Содержание.

[Постановка задачи 4](#_Toc130050118)

[Этапы реализации 6](#_Toc130050119)

[1. Средства разработки 7](#_Toc130050120)

[1.1. СУБД «PostgreSQL» 7](#_Toc130050121)

[1.2. Язык программирования С# 7](#_Toc130050122)

[2. Структура БД. 9](#_Toc130050123)

[2.1. Предметная область 9](#_Toc130050124)

[2.2. Логическая и физическая модель 9](#_Toc130050125)

[2.3. Описание сущностей БД 9](#_Toc130050126)

[3. Программный продукт 11](#_Toc130050127)

[Выводы 16](#_Toc130050128)

[Список литературы 17](#_Toc130050129)

[Приложение 1 18](#_Toc130050130)

[Приложение 2 19](#_Toc130050131)

# Постановка задачи

Создание системы связано с обеспечением работы непрерывного общения между посетителями и библиотекой. Создание такой системы позволит пользователям дистанционно получать информацию о книгах, хранящихся в фонде библиотеки.

Система предназначена:

* для определения наличия или отсутствия данной книги в библиотеке;
* для оформления заказа книги для получения в библиотеке;
* для ознакомления с книгами, находящимися в фонде библиотеки.

Цели создания системы:

* сокращение времени поиска книги в библиотеке для разных групп пользователей;
* обеспечивать сохранность печатного материала фондов библиотек
* создавать условия для предоставления локального и удаленного доступа пользователей к информационным ресурсам;
* предоставлять пользователям расширяющийся спектр информационных услуг обеспечение доступа к информации

В системе предусмотрено 2 вида роли: пользователь и администратор. К функциям роли пользователя относится возможность просмотра каталога библиотеки и возможность делать заказы на получение книги Функции роли администратор: возможность редактирования/удаления/добавления пользователей и возможность изменения статуса заказа пользователя.

Вся система состоит из 5 основных модулей: управление пользователями, мои заказы, управление заказами, просмотр изданий, отчеты и аналитика. Также в систему входит вспомогательный модуль Хранение данных.

Каждый модуль должен удовлетворять требованиям, которые отображают функционал системы относительно него.

Модуль «Управление пользователями» содержит в себе следующие функциональные возможности:

1. добавление / удаление пользователя;
2. редактирование профиля пользователя;
3. управление атрибутами пользователя, содержащими следующие сведения:
   * идентификационный номер (присваивается автоматически, при создании пользователя);
   * дата создания (присваивается автоматически, при создании пользователя);
   * имя пользователя (логин при авторизации);
   * е-mail;
   * имя;
   * фамилия;
   * отчество;
   * телефон
   * дата получения абонемента
   * дата окончания абонемента

Модуль «Управление заказами» содержит в себе следующие функциональные возможности:

1. изменение статуса заказа;
2. возможность отмены заказа пользователя.

Модуль «Мои заказы» содержит в себе следующие функциональные возможности: просмотр своих заказов в системе и читательского билета. Также предоставлять информацию для модуля управление заказами.

Модуль «Отчеты и аналитика» содержит в себе следующие функциональные возможности по формированию отчета. В рамках предметной области системы выделены 5 отчетов (таб. 2).

Таблица 2 – Виды отчетов и формат вывода

|  |  |
| --- | --- |
| Название отчета | Формат вывода отчета |
| Отчет по кол-ву пользователей в системе | doc |
| Отчет по действиям (заказам) пользователей в системе | doc |
| Отчет по кол-ву изданий на каждое произведение | doc |
| Отчет по книгам, находящихся у пользователей. | doc |
| Отчет по отмененным и выполненным заказам | doc |

Модуль «Просмотра изданий» содержит в себе следующие функциональные возможности по просмотру информации о авторе произведения, произведение и доступных изданий в библиотеке.

# Этапы реализации

На основании сформулированного задания были выделены 5 основных этапов реализации системы «Каталога для библиотеки» и определены сроки начала и окончания выполнения каждого этапа. Также выбрана форма отчетности после каждого этапа в виде отчета в формате doc. Подробная информация о сроках отображена в таблице 1.

Таблица 1 – Сроки реализации.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Название этапа | Сроки | | Форма отчетности |
| Дата начала | Дата окончания |
|  | Проектирование хранилища данных | 9.02.23 | 16.02.23 | Отчет |
|  | Проектирование интерфейса | 16.02.23 | 23.02.23 | Отчет |
|  | Разработка структур классов | 23.02.23 | 2.03.23 | Отчет |
|  | Программа реализации ИС | 2.03.23 | 30.03.23 | Отчет |
|  | Тестирование | 30.03.23 | 6.04.23 | Отчет |

# Средства разработки

В предыдущем пункте были выделены основные этапы реализации разрабатываемого программного продукта. Для работы с базой данных была выбрана СУБД «PostgreSQL» и для написания кода – язык программирования С# в среде разработки Visual Studio.

## СУБД «PostgreSQL»

Программный продукт выбран по причине того, что это одна из баз данных, которая была изучена и интересна для практического использования в качестве хранилища данных для разрабатываемого продукта. Также стоит отметить, что интерфейс данного приложения удобен в использовании.

Система управления базой данных «PostgreSQL» необходима для операций над данными, например, выборка, запись или обработка. PostgreSQL — это система управления реляционными базами данных (СУБД) с открытым исходным кодом, то есть она содержит данные, которые связаны между собой, и эту связь можно изобразить в виде таблиц [2].

Система управления базой данных имеет клиент-серверную архитектуру. Это означает, что база данных хранится в одном источнике — на сервере. А связываться с ней могут клиенты — сторонние устройства. Клиенты отправляют запросы к БД, а потом получают от сервера информацию. В основном PostgreSQL используется для работы со средними и небольшими проектами. Она имеет открытый исходный код, поэтому любой разработчик может с ним ознакомиться.

Преимущества PostgreSQL [1]:

* Свободный доступ (Любой пользователь может бесплатно использовать СУБД)
* Можно установить на любую платформу
* Поддерживает разные форматы данных (PostgreSQL поддерживает много разных типов данных, к ним относится сетевые адреса , данные хранящиеся в формате JSON и геометрические данные для координат геопозиций и тд. Также имеется возможность создавать пользовательские типы данных, чтобы упростить работу с БД или установить ограничения)
* Позволяет работать с большими размерами данных (Размер БД зависит от кол-ва сколько свободной памяти есть в месте хранения)
* Соответствует требованиям ACID (СУБД соответсвует четырем требованиям для надежной работы систем: атомарность, согласованность, изолированность, устойчивость)
* Поддерживает все функции, которые есть в современных базах данных
* Есть свой диалект языка SQL
* Можно настроить синхронное дублирование данных
* Можно без потерь перенести данные из другой СУБД

Как любой программный продукт PostgreSQL имеет некоторые недостатки [3]. Во-первых, при выполнении более простых операций она может работать медленнее, чем ее «конкуренты» (MySQL, MSSQL и Oracle Database). Во-вторых, низкая скорость разработки, следовательно, PostgreSQL не достает некоторого технического совершенства, что может отразится на разработке.

## Язык программирования С#

Программный продукт реализован под средством языкового программирования С#. Выбор данного языка обусловлен простотой и удобством создаваемого интерфейса с помощью компонентов windows Form, понятным синтаксисом, который упрощает разработку программы. Помимо этого, выбор языка связан с минимизацией затрат при использовании дополнительных программных компонентов. Также язык С# универсален и используется в создание любого ПО. Ещё одна причина выбора – это простота в подключении базы данных к среде разработки Visual Studio для создания интерфейса.

Данный язык разработан компанией Майкрософт, один из самых популярных современных языков программирования. Язык С# появился как внутренний язык для платформы Microsoft .NET, постепенно язык начали развивать, в следствии чего стал одним из распространенных языков программирования. Также стоит отметить, что компания Microsoft уделяет значительное внимание поддержке языка разработки, а потому очень часто появляются дополнения и обновление программных компонентов [4].

У данного я зыка есть ряд преимуществ перед другими языками:

* Сборка мусора – это очистка памяти от неиспользуемых объектов
* Обработка исключений – это инструмент для выявления о обработки ошибок в коде. Среда сама может указать как можно исправить данную ошибку и вывести причину ошибки.
* Единая система типов
* Управление версиями. В случае если программа написана в среде другой версии и не запускается, то можно исправить это изменив текущую версию.

# Структура БД.

## Предметная область

. В качестве предметной области была выбрана тема система каталога для библиотеки и под средством этого иметь возможность делать заказы книг.

Система проектируется для сущности пользователь, который будет использовать систему и сможет с помощью сущности выдача оформить несколько заказов на книги для получения в библиотеке. Пользователь может взять одно или несколько изданий, т. е. книг.

У сущности произведение есть несколько изданий, также одно издание может содержать в себе несколько произведений. Также у произведения может быть несколько авторов, также у разных авторов может быть одинаковое название произведений.

У сущности издание есть связь со сущностью издательство, которое публикует книгу в печатном экземпляре, как правило издательство может выпускать огромное кол-во книг.

Сущность город связана с сущностью издательство, так как в одном городе может находится несколько издательств.

Сущность корзина связана с сущностью издание и пользователь, так как один пользователь может добавить в корзину несколько изданий.

Сущность статус заказа связана с сущностью выдача, у одного заказа есть определенный статус и дата хранения.

Сущность роль пользователя связана с пользователем, так как у пользователя может быть только один статус.

## Логическая и физическая модель

На рисунке 1 изображена спроектированная логическая модель, построенная в diagrams.net.

