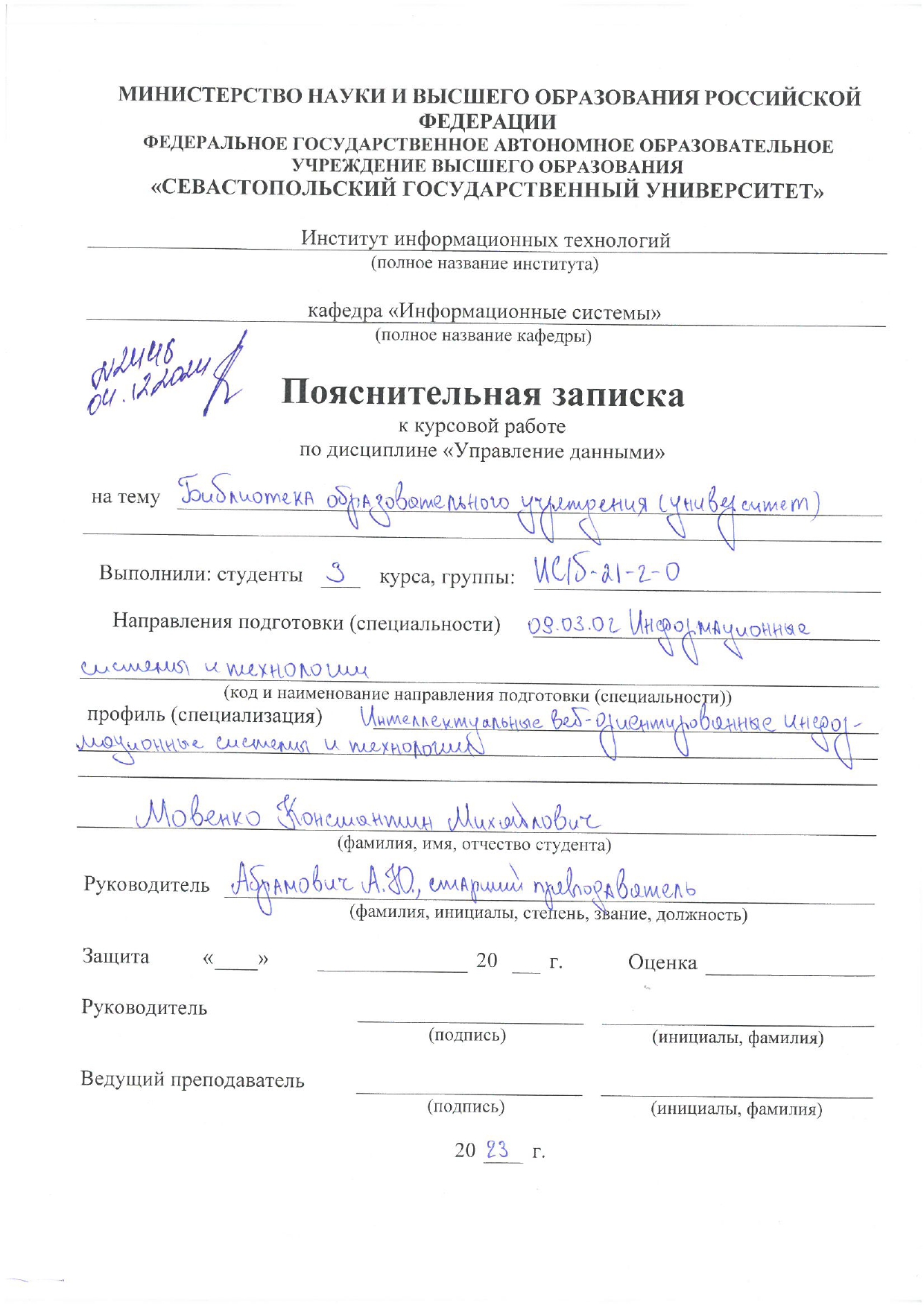
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | | | | | | | | | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | | | | | | | | | |
|  |  | | | | | | | | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | | | | | | | | | |
|  | | | | |  | | |  | | | | | | | | |  | | | | | | | |
|  | | | | | | | | | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | |  | | | | | | | |
|  | | | | | | | | | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | | | | | | | | | |
|  | | | | | | | | | |  | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | | | | | | | | | |
|  | | |  | | | | | | | | | | | | | | | | | | | | | |
|  | | |  | | | | | | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | | | