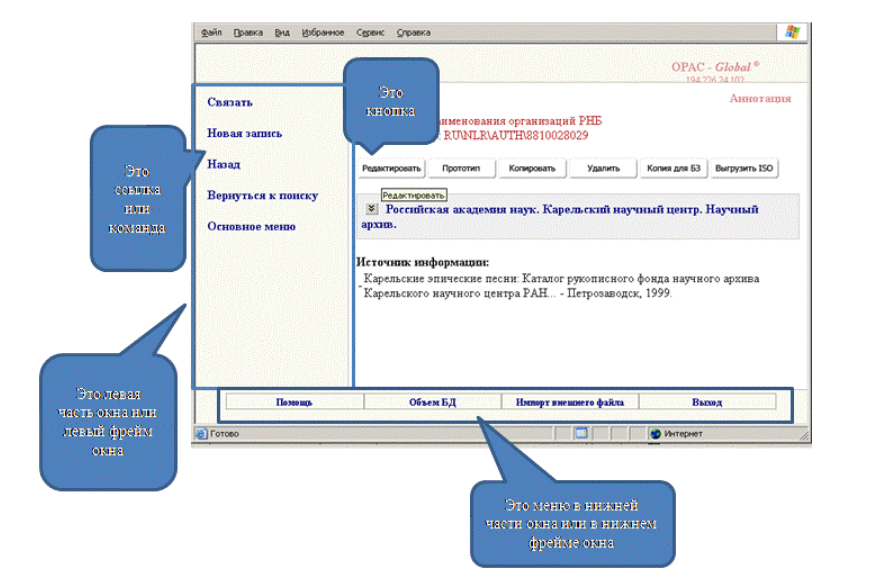


Рисунок .1 – Интерфейс системы управления библиотекой «ОРАС-Global»

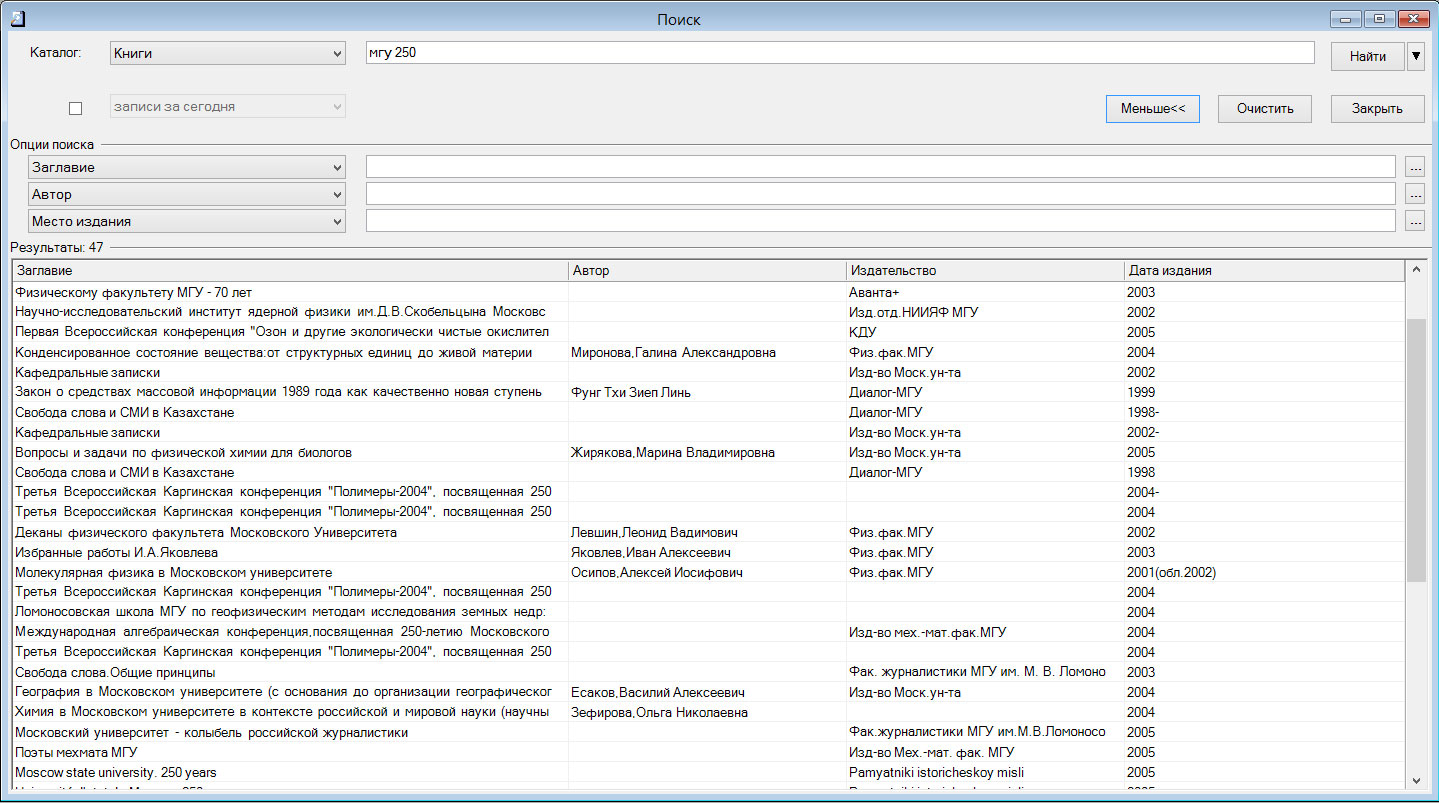


Рисунок 1.2 - Интерфейс автоматизированной библиотечно-информационной системы «Фундаментальная библиотека»

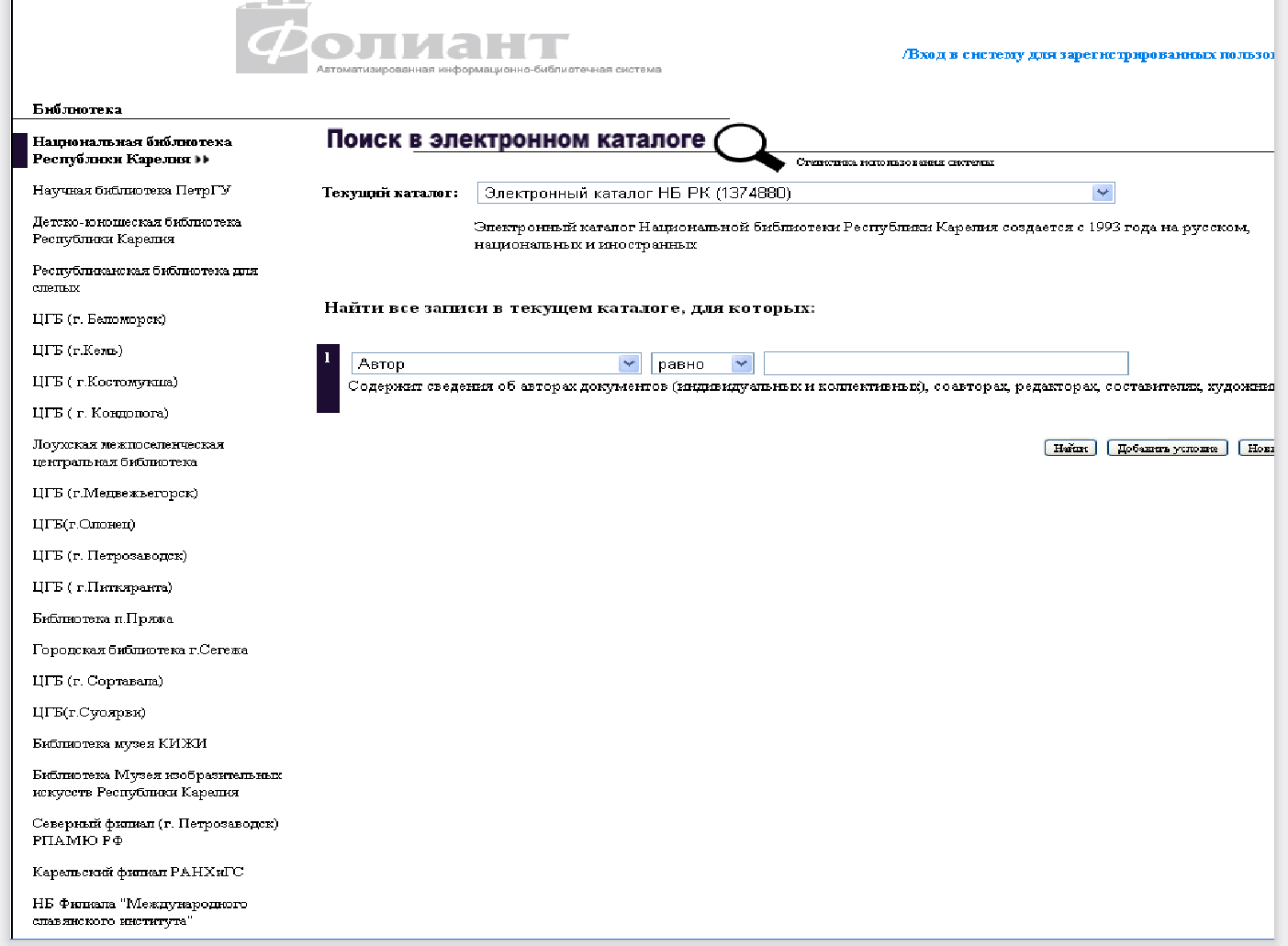


Рисунок 1.3 - Интерфейс системы комплексной автоматизации библиотечных процессов «Фолиант»

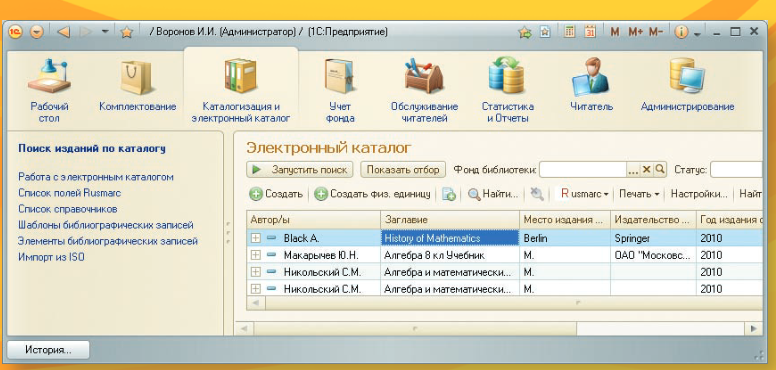


Рисунок 1.4 - Интерфейс автоматизированной системы библиотечных процессов «1С: Библиотека»

* 1. Постановка задачи

Целью данной работы является создание автоматизированной ИС библиотеки. Система предназначена для повышения эффективности работы обслуживающего персонала библиотеки и быстрого обслуживания читателей за счет услуги автоматизированного поиска книги, сокращения времени поиска и оформления выдачи необходимого издания читателю [1].

ИС библиотеки работает в трех режимах в зависимости от типов пользователей: в режиме библиотекаря, администратора или кадровика.

Автоматизация технологических процессов в библиотеке, предоставляемая разрабатываемой системой призвана обеспечить следующие возможности:

1. В режиме работы библиотекаря:

* + чтение, добавление, редактирование и сохранение объектов таблицы книг, карточек студентов и карточек учителей;
  + чтение таблиц учителей, студентов, библиотекарей, библиотек, групп, факультетов, кафедр

1. Возможности в режиме работы администратора:
   * просмотр информации о библиотекарях, зарегистрированных в системе;
   * возможность редактирования данных библиотекарей, а также добавление и удаление библиотекарей.
2. Возможности в режиме работы кадровика:
   * чтение, добавление, редактирование и сохранение объектов таблицы студентов, учителей, библиотек, групп, факультетов и кафедр;
   1. Вывод по главе

В данной главе было определено понятие системы управления библиотекой, и были рассмотрены причины внедрения данных систем. Также был проведён анализ существующих систем управления библиотекой. На основе проведенного обзора была поставлена задача и описаны функции, которые будет выполнять разрабатываемое программное средство.

#### Алгоритмическое конструирование

В данном разделе выполняется алгоритмическое конструирование процессов добавления объекта, изменения и удаления объекта.

1. 1. Алгоритм добавления объекта

На рисунке 2.1 представлена схема алгоритма добавления объекта. Его можно представить в виде следующей последовательности шагов:

1. выбор таблицы, в которую нужно добавить объект
2. ввод данных для добавляемого объекта;
3. нажатие кнопки “Добавить”;
4. создание объекта из выбранной таблицы;
5. добавление объекта в таблицу.
   1. Алгоритм изменения объекта

На рисунке 2.2 представлена схема алгоритма изменения объекта. Его можно представить в виде следующей последовательности шагов:

1. выбор таблицы, в которую нужно добавить объект;
2. ввод данных искомого объекта и нажатие кнопки “Найти”;
3. объект сохраняется в программе.
4. ввод данных для изменения объекта;
5. нажатие кнопки “Изменить”;
6. поиск сохраненного объекта в таблице;
7. изменение данных.
   1. Алгоритм удаления объекта

На рисунке 2.3 изображена схема алгоритма удаления объекта. Его можно представить в виде следующей последовательности шагов:

1. выбор таблицы, в которую нужно добавить объект;
2. ввод данных объекта;
3. поиск объекта в таблице;
4. удаление объекта.
   1. Схемы алгоритмов



Рисунок 2.1 – Схема алгоритма добавления объекта



Рисунок 2.2 – Схема алгоритма изменения объекта



Рисунок 2.3 – Схема алгоритма удаления объекта

* 1. Вывод по главе

В данной главе были рассмотрены следующие алгоритмы:

* алгоритм добавления объекта;
* алгоритм изменения объекта;
* алгоритм удаления объекта;

Были приведены схемы работы данных алгоритмов. Данные алгоритмы используются для дальнейшего проектирования программного средства.

1. Программное конструирование

В данном разделе проводится обоснование выбора средств разработки программного средства и описание структуры программного средства. Также проводится описание подпрограмм, которые реализуют необходимый функционал разрабатываемой информационной системы.

* 1. Выбор языка программирования

Требования по выбору языка программирования для решения поставленной задачи:

* высокая производительность;
* поддержка большого количества вычислений;
* поддержка объектно-ориентированного программирования;
* наличие стандартных и сторонних подключаемых библиотек;
* контроль типов и обработка исключений;
* распространенность;
* доступность справочной информации и информационных ресурсов.

На основе этих требований был выбран язык программирования С#. Это компилируемый язык общего назначения, поддерживающий объектно-ориентированную парадигму программирования [7].

Преимущества языка С#:

* модульность и раздельная компиляция;
* наличие стандартной библиотеки с широкими возможностями;
* возможность подключения сторонних библиотек;
* возможность выбора средств разработки;
* абстракция данных;
* объявление типов (классов) объектов;
* сочетание свойств высокоуровневых и низкоуровневых языков;
* компиляция программного кода непосредственно в машинный код.
  1. Выбор среды разработки

Для написания кода программного средства был выбран компилятор Visual Studio (.NET Framework)[4].

Преимущества Visual Studio:

* поддержка множества языков программирования;
* кроссплатформенность разработки, встроенный контроль за выполнением многопоточного кода;
* наличие бесплатной редакции Community;
* запись происходящего во время отладки (функция IntelliTrace);
* возможность хранения данных в облаке;
* наличие русского языка, вывод списка ошибок и предупреждений;

Недостатки Visual Studio:

* высокое потребление памяти;
* лог-файлы занимают много места на диске.
  1. Выбор дополнительных инструментов

Для того чтобы пользователь мог взаимодействовать с программой в браузере необходимо создание пользовательского интерфейса. Для этих целей был выбран фреймворк Windows Presentation Foundation (WPF)[6].

Windows Presentation Foundation включает широкий набор элементов управления, которые можно добавлять на формы: текстовые поля, кнопки, раскрывающиеся списки, переключатели и даже веб-страницы.

Для управления базой данных используется система управления базами данных Microsoft SQL Server. Данная система поддерживает многопоточность, достаточно проста в использовании и обладает обширным функционалом [5].

Для взаимодействия с базой данных используется библиотека Entity Framework. Данная библиотека предназначена для решения задач объектно-реляционного отображения (ORM).

* 1. Описание классов программного средства

Для реализации сформулированных выше алгоритмов необходимо спроектировать и реализовать структуру программного средства. Структура программы представляет из себя набор классов, которые содержатся в определенных пакетах.

Пакет Models содержит следующие классы:

* Book – представляет книгу;
* Department – представляет кафедру;
* Faculty – представляет факультет;
* Librarian – представляет библиотекаря;
* Lib – представляет библиотеку;
* SCard– представляет карточку студента;
* Student – представляет студента;
* TCard – представляет карточку учителя;
* Teacher – представляет учителя;
* User – представляет пользователя;
* Admin – представляет администратора.
* Officer – представляет кадровика.

Пакет Library состоит из классов:

* Admin\_panel – представляет окно администратора;
* MainWindow – представляет окно пользователя;
* Officer\_panel – представляет окно пользователя;
* Register – представляет окно регистрации;
* Authorize – представляет окно авторизации;

Рассмотрим наиболее важные методы данных классов. В классах, представляющих объекты библиотеки не описана логика работы программы. Она описана в таких классах, как MainWindow, Admin\_panel, Authorize, Register.

В таблице 3.1 представлены методы класса Register.

Таблица 3.1 – Класс Register

|  |  |
| --- | --- |
| Класс | Register |
| Описание класса | Выполняет действия, которые связаны с регистрацией |
| Методы класса | Описание метода |
| Register | Конструктор класса |
| ShowPassword\_Checked | Показывает пароль |
| ShowPassword\_Unchecked | Скрывает пароль |
| Login\_Click | Закрывает окно регистрации и открывает окно авторизации |
| Register\_Click | Регистрирует пользователя |

В таблице 3.2 представлены методы класса Authorize.

Таблица 3.2 – Класс Authorize

|  |  |
| --- | --- |
| Класс | Authorize |
| Описание класса | Выполняет действия, которые связаны с авторизацией |
| Методы класса | Описание метода |
| Authorize | Конструктор класса |
| ShowPassword\_Checked | Показывает пароль |
| ShowPassword\_Unchecked | Скрывает пароль |
| Login\_Click | Авторизует пользователя или администратора |

В таблице 3.3 представлены методы класса Admin\_panel.

Таблица 3.3 – Класс Admin\_panel

|  |  |
| --- | --- |
| Класс | Admin\_panel |
| Описание класса | Выполняет действия, которые связаны с работой администратора |
| Методы класса | Описание метода |
| Admin\_panel | Конструктор класса |
| Add\_Object | Событие добавления объекта в таблицу БД. Добавляет объект в выбранную таблицу из списка |
| UpdateObject | Событие изменения объекта в таблице БД |
| Методы класса | Описание метода |
| DeleteObject | Событие удаление объекта из таблицы БД. Удаляет объект в выбранной таблице. |
| RefreshData | Обновляет данные в таблице |
| LibraryView | Выводит выбранную таблицу на экран |
| ExportObject | Событие экспорта БД, вызывает проводник и функцию ExportDatabase |
| ExportDatabase | Создает резервную копию БД |
| ImportObject | Событие импорта БД, вызывает проводник и функцию ImportDatabase |
| ImportDatabase | Импортирует резервную копию БД |

В таблице 3.4 представлены методы класса MainWindow.

Таблица 3.4 – Класс MainWindow

|  |  |
| --- | --- |
| Класс | MainWindow |
| Описание класса | Обеспечивает создание и отображение окна пользователя |
| Методы класса | Описание метода |
| MainWindow | Конструктор класса |
| Add\_Object | Событие добавления объекта в таблицу БД. Добавляет объект в выбранную таблицу из списка |
| UpdateObject | Событие изменения объекта в таблице БД |
| DeleteObject | Событие удаление объекта из таблицы БД. Удаляет объект в выбранной таблице. |
| RefreshData | Обновляет данные в таблице |
| LibraryView | Выводит выбранную таблицу на экран |
| ShowButtons | Показывает кнопки добавления, изменения, удаления |
| HideButtons | Скрывает кнопки добавления, изменения, удаления |

В таблице 3.5 представлены методы класса Officer\_panel.

Таблица 3.5 – Класс Officer\_panel

|  |  |
| --- | --- |
| Класс | Officer\_panel |
| Описание класса | Обеспечивает создание и отображение окна пользователя |
| Методы класса | Описание метода |
| Officer\_panel | Конструктор класса |
| Add\_Object | Событие добавления объекта в таблицу БД. Добавляет объект в выбранную таблицу из списка |
| UpdateObject | Событие изменения объекта в таблице БД |
| DeleteObject | Событие удаление объекта из таблицы БД. Удаляет объект в выбранной таблице. |
| RefreshData | Обновляет данные в таблице |
| LibraryView | Выводит выбранную таблицу на экран |

UML диаграммы данных классов приведены в приложении А. Подробный листинг вышеперечисленных классов приведен в приложении Б.

* 1. Даталогическое проектирование базы данных

В результате даталогического проектирования были созданы отношения (author, book, category, department, faculty, group, lib, librarian, press, scard, student, tcard, teacher, theme).