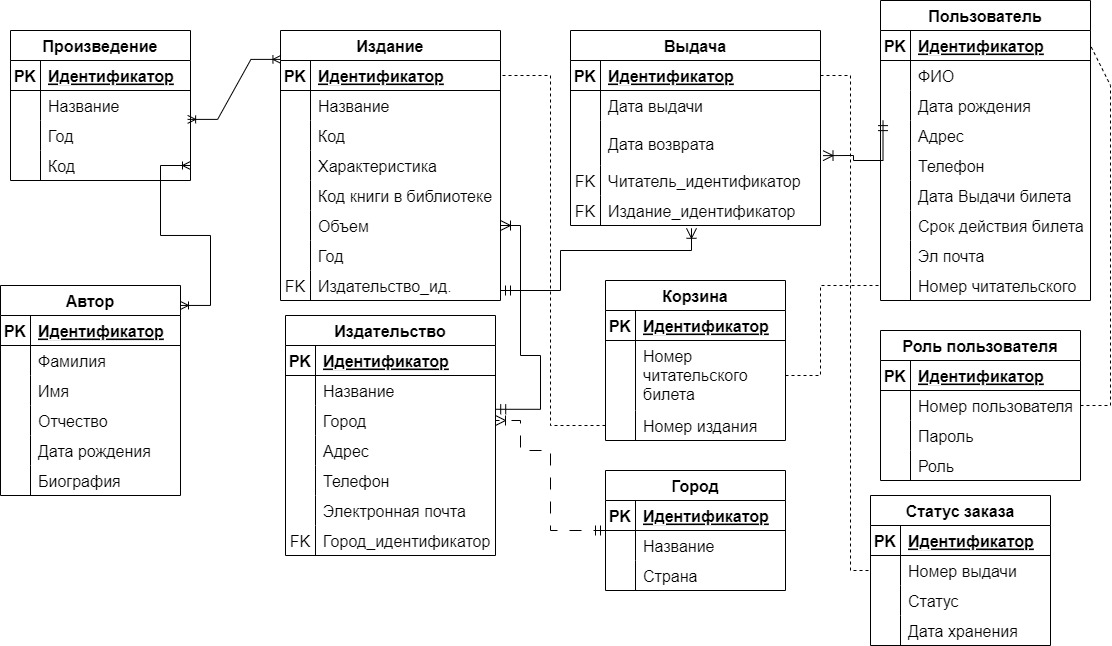


Рисунок 1 – Логическая модель БД

На логической модели отображено 10 основных таблиц – сущностей.

## Описание сущностей БД

На основе логической модели построена физическая модель БД (см. Приложение), на которой кроме имен полей отображен их типы данных. Схема физической модели построена с помощью PostgreSQL.

На физической модели отображено 10 основных таблиц – сущностей и 2 дополнительные таблицы, которые поддерживают связь многие ко многим между основными сущностями.

Сущность автор (author) предназначена для хранения информации об авторе, его имя и биография. Для записи в поля фамилия, имя, отчество используется символьная строка типа varchar длиной 100 символов. Для записи даты рождения используется календарная дата типа date. Для записи поля биографии используется также тип varchar длиной 1000 символов.

Таблица Произведение (composition) предназначена для хранения имени и года написания книги. Для имени произведения используется строковый тип данных длиной 45 символов. Для кода книги в библиотеке и года написания используется целочисленный тип данных.

Таблица Автор/Произведение (composition\_author) содержит поля внешних ключей, которые используют целочисленный тип данных.

Сущность издание (publish), предназначена для хранения информации о разных изданиях одного или нескольких произведений. Поля код издания, год и объем имеют целочисленный тип данных. Название издания используют строковый тип данных длиной 100 символов, характеристика также относится к этому типу данных, только кол-во символов достигает 1200 символов.

Таблица Издание/Произведение (composition\_publish) содержит поля внешних ключей, которые используют целочисленный тип данных.

Таблица издательство (publisher) предназначена для хранения контактной информации о издательстве. Поля название, адрес и эл. почта имеют строковый тип данных длиной 100 символов. Номер телефона имеет такой же тип данных, только кол-во символов 15, в данном случае был выбран такой тип данных, потому что используется номера с кодом «+7».

Сущность город (city) предназначена для хранения информации о городах нахождения издательств. Поля названия и страна относятся к целочисленному типу данных с длиной символов 100.

Таблица выдача (release) хранит информацию о выдаче книг, а именно даты, идентификатор книги и пользователя. Поля дата выдачи и возврата используется календарная дата типа date. Также есть поле статус выдачи книги, который принимает тип символьной строки длиной 20 символов.

Сущность пользователь, предназначена для хранения всех данных человека, авторизованного в системе. Поля дата рождения, выдачи и срока действия читательского билета имеют тип календарной даты. Для хранения имени и адреса проживания используется строковой тип данных длиной 100 символов, а также такой тип используется для хранения номера телефона и роли пользователя длиной 50 символов. Целочисленный тип данных имеет поле номер читательского билета.

Идентификаторы и внешние ключи всех таблиц имеют целочисленный тип данных.

# Программный продукт

Для работы с БД была подключена библиотека Npgsql.dll [7] и для работы с документами формата doc – Microsoft.Office.Interop.Word.dll.

В системе было реализовано 2 роли: читатель и администратор. Функционал вкладок для этих пользователей значительно отличается друг от друга.

На рисунках 2 и 3 отображен функционал вкладок доступных для пользователя с ролью читатель: авторизация, просмотр изданий и читательский билет.

На вкладке «Просмотр изданий» (рис. 2) пользователю доступна возможность выбора произведения для заказа и просмотр информации о нем, для этого необходимо кликнуть по необходимому произведению в таблице.

В выпадающем поле пользователь может выбрать конкретное издание для добавления заказа в корзину.

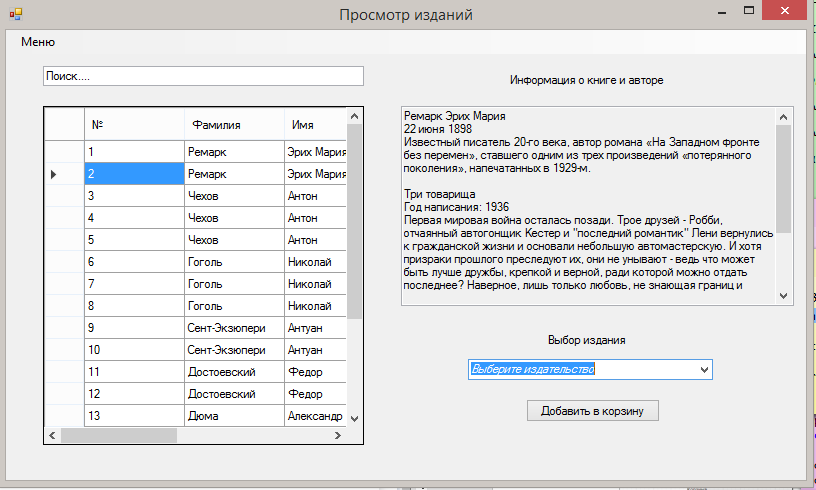


Рисунок 2 – Функционал вкладки «Выбор произведения».

На вкладке «Читательский билет» (рис. 3) реализованы функции: просмотр существующих и предыдущих заказов, просмотр книг, добавленных в корзину. Также возможно удаление выбранной книги из корзины.

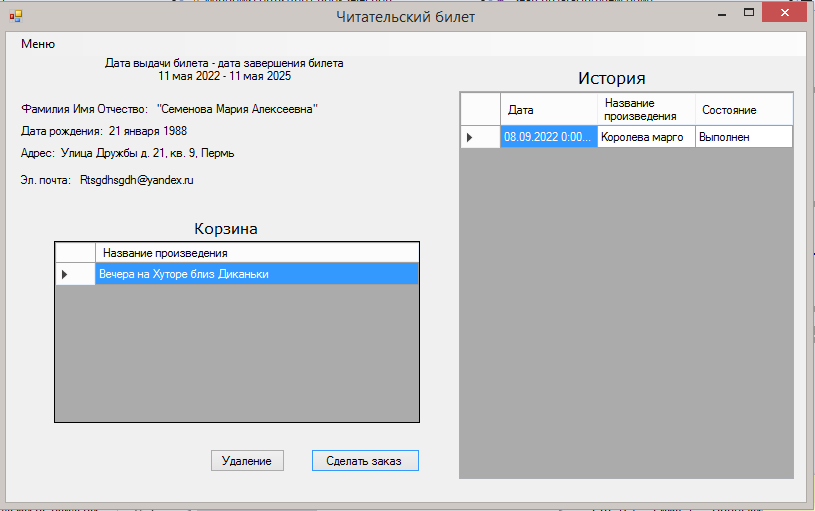


Рисунок 3 – Функционал вкладки «Читательский билет».

На рисунках 4 – 7 отображен функционал вкладок доступных для пользователя с ролью администратор: авторизация, пользователи, управление заказами книг, выбор изданий.

Вкладка «Пользователи» включает в себя таблицу с полями: ФИО, адрес, электронная почта, телефон, дата выдачи билета, дата завершения действия билета, состояние билета на текущий момент (действует/окончен). Внизу расположен блок «режим редактирования», где можно

• вносить изменения в данные существующих пользователей

• добавлять новых пользователей

• удаления данных пользователей из системы.

Также в данной вкладке находятся кнопки по формированию отчетов по действиям пользователей и по кол-ву пользователей, которые реализованы через запрос к БД.

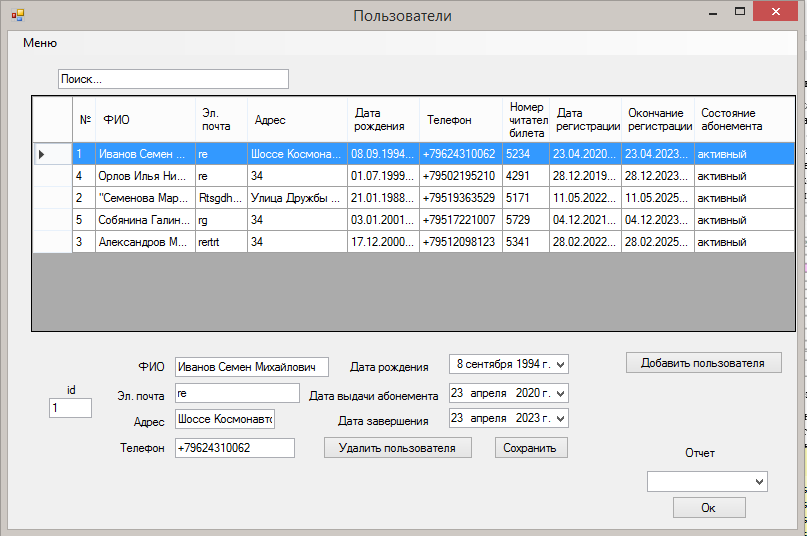


Рисунок 4 – Функционал вкладки «Пользователи».