Далее приведены запросы, которые являются необходимыми для создания схемы базы данных:

CREATE TABLE [dbo].[Books](

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

[Name] [nvarchar](100) NOT NULL,

[Pages] [int] NOT NULL,

[YearPress] [int] NOT NULL,

[Theme] [nvarchar](100) NOT NULL,

[Category] [nvarchar](100) NOT NULL,

[Author] [nvarchar](100) NOT NULL,

[Press] [nvarchar](100) NOT NULL,

[Id\_Lib] [int] NOT NULL,

[Comment] [nvarchar](50) NULL,

[Quantity] [int] NOT NULL,

CONSTRAINT [PK\_Books] PRIMARY KEY CLUSTERED

( [Id] ASC)

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

) ON [PRIMARY]

CREATE TABLE [dbo].[Departments](

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

[Name] [nvarchar](40) NOT NULL,

CONSTRAINT [PK\_Departments] PRIMARY KEY CLUSTERED

( [Id] ASC)

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

) ON [PRIMARY]

CREATE TABLE [dbo].[Faculties](

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

[Name] [nvarchar](40) NOT NULL,

CONSTRAINT [PK\_Faculties] PRIMARY KEY CLUSTERED

( [Id] ASC)

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

) ON [PRIMARY]

CREATE TABLE [dbo].[Groups](

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

[Name] [nvarchar](40) NOT NULL,

[Id\_Faculty] [int] NOT NULL,

CONSTRAINT [PK\_Groups] PRIMARY KEY CLUSTERED

( [Id] ASC)

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

) ON [PRIMARY]

CREATE TABLE [dbo].[Libs](

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

[Name] [nvarchar](50) NOT NULL,

CONSTRAINT [PK\_Libs] PRIMARY KEY CLUSTERED

( [Id] ASC)

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

) ON [PRIMARY]

CREATE TABLE [dbo].[Students](

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

[FirstName] [nvarchar](25) NOT NULL,

[LastName] [nvarchar](35) NOT NULL,

[Id\_Group] [int] NOT NULL,

[Term] [int] NOT NULL,

CONSTRAINT [PK\_Students] PRIMARY KEY CLUSTERED

( [Id] ASC)

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

) ON [PRIMARY]

CREATE TABLE [dbo].[T\_Cards](

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

[Id\_Teacher] [int] NOT NULL,

[Id\_Book] [int] NOT NULL,

[DateOut] [datetime] NOT NULL,

[DateIn] [datetime] NULL,

[Id\_Lib] [int] NOT NULL,

[Id\_Librarian] INT NOT NULL,

CONSTRAINT [PK\_T\_Cards] PRIMARY KEY CLUSTERED

( [Id] ASC)

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

) ON [PRIMARY]

CREATE TABLE [dbo].[Teachers](

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

[FirstName] [nvarchar](25) NOT NULL,

[LastName] [nvarchar](35) NOT NULL,

[Id\_Dep] [int] NOT NULL,

CONSTRAINT [PK\_Teachers] PRIMARY KEY CLUSTERED

( [Id] ASC)

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

) ON [PRIMARY]

CREATE TABLE [dbo].[Users](

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

[FirstName] [nvarchar](50) NOT NULL,

[LastName] [nvarchar](50) NOT NULL,

[Email] [nvarchar](100) NOT NULL,

[Password] [nvarchar](50) NOT NULL,

CONSTRAINT [PK\_Users] PRIMARY KEY CLUSTERED

( [Id] ASC)

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

) ON [PRIMARY]

CREATE TABLE [dbo].[Admins](

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

[FirstName] [nvarchar](50) NOT NULL,

[LastName] [nvarchar](50) NOT NULL,

[Email] [nvarchar](100) NOT NULL,

[Password] [nvarchar](50) NOT NULL,

CONSTRAINT [PK\_Admins] PRIMARY KEY CLUSTERED

( [Id] ASC)

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

) ON [PRIMARY]

CREATE TABLE [dbo].[Officers](

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

[FirstName] [nvarchar](50) NOT NULL,

[LastName] [nvarchar](50) NOT NULL,

[Email] [nvarchar](100) NOT NULL,

[Password] [nvarchar](50) NOT NULL,

CONSTRAINT [PK\_Officers] PRIMARY KEY CLUSTERED

( [Id] ASC)

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

) ON [PRIMARY]

ALTER TABLE [dbo].[Books] WITH CHECK ADD CONSTRAINT [FK\_Books\_Author] FOREIGN KEY([Id\_Author])

REFERENCES [dbo].[Authors] ([Id])

GO

ALTER TABLE [dbo].[Books] CHECK CONSTRAINT [FK\_Books\_Author]

GO

ALTER TABLE [dbo].[Books] WITH CHECK ADD CONSTRAINT [FK\_Books\_Category] FOREIGN KEY([Id\_Category])

REFERENCES [dbo].[Categories] ([Id])

GO

ALTER TABLE [dbo].[Books] CHECK CONSTRAINT [FK\_Books\_Category]

GO

ALTER TABLE [dbo].[Books] WITH CHECK ADD CONSTRAINT [FK\_Books\_Press] FOREIGN KEY([Id\_Press])

REFERENCES [dbo].[Press] ([Id])

GO

ALTER TABLE [dbo].[Books] CHECK CONSTRAINT [FK\_Books\_Press]

GO

ALTER TABLE [dbo].[Books] WITH CHECK ADD CONSTRAINT [FK\_Books\_Lib] FOREIGN KEY([Id\_Lib])

REFERENCES [dbo].[Libs] ([Id])

GO

ALTER TABLE [dbo].[Books] CHECK CONSTRAINT [FK\_Books\_Theme]

GO

ALTER TABLE [dbo].[Groups] WITH CHECK ADD CONSTRAINT [FK\_Groups\_Faculty] FOREIGN KEY([Id\_Faculty])

REFERENCES [dbo].[Faculties] ([Id])

GO

ALTER TABLE [dbo].[Groups] CHECK CONSTRAINT [FK\_Groups\_Faculty]

GO

ALTER TABLE [dbo].[Librarians] WITH CHECK ADD CONSTRAINT [FK\_Librarians\_Lib] FOREIGN KEY([Id\_Lib])

REFERENCES [dbo].[Libs] ([Id])

GO

ALTER TABLE [dbo].[Librarians] CHECK CONSTRAINT [FK\_Librarians\_Lib]

GO

ALTER TABLE [dbo].[S\_Cards] WITH CHECK ADD CONSTRAINT [FK\_S\_Cards\_Book] FOREIGN KEY([Id\_Book])

REFERENCES [dbo].[Books] ([Id])

GO

ALTER TABLE [dbo].[S\_Cards] CHECK CONSTRAINT [FK\_S\_Cards\_Book]

GO

ALTER TABLE [dbo].[S\_Cards] WITH CHECK ADD CONSTRAINT [FK\_S\_Cards\_Libr] FOREIGN KEY([Id\_Librarian])

REFERENCES [dbo].[Librarians] ([Id])

GO

ALTER TABLE [dbo].[S\_Cards] CHECK CONSTRAINT [FK\_S\_Cards\_Libr]

GO

ALTER TABLE [dbo].[S\_Cards] WITH CHECK ADD CONSTRAINT [FK\_S\_Cards\_Lib] FOREIGN KEY([Id\_Lib])

REFERENCES [dbo].[Libs] ([Id])

GO

ALTER TABLE [dbo].[S\_Cards] CHECK CONSTRAINT [FK\_S\_Cards\_Lib]

GO

ALTER TABLE [dbo].[S\_Cards] WITH CHECK ADD CONSTRAINT [FK\_S\_Cards\_Stud] FOREIGN KEY([Id\_Student])

REFERENCES [dbo].[Students] ([Id])

GO

ALTER TABLE [dbo].[S\_Cards] CHECK CONSTRAINT [FK\_S\_Cards\_Stud]

GO

ALTER TABLE [dbo].[Students] WITH CHECK ADD CONSTRAINT [FK\_Students\_Group] FOREIGN KEY([Id\_Group])

REFERENCES [dbo].[Groups] ([Id])

GO

ALTER TABLE [dbo].[Students] CHECK CONSTRAINT [FK\_Students\_Group]

GO

ALTER TABLE [dbo].[T\_Cards] WITH CHECK ADD CONSTRAINT [FK\_T\_Cards\_Book] FOREIGN KEY([Id\_Book])

REFERENCES [dbo].[Books] ([Id])

GO

ALTER TABLE [dbo].[T\_Cards] CHECK CONSTRAINT [FK\_T\_Cards\_Book]

GO

ALTER TABLE [dbo].[T\_Cards] WITH CHECK ADD CONSTRAINT [FK\_T\_Cards\_Lib] FOREIGN KEY([Id\_Lib])

REFERENCES [dbo].[Libs] ([Id])

GO

ALTER TABLE [dbo].[T\_Cards] CHECK CONSTRAINT [FK\_T\_Cards\_Lib]

GO

ALTER TABLE [dbo].[T\_Cards] WITH CHECK ADD CONSTRAINT [FK\_T\_Cards\_Libr] FOREIGN KEY([Id\_Librarian])

REFERENCES [dbo].[Librarians] ([Id])

GO

ALTER TABLE [dbo].[T\_Cards] CHECK CONSTRAINT [FK\_T\_Cards\_Libr]

GO

ALTER TABLE [dbo].[T\_Cards] WITH CHECK ADD CONSTRAINT [FK\_T\_Cards\_Teacher] FOREIGN KEY([Id\_Teacher])

REFERENCES [dbo].[Teachers] ([Id])

GO

ALTER TABLE [dbo].[T\_Cards] CHECK CONSTRAINT [FK\_T\_Cards\_Teacher]

GO

ALTER TABLE [dbo].[Teachers] WITH CHECK ADD CONSTRAINT [FK\_Teachers\_Dep] FOREIGN KEY([Id\_Dep])

REFERENCES [dbo].[Departments] ([Id])

GO

ALTER TABLE [dbo].[Teachers] CHECK CONSTRAINT [FK\_Teachers\_Dep]

GO

* 1. Вывод по главе

В данной главе был обоснован выбор средств разработки программного средства. Кроме того, была описана структура программного средства, которая представляет из себя набор классов. В таблицах были приведены описания наиболее важных методов данных классов. Также было осуществлено даталогическое проектирование базы данных.

1. Тестирование программного средства

В данном разделе проводится тестирование программного средства на основе различных входных данных с подтверждающими работу программного средства рисунками.

1. 1. Описание процесса тестирования

После запуска программы на экране появляется окно для регистрации нового пользователя, в котором пользователь должен ввести имя, фамилию, почту(логин) и пароль (Рисунок 4.1).