Вкладка «Управление заказами книг» (рис 5) включает в себя таблицу с полями: ФИО, номер читательского билета, дата выдачи и возврата книги, статус готовности заказа. На данной странице реализованы следующие функции

• Функция поиска по таблице

• Функция изменения статуса заказа и даты возврата

Также в данной вкладке находятся кнопки по формированию отчетов по кол-ву книг у пользователей и о выполненных и отмененных заказах.

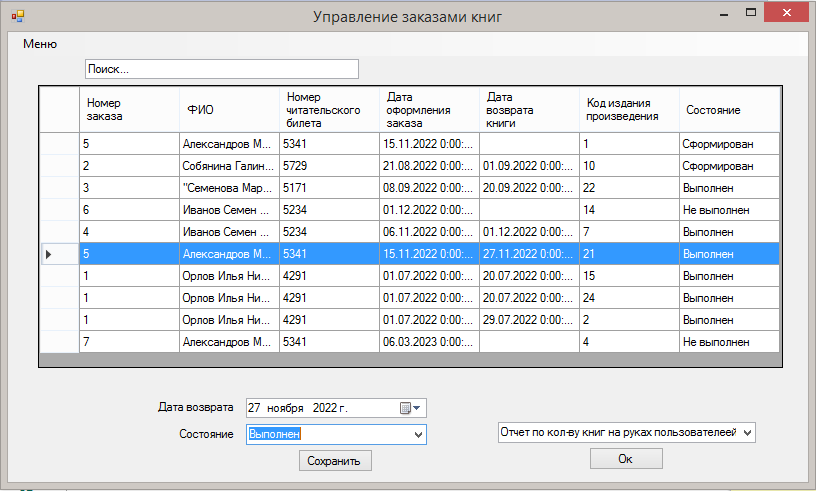


Рисунок 5 – Функционал вкладки «Управление заказами книг».

Функционал вкладки «Выбор изданий» (рис. 6) включает в себя просмотр информации о выбранном произведении и ознакомление с издательствами, доступными в библиотеке. Также просмотр отчета о кол-ве изданий в библиотеке.

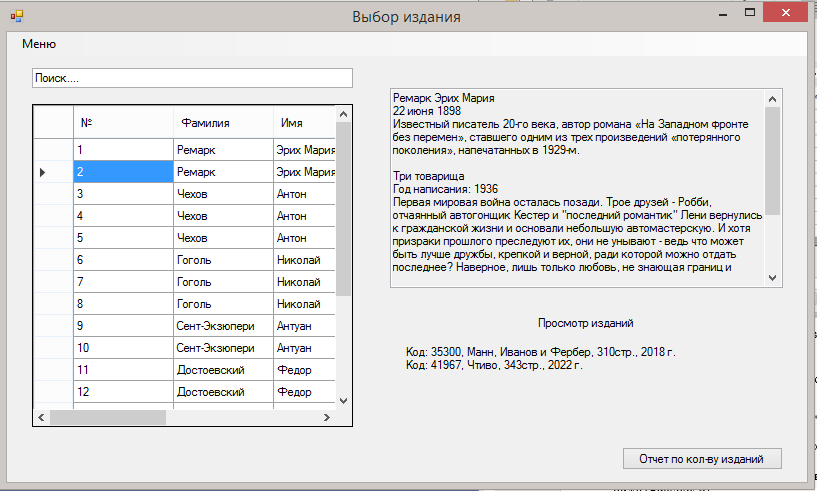


Рисунок 6 – Функционал вкладки «Выбор изданий»

Функционал направленный на редактирование, добавление и удаление информации В БД присутствует на каждой вкладке системы. Например, на рисунке 4 реализованы сразу три взаимодействия с БД: добавление нового читателя в систему, редактирование информации о существующем пользователе, удаление пользователя из системы. Все эти функции реализованы через запросы.

Функционал вкладки «Отчет» отображен ниже (Рисунок 6). На данной вкладки реализован просмотр запроса к БД, с возможностью написать имя, запрашиваемого отчет.

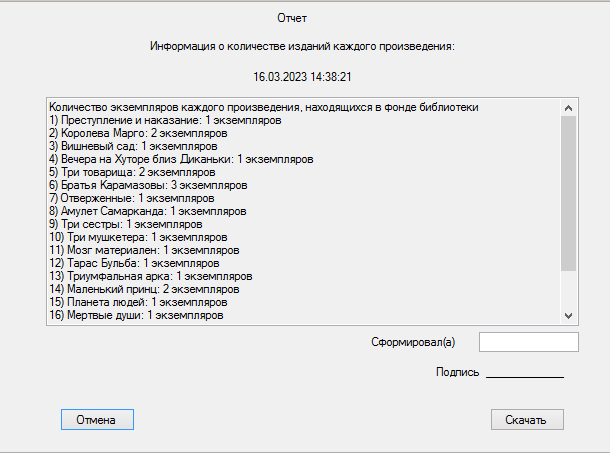


Рисунок 7 – Вкладка «Отчеты»

На рисунке 8 показан вид отчета, выводимого из системы в формат doc.

При ошибочном вводе данных, под полями ввода будет всплывать сообщение. Если ошибка связана с вводом информации в некорректном формате, то текст ошибки: «Некорректный формат введенных данных». Если ошибка связана с несоответствием введенной информации, например, неправильно введен логин или пароль, то текст ошибки: «Неверный логин или пароль» (рис. 9)

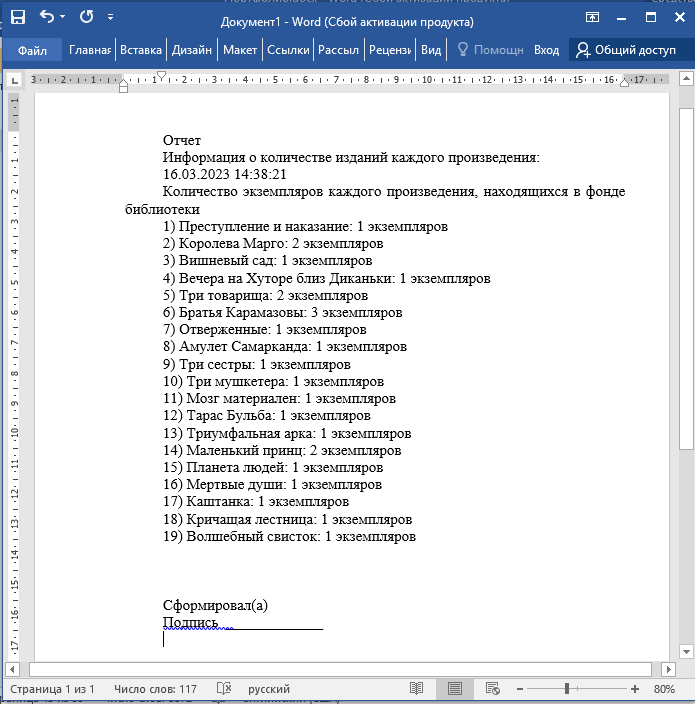


Рисунок 8 – Вид отчета в формате doc

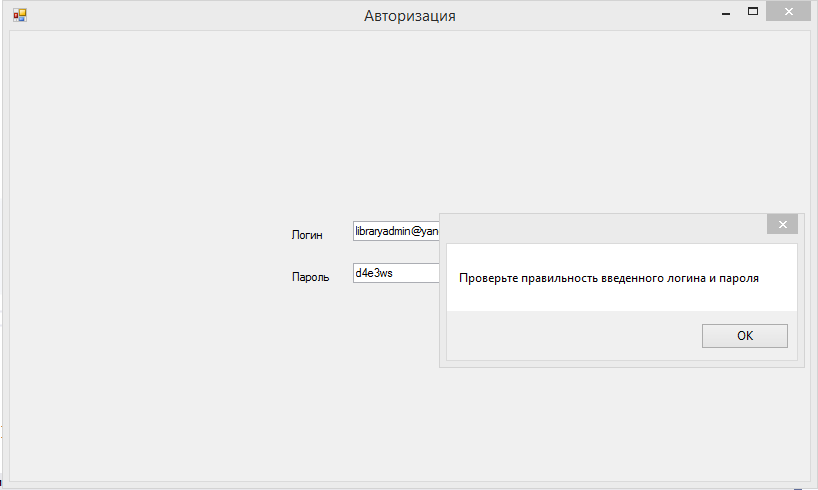


Рисунок 9 – Окно ошибки на введенный неверный пароль

# Выводы

В рамках работы была проанализирована предметная область и написано техническое задание для разработки. Было спроектировано хранилище данных, составлены логическая и физическая модели БД. На основе спроектированного интерфейса, созданной БД и разработанной структуры классов, написан программный код реализации.

Реализовав приложение с использованием программных компонентов, связанных с использованием базы данных, и информации полученной из неё, можно отметить, что цель и задачи, определенные в начале исследования, достигнуты.