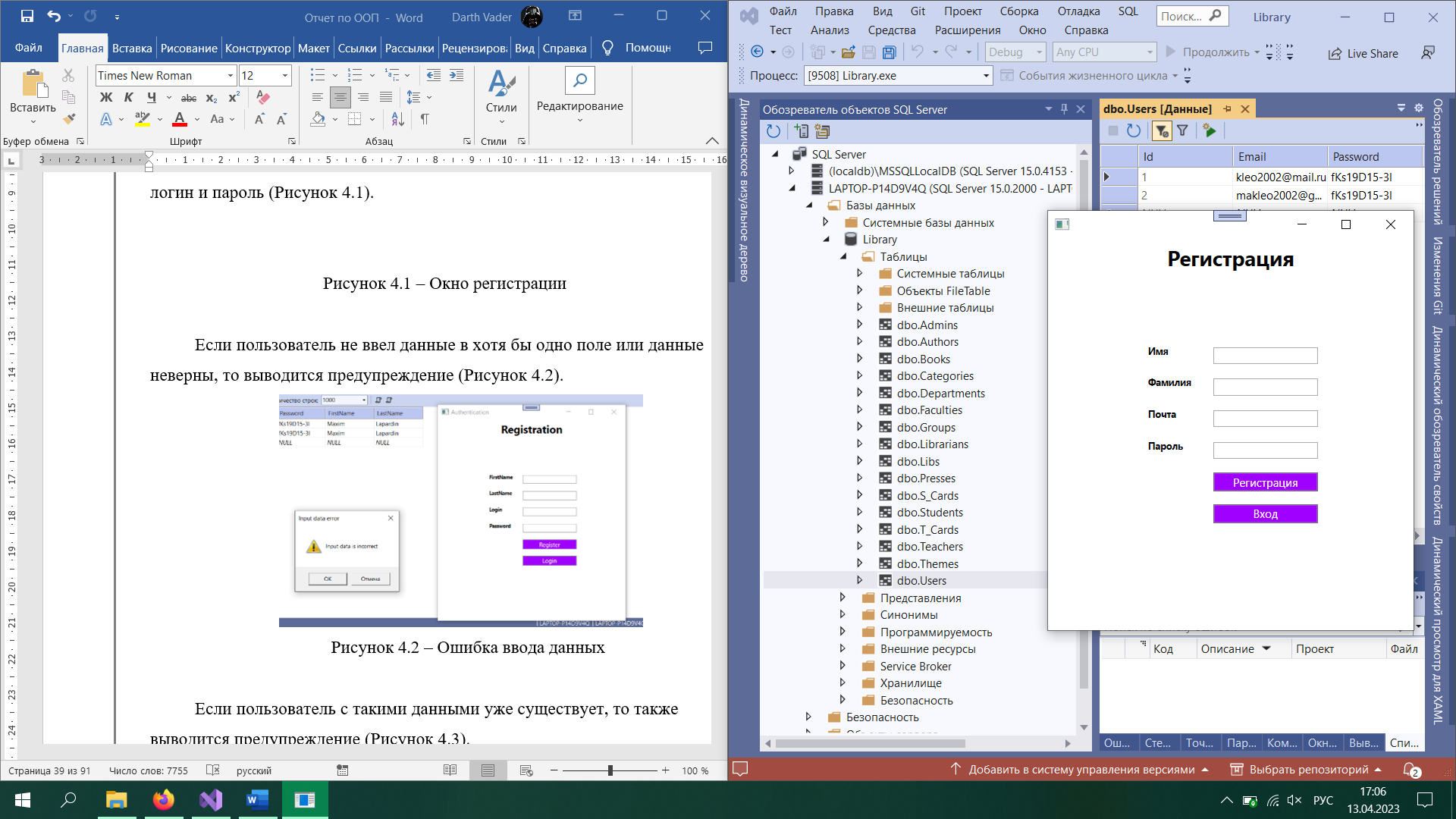


Рисунок 4.1 – Окно регистрации

Если пользователь не ввел данные в хотя бы одно поле или данные неверны, то выводится предупреждение (Рисунок 4.2).

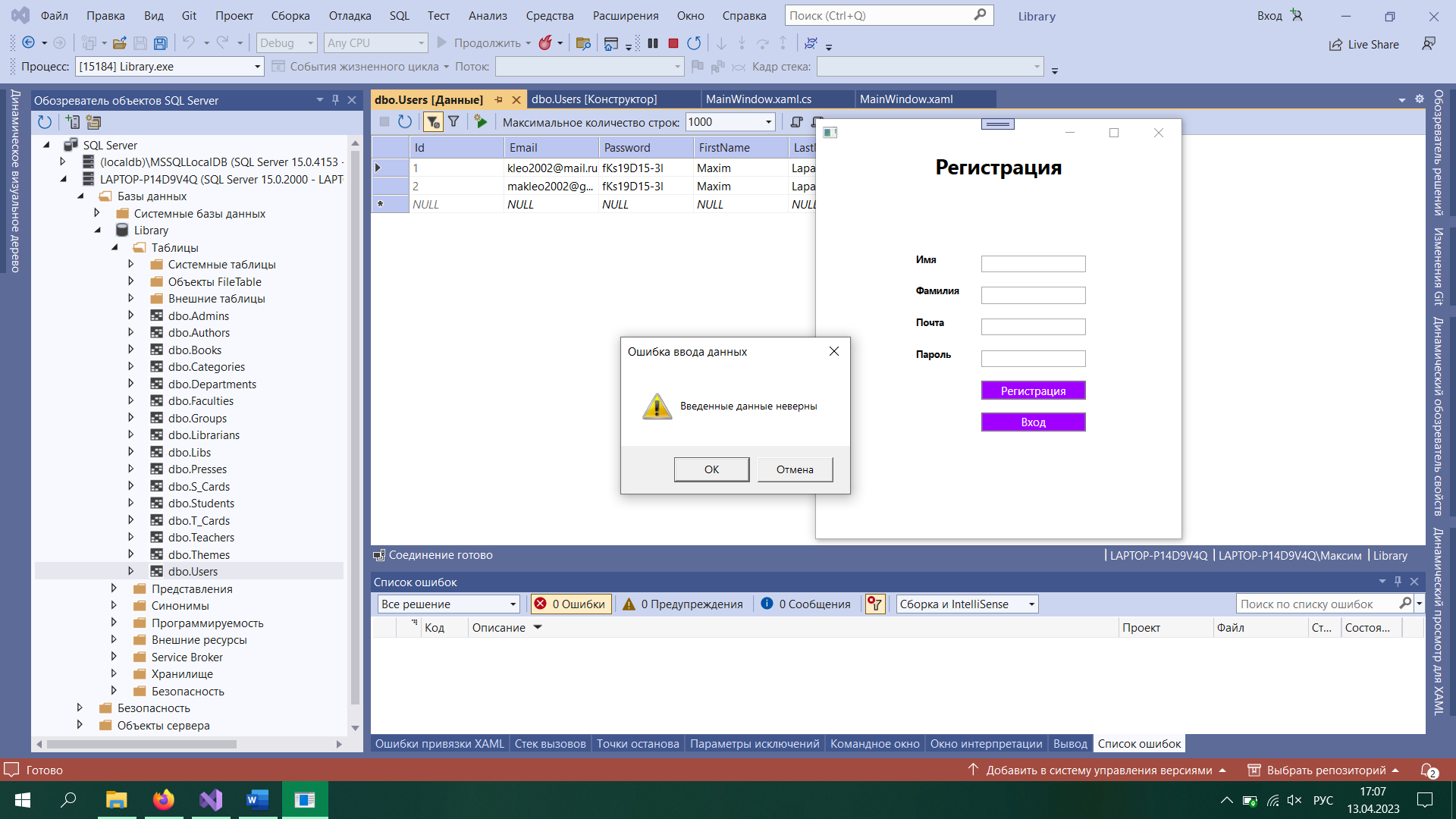


Рисунок 4.2 – Ошибка ввода данных

Если пользователь с такими данными уже существует, то также выводится предупреждение (Рисунок 4.3).

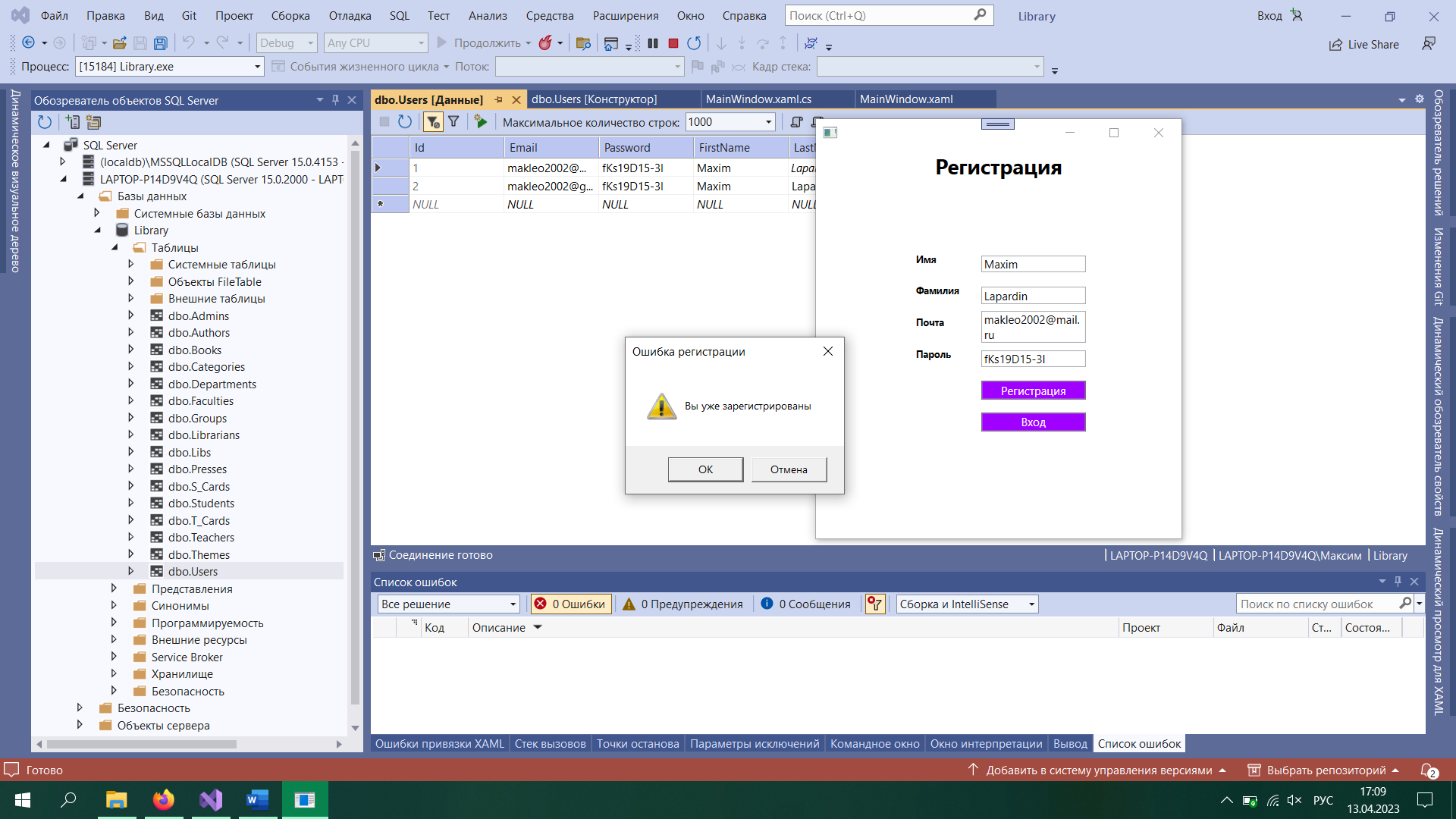


Рисунок 4.3 – Ошибка регистрации

Если пользователь уже зарегистрирован в приложении, то он нажимает кнопку «Вход» и ему открывается новое окно авторизации (Рисунок 4.4).