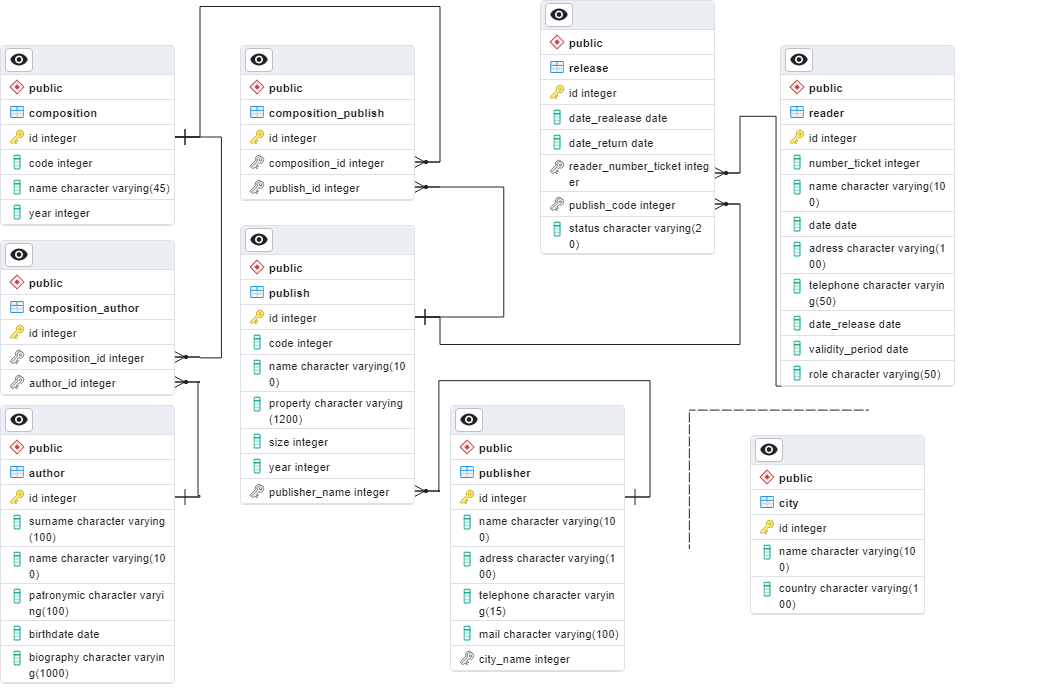
Приложение было протестировано и выявлено одна проблема – это не фиксированная таблица базы данных, чтоб просмотреть полностью по ширине столбец его нужно расширять.

Также следует добавить, что воспроизведение информации в табличном виде более привычна и понятна человеческому глазу, с ее помощью можно донести любые мысли и идеи.

# Список литературы

1. СУБД PostgreSQL: почему её стоит выбрать для работы с данными и как установить / Режим доступа: url: (https:// https://practicum.yandex.ru/blog/chto-takoe-subd-postgresql/) – Дата обращения: (17.03.2023).
2. PostgreSQL / Режим доступа: url: (https:// traditio.wiki/PostgreSQL/) – Дата обращения: (18.03.2023).
3. Что такое PostgreSQL? - Особенности - Преимущества и недостатки / Режим доступа: url: (https://ru.education-wiki.com/5154595-what-is-postgresql/) – Дата обращения: (18.03.2023).
4. А. Т. Сейранян, Программирование в среде.NET// Междисциплинарность науки как фактор инновационного развития. – 2021. – С. 73-76./
5. Программная работа с документами word с помощью библиотеки Microsoft.Office.Interop.Word/ Режим доступа: url: ([https://nationalteam.worldskills.ru/skills/programmnaya-rabota-s-dokumentami-word-s-pomoshchyu-biblioteki-microsoft-office-interop-word/](https://nationalteam.worldskills.ru/skills/programmnaya-rabota-s-dokumentami-word-s-pomoshchyu-biblioteki-microsoft-office-interop-word/%20) ) – Дата обращения: (12.03.2023).
6. C#. ЭЛЕМЕНТ УПРАВЛЕНИЯ DATAGRIDVIEW / Режим доступа: url: (<https://www.bestprog.net/ru/2018/02/17/the-datagridview-control_ru/>) – Дата обращения: (20.05.2022).
7. PostgreSQL / Режим доступа: url: (https:// https:// https://metanit.com/sharp/efcore/7.3.php/) – Дата обращения: (1.03.2023).
8. Выбор клиент-серверной СУБД для реализации информационной системы // Современные информационные технологии – 2016. – С. 11-15/
9. Гвоздева Т.В., Баллод Б.А. Проектирование информационных систем. Серия «Высшее образование». – М.: Феникс, 2009. – 512 С.

# Приложение 1



# Приложение 2

Разделение доступа

private void button1\_Click(object sender, EventArgs e)

{

login=textBox1.Text;

password = textBox2.Text;

OpenDBuser();

if (role == "администратор")

{

OrderManagment om = new OrderManagment();

om.Show();

this.Visible = false; }

if (role == "читатель")

{

BookSelection bs = new BookSelection();

bs.Show();

bs.Transfer\_subscription\_number(Number\_ticket);

this.Visible = false; }}

Добавление, изменение и удаление данных БД

// добавление

public void Add\_user()

{

var id\_us = textBox6.Text;

var FIO = textBox2.Text;

var emaill = textBox3.Text;

var adress = textBox4.Text;

var phone = textBox5.Text;

var date = dateTimePicker1.Text;

var date\_issue = dateTimePicker2.Text;

var date\_completion = dateTimePicker3.Text;

//генерация абонемента

Random rnd = new Random();

var ticket = 0;

while (true)

{

int n = 0;

ticket = rnd.Next(1000, 10000);

foreach (DataGridViewRow row in dataGridView1.Rows)

{

if (row.Cells[0].Value != null) // Need to check for null if new row is exposed

{

int cellId = Convert.ToInt32(row.Cells[6].Value);

if (cellId != ticket)

n++;}}

if (n == dataGridView1.Rows.Count)

break; }

int idd = Convert.ToInt32(id\_us) + 1;

var addQuery = $"insert into reader (id, name, emaill, adress, date, telephone, number\_ticket, date\_release, validity\_period, role ) values ('{idd}', '{FIO}', '{emaill}', '{adress}', '{Convert.ToDateTime(date)}', '{phone}', '{ticket}', '{Convert.ToDateTime(date\_issue)}', '{Convert.ToDateTime(date\_completion)}','читатель')";

var command = new NpgsqlCommand(addQuery, db.getConnection());

command.ExecuteNonQuery();

MessageBox.Show("Запись успешно создана!"); }

// изменение

private void Update\_user()

{

var selectedRowIndex = dataGridView1.CurrentCell.RowIndex;

var id\_us = textBox6.Text;

var FIO = textBox2.Text;

var emaill = textBox3.Text;

var adress = textBox4.Text;

var phone = textBox5.Text;

var date = dateTimePicker1.Text;

var date\_issue = dateTimePicker2.Text;

var date\_completion = dateTimePicker3.Text;

if (dataGridView1.Rows[selectedRowIndex].Cells[0].Value.ToString() != string.Empty)

{

dataGridView1.Rows[selectedRowIndex].SetValues(id\_us, FIO, emaill, adress, phone, date, date\_issue, date\_completion);

int index = dataGridView1.CurrentCell.RowIndex;

id\_us = dataGridView1.Rows[index].Cells[0].Value.ToString();

FIO = dataGridView1.Rows[index].Cells[1].Value.ToString();

emaill = dataGridView1.Rows[index].Cells[2].Value.ToString();

adress = dataGridView1.Rows[index].Cells[3].Value.ToString();

phone = dataGridView1.Rows[index].Cells[4].Value.ToString();

date = dataGridView1.Rows[index].Cells[5].Value.ToString();

date\_issue = dataGridView1.Rows[index].Cells[6].Value.ToString();

date\_completion= dataGridView1.Rows[index].Cells[7].Value.ToString();

String changeQuery = $"update reader set name='{FIO}', emaill='{emaill}', date='{Convert.ToDateTime(date)}', adress='{adress}', telephone='{phone}', date\_release='{Convert.ToDateTime(date\_issue)}', validity\_period='{Convert.ToDateTime(date\_completion)}' where id='{id\_us}'";

var command = new NpgsqlCommand(changeQuery, db.getConnection());

command.ExecuteNonQuery();

MessageBox.Show("Запись успешно изменена!"); }}

//удаление

private void Delate()

{

int index = dataGridView1.CurrentCell.RowIndex;

db.openConnection();

var id = Convert.ToInt32(dataGridView1.Rows[index].Cells[0].Value);

var delateQuery = $"delete from reader where id='{id}'";

var command = new NpgsqlCommand(delateQuery, db.getConnection());

command.ExecuteNonQuery();

MessageBox.Show("Запись успешно удалена!");

dataGridView1.Rows[index].Visible = false; }

Формирование отчетов

exportWord.Application wordapp = new exportWord.Application();

wordapp.Visible = true;

exportWord.Document worddoc;

object wordobj = System.Reflection.Missing.Value;

worddoc = wordapp.Documents.Add(ref wordobj);

wordapp.Selection.TypeText(label1.Text+Environment.NewLine);

wordapp.Selection.TypeText(label2.Text + Environment.NewLine);

wordapp.Selection.TypeText(label3.Text + Environment.NewLine);

wordapp.Selection.TypeText(textBox2.Text + Environment.NewLine);

wordapp.Selection.TypeText("" + Environment.NewLine);

wordapp.Selection.TypeText("" + Environment.NewLine);

wordapp.Selection.TypeText(label5.Text+textBox1.Text + Environment.NewLine);

wordapp.Selection.TypeText(label6.Text + Environment.NewLine);

wordapp = null;

Обработка исключительных ситуаций

String role = "";

int Number\_ticket = 0;

String login = "";

String password = "";

private void OpenDBuser()

{

string text\_request\_author = $"select r.number\_ticket, r.emaill, ru.password\_user, ru.role\_user from reader r JOIN role\_user ru ON r.id = ru.id\_user";

NpgsqlCommand com = new NpgsqlCommand(text\_request\_author, db.getConnection());

db.openConnection();

NpgsqlDataReader read = com.ExecuteReader();

read.Read();

while (read.Read())

{

if (login.Equals(read.GetString(1)))

{

String Pas = "";

if (!read.IsDBNull(2))

Pas = read.GetString(2);

if (password.Equals(Pas))

{

role = read.GetString(3);

Number\_ticket = read.GetInt32(0);

else

MessageBox.Show("Проверьте правильность введенного логина и пароля");}}

read.Close();