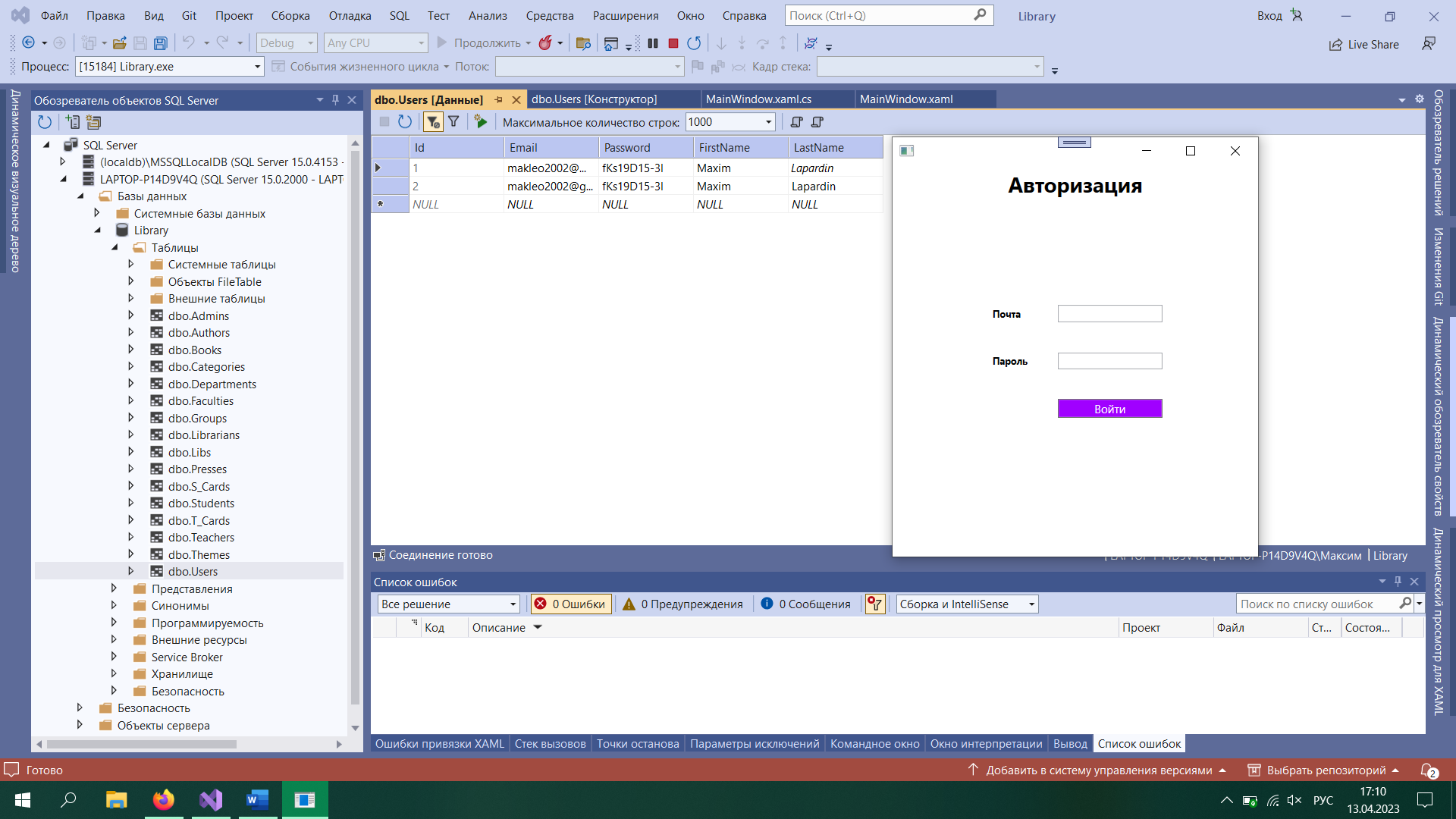


Рисунок 4.4 – Окно авторизации

Если пользователь ввел неверные данные или не заполнил поля, то выводится предупреждение (Рисунок 4.5).

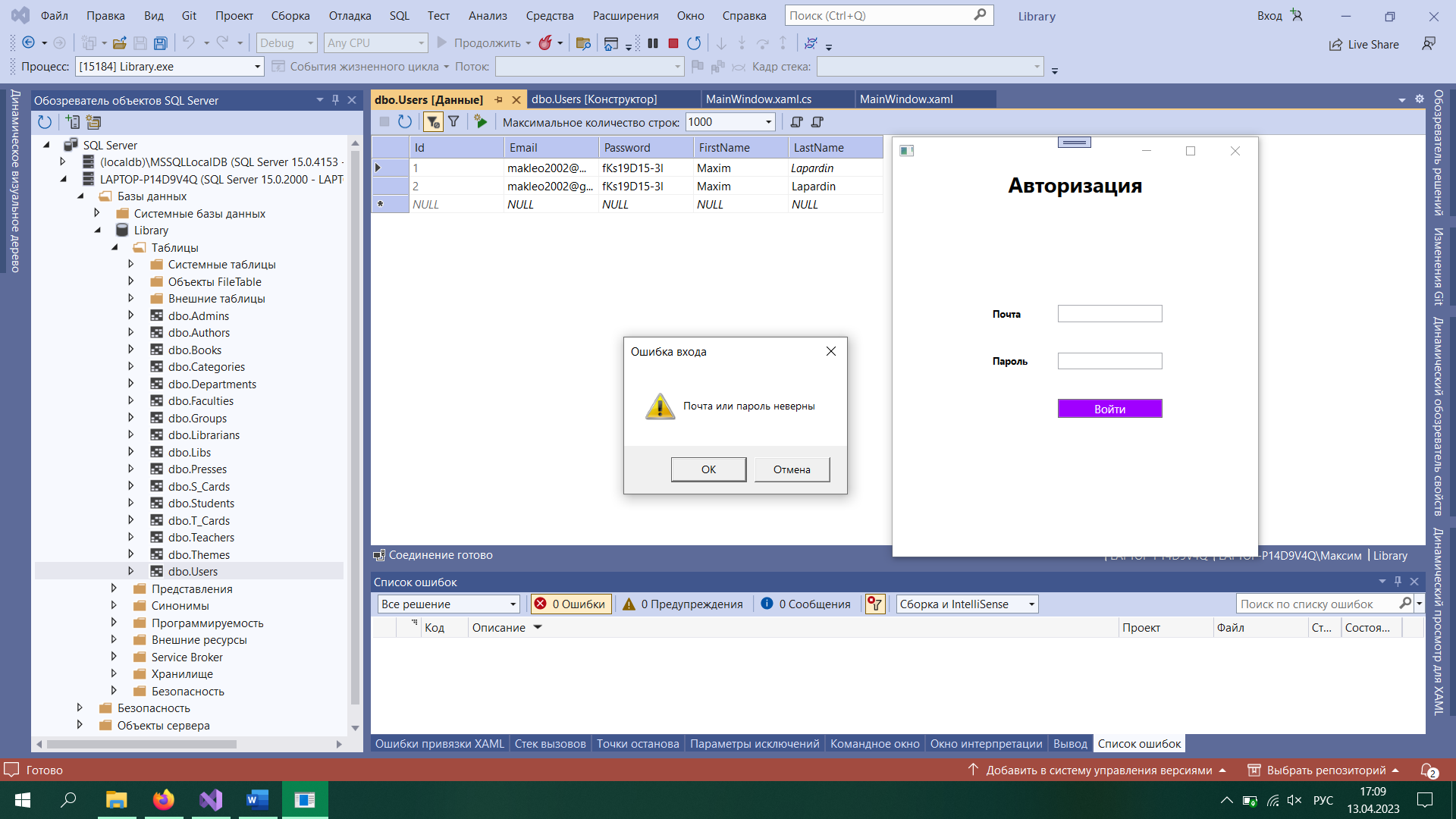


Рисунок 4.5 – Ошибка входа

Если пользователь ввел верные данные, то выводится соответствующее сообщение и открывается окно главной программы (Рисунок 4.6).

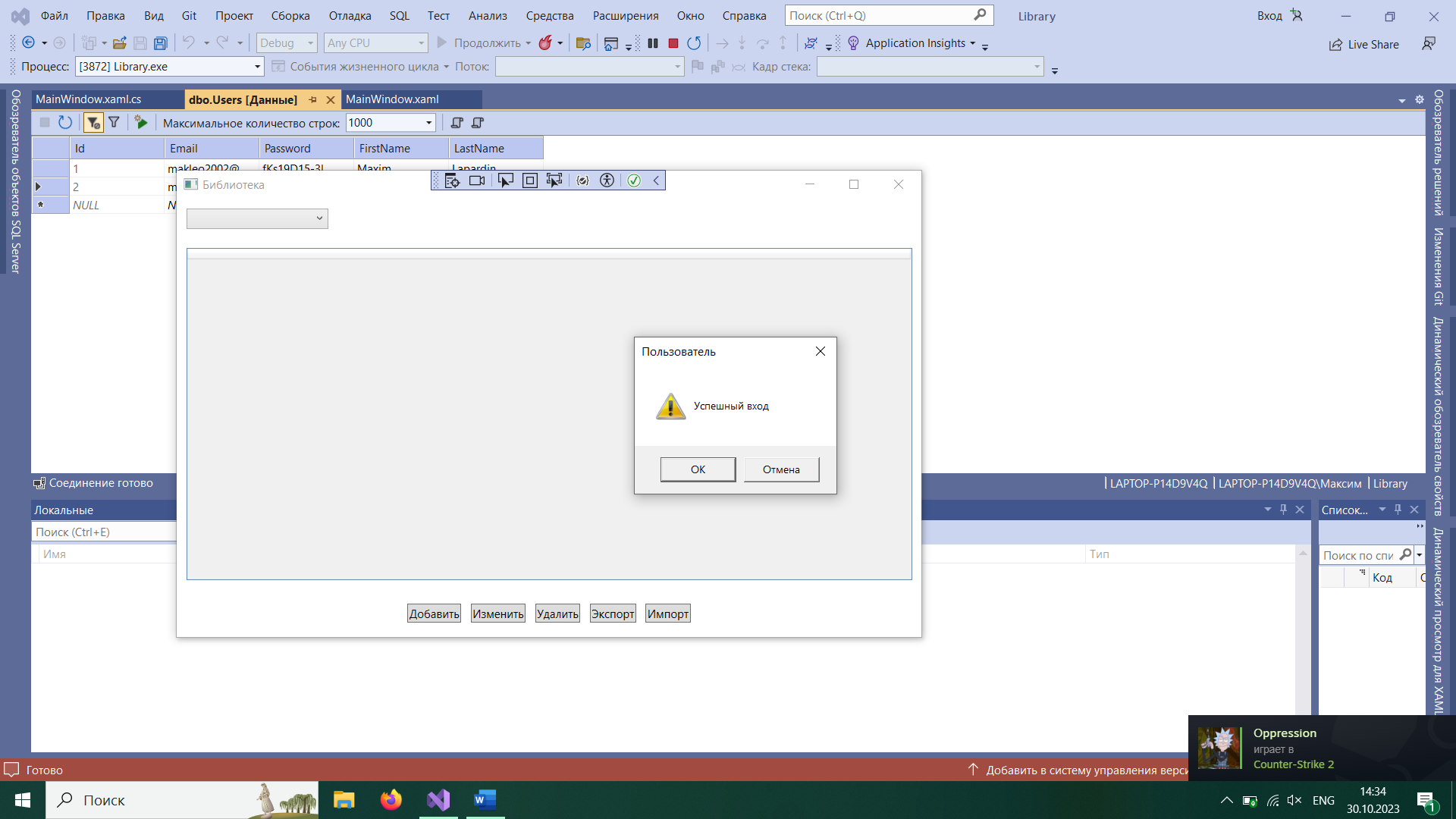


Рисунок 4.6 – Успешный вход пользователя

Если администратор ввел верные данные, то выводится соответствующее сообщение и открывается панель администратора (Рисунок 4.7).

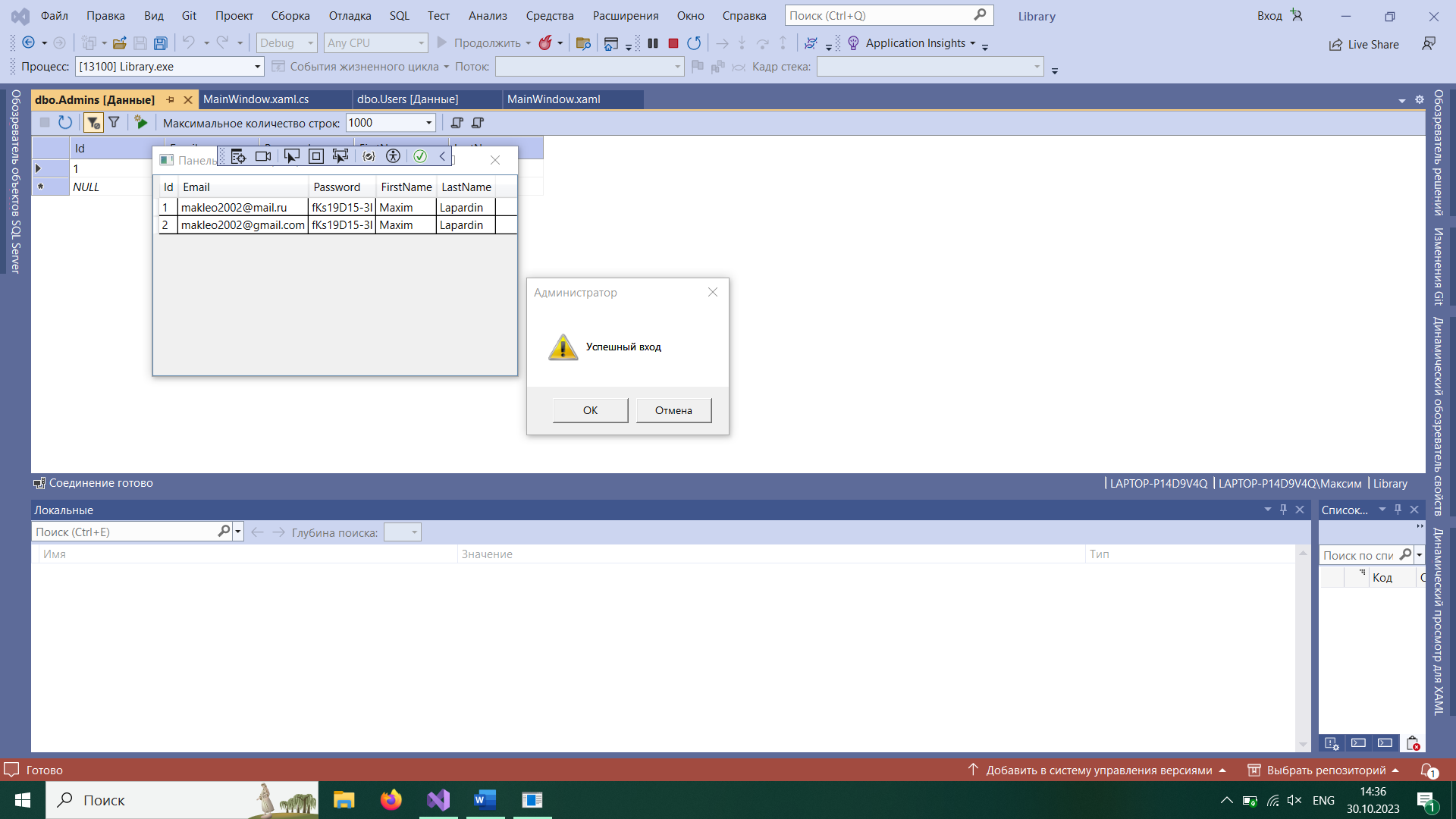


Рисунок 4.7 – Успешный вход администратора

Когда мы выбираем таблицу из списка, она выводится снизу. Для примера выберем таблицу «Libs» (Рисунок 4.8).

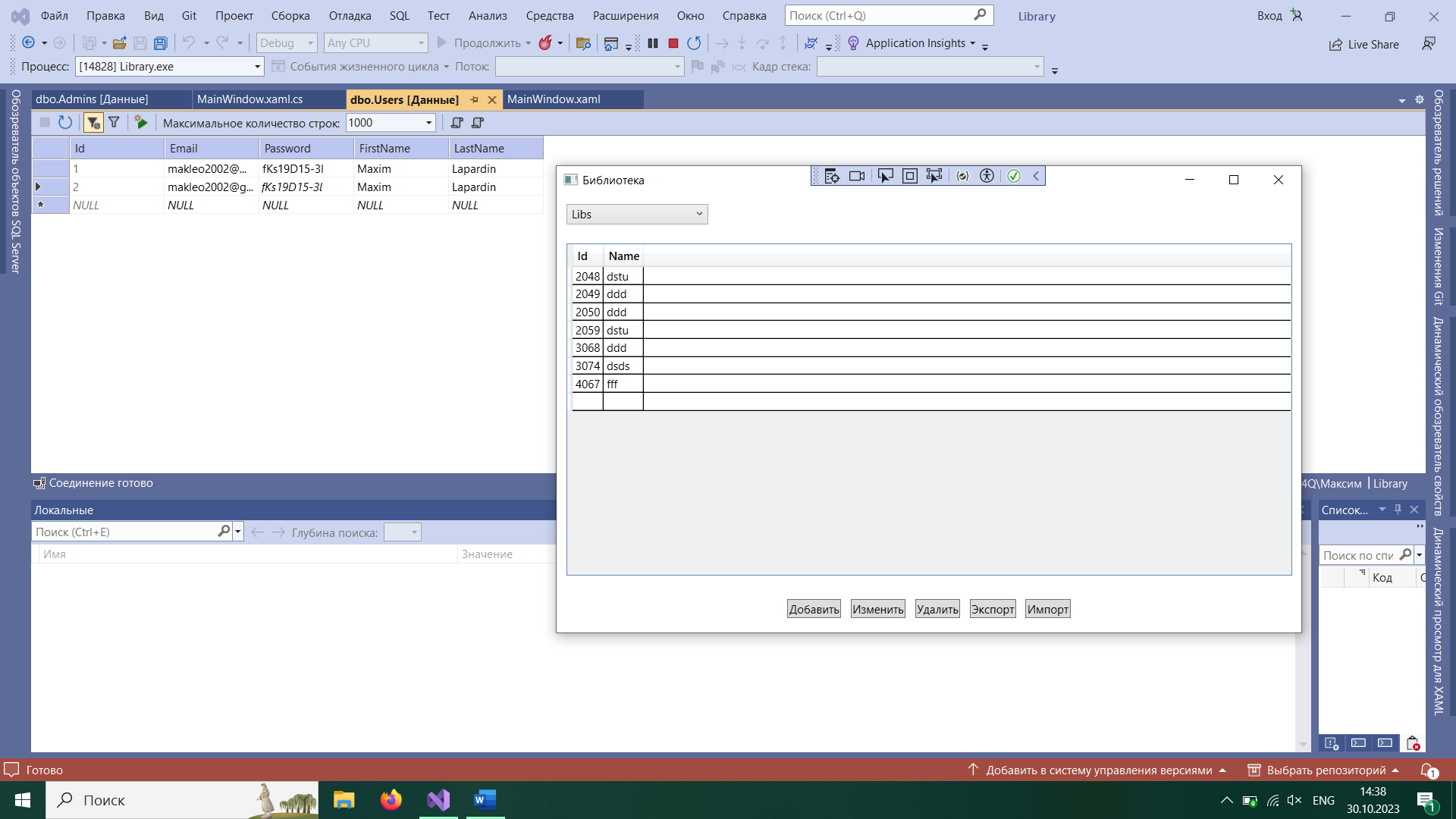


Рисунок 4.8 – Таблица «Libs»

Добавим в таблицу «Libs» новый объект с именем «Library» (Рисунок 4.9).

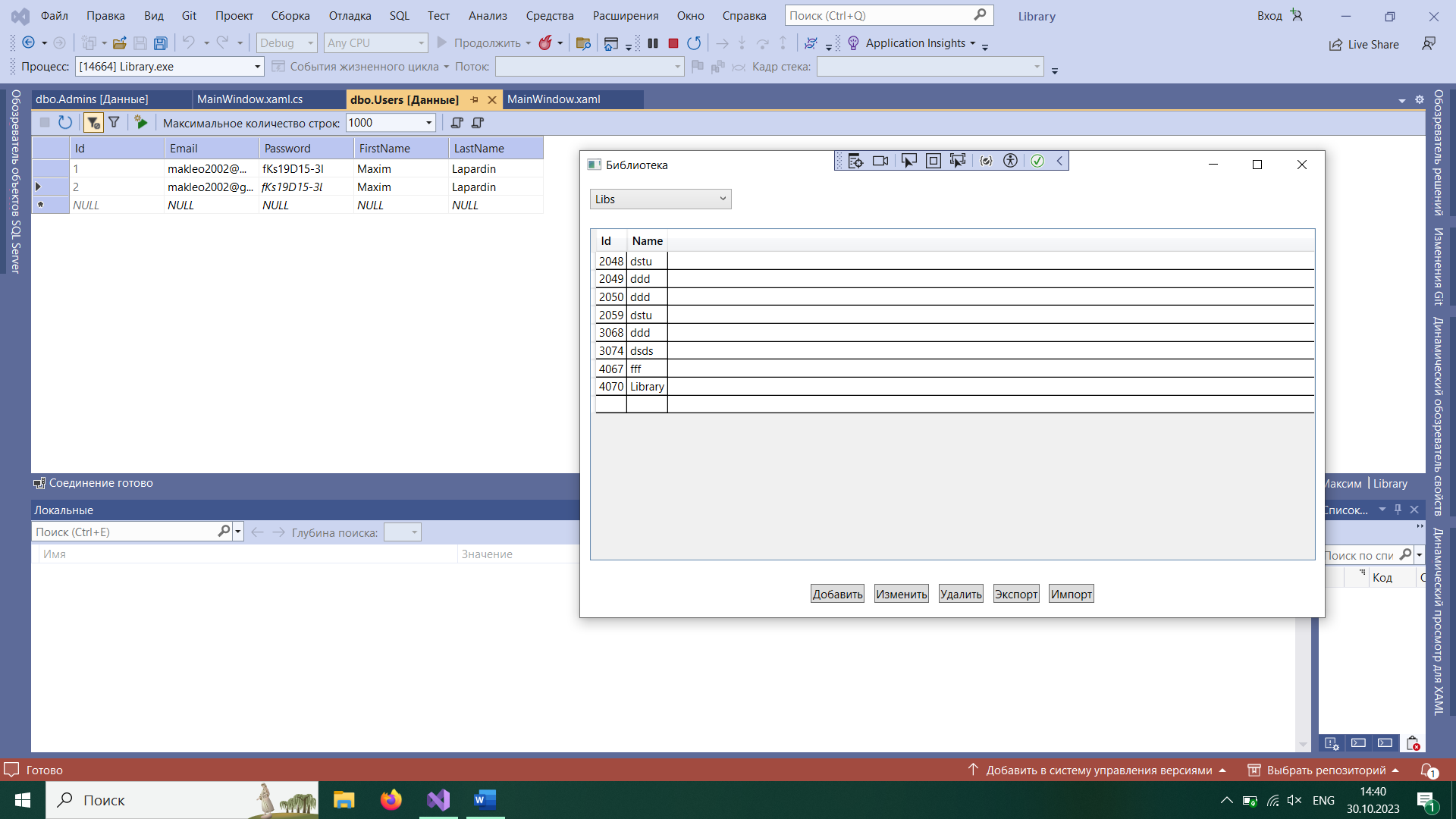


Рисунок 4.9 – Добавление элемента в таблицу «Libs»

Изменим имя объекта «Library» в таблице «Libs» на имя «Library1» (Рисунок 4.10).

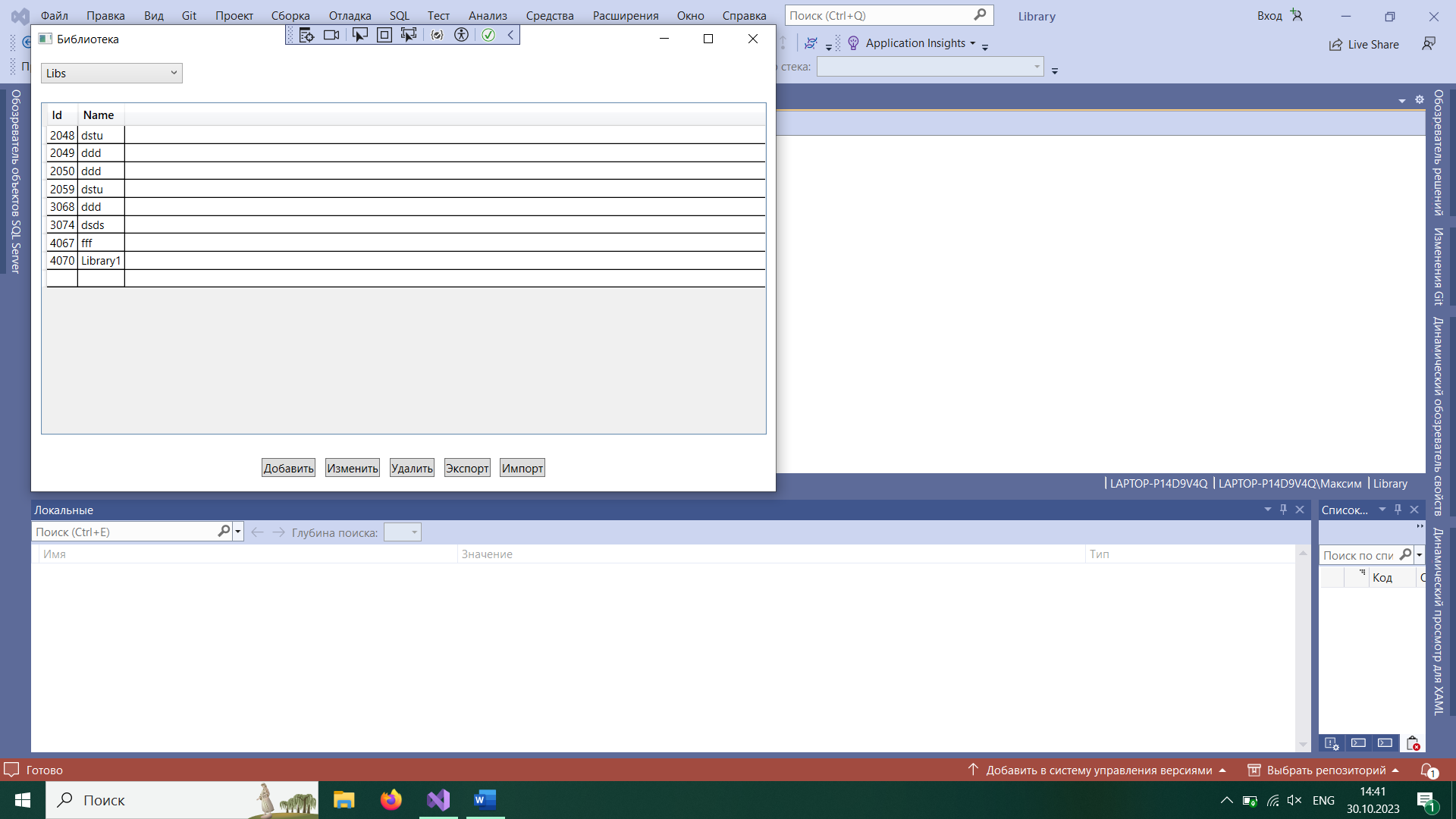


Рисунок 4.10 – Изменение элемента в таблице «Libs»

Удалим из таблицы «Libs» объект с именем «Library1» (Рисунок 4.11).

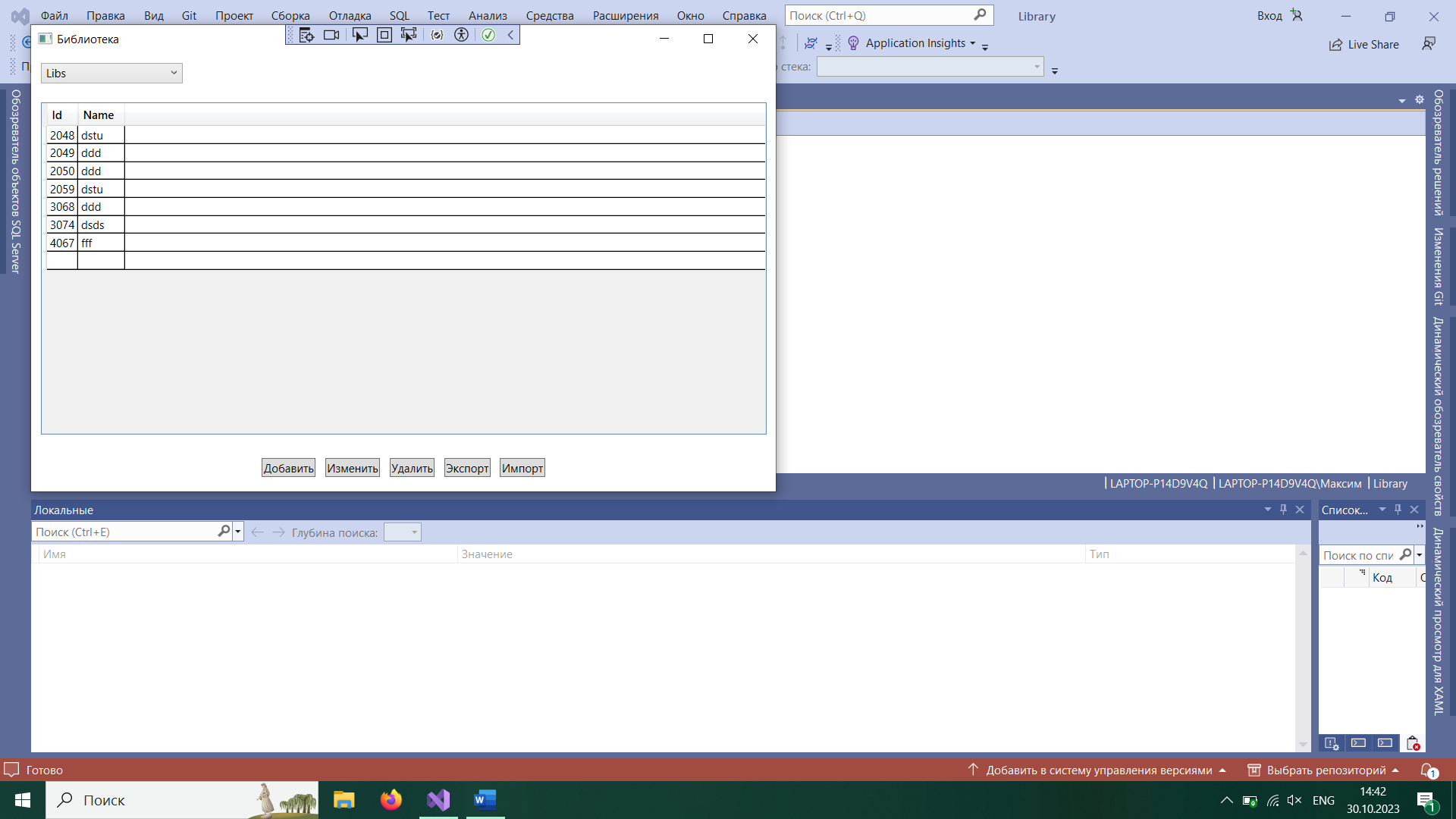


Рисунок 4.11 – Удаление элемента в таблице «Libs»

* 1. Вывод по главе

В данной главе была показана работа программного средства. Программное средство реализует все заявленные функции. Программное средство работает стабильно.

Заключение

В ходе выполнения курсовой работы были решены следующие задачи:

* проведен детальный обзор существующих систем управления библиотекой, что позволило выделить основные требования. Сравнительный анализ был проведен с целью выявить преимущества и недостатки конкурирующих решений;
* Разработаны и реализованы алгоритмы, обеспечивающие правильное функционирование программного средства. Это включает в себя логику работы методов классов, описанных в разделе алгоритмов;
* Обоснован выбор языка программирования C# и среды разработки Visual Studio (.NET Framework) с учетом требований проекта. Эти решения обеспечивают эффективную разработку и высокую производительность программного средства;
* Разработаны и реализованы классы, описанные в структуре программы. Особое внимание уделено методам классов Register, Authorize, Admin\_panel, MainWindow, которые отвечают за регистрацию, авторизацию, работу администратора и главное окно пользователя. Приведены UML-диаграммы для визуализации структуры классов;
* В ходе тестирования проведены проверки работы программы с различными входными данными. Изображены ключевые моменты тестирования, включая успешную регистрацию, авторизацию пользователя и администратора, а также операции добавления, изменения и удаления данных из базы;

Таким образом, цель по созданию функционального и стабильного программного средства для управления библиотекой успешно достигнута. В результате выполнения курсовой работы была создана работающая и эффективная система, полностью соответствующая поставленным задачам.

Перечень использованных информационных ресурсов

1. Автоматизированные информационные системы (АИС):   
   URL: https://infotaktika.ru/resheniya-i-uslugi/informaczionnoe-vzaimodejstvie/avtomatizirovannyie-informaczionnyie-sistemyi/ (дата обращения 25.10.2022 г.)
2. Библиотека: URL: https://pandia.ru/154138/ (дата обращения 25.10.2022 г.)
3. Библиотека и база данных:

URL: https://files.student-it.ru/previewfile/166699 (дата обращения 26.10.2022 г.)

1. Visual Studio:

URL: https://en.wikipedia.org/wiki/Visual\_Studio (дата обращения 27.10.2022 г.)

1. Microsoft SQL Server:

URL: <https://navicongroup.ru/platforms/4025/> (дата обращения 27.10.2022 г.)

1. WPF(Windows Presentation Foundation):

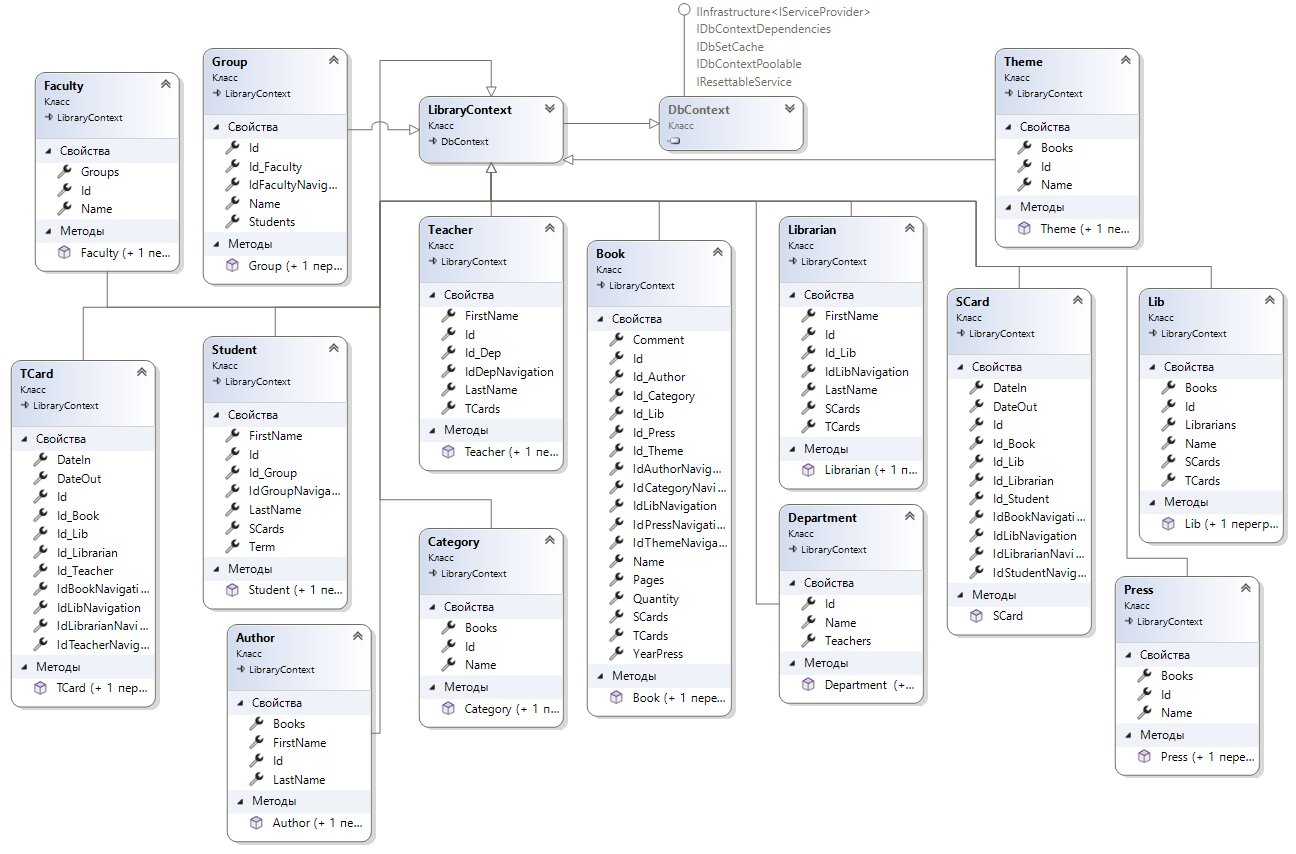
URL:  [https://en.wikipedia.org/wiki/Windows\_Presentation\_Foundation](https://ru.wikipedia.org/wiki/SQL) (дата обращения 28.10.2022 г.)

1. C# – Энциклопедия языков программирования:

URL: <http://progopedia.ru/language/csharp/>(дата обращения 28.10.2022 г.)

Приложение А

UML-диаграмма классов из пакета «Models»



Приложение Б

Листинг основных классов программного средства

Полный код программного средства находится на съемном носителе, прилагаемом к данной работе.

using System;

using System.Windows;

using System.Windows.Controls;

using System.Data;

using System.Data.SqlClient;

using Microsoft.Win32;

namespace Library

{

public partial class MainWindow : Window

{

private string connectionString;

private DataTable dataTable;

public MainWindow()

{

InitializeComponent();

connectionString = @"Data Source=LAPTOP-P14D9V4Q;Initial Catalog=Library; Integrated Security=True";

// LibraryGrid.AutoGeneratingColumn += LibraryGrid\_AutoGeneratingColumn;

}

private void LibraryView(object sender, SelectionChangedEventArgs e)

{

string selectedTable = ((ComboBoxItem)Tables.SelectedItem).Content.ToString(); // Получаем имя выбранной таблицы

string query = "SELECT \* FROM " + selectedTable;

try

{

using (SqlConnection connection = new SqlConnection(connectionString))

{

connection.Open();

SqlDataAdapter dataAdapter = new SqlDataAdapter(query, connection);

dataTable = new DataTable();

dataAdapter.Fill(dataTable);

LibraryGrid.ItemsSource = dataTable.DefaultView;

}

}

catch (Exception ex)

{

MessageBox.Show("Ошибка при выполнении запроса: " + ex.Message);

}

}

private void AddObject(object sender, RoutedEventArgs e)

{

try

{

using (SqlConnection connection = new SqlConnection(connectionString))

{

connection.Open();

// Set the SelectCommand property for dataAdapter

SqlDataAdapter dataAdapter = new SqlDataAdapter("SELECT \* FROM " + ((ComboBoxItem)Tables.SelectedItem).Content, connection);

dataAdapter.UpdateCommand = new SqlCommandBuilder(dataAdapter).GetUpdateCommand();

try

{

dataAdapter.Update(dataTable);

}

catch (Exception)

{

string messageBoxText = "Неправильно введено Id";

string caption = "Ошибка ввода" + Tables.SelectedItem.ToString();

MessageBoxButton button = MessageBoxButton.OKCancel;

MessageBoxImage icon = MessageBoxImage.Warning;

MessageBoxResult result = MessageBox.Show(messageBoxText, caption, button, icon);

}

RefreshData();

}

}

catch (Exception ex)

{

MessageBox.Show("Ошибка при выполнении запроса: " + ex.Message);

}

}

private void UpdateObject(object sender, RoutedEventArgs e)

{

DataRowView selectedRow = LibraryGrid.SelectedItem as DataRowView;

if (selectedRow != null)

{

try

{

using (SqlConnection connection = new SqlConnection(connectionString))

{

connection.Open();

// Set the SelectCommand property for dataAdapter

SqlDataAdapter dataAdapter = new SqlDataAdapter("SELECT \* FROM " + ((ComboBoxItem)Tables.SelectedItem).Content, connection);

dataAdapter.UpdateCommand = new SqlCommandBuilder(dataAdapter).GetUpdateCommand();

dataAdapter.Update(dataTable);

RefreshData();

}

}

catch (Exception ex)

{

MessageBox.Show("Ошибка при выполнении запроса: " + ex.Message);

}

}

}

private void DeleteObject(object sender, RoutedEventArgs e)

{

DataRowView selectedRow = LibraryGrid.SelectedItem as DataRowView;

if (selectedRow != null)

{

try

{

using (SqlConnection connection = new SqlConnection(connectionString))

{

connection.Open();

selectedRow.Row.Delete();

// Set the SelectCommand property for dataAdapter

SqlDataAdapter dataAdapter = new SqlDataAdapter("SELECT \* FROM " + ((ComboBoxItem)Tables.SelectedItem).Content, connection);

dataAdapter.UpdateCommand = new SqlCommandBuilder(dataAdapter).GetUpdateCommand();

dataAdapter.Update(dataTable);

RefreshData();

}

}

catch (Exception ex)

{

MessageBox.Show("Ошибка при выполнении запроса: " + ex.Message);

}

}

}

private void RefreshData()

{

string selectedTable = ((ComboBoxItem)Tables.SelectedItem).Content.ToString();

string query = "SELECT \* FROM " + selectedTable;

try

{

using (SqlConnection connection = new SqlConnection(connectionString))

{

connection.Open();

SqlDataAdapter dataAdapter = new SqlDataAdapter(query, connection);

dataTable.Clear();

dataAdapter.Fill(dataTable);

LibraryGrid.ItemsSource = dataTable.DefaultView;

}

}

catch (Exception ex)

{

MessageBox.Show("Ошибка при выполнении запроса: " + ex.Message);

}

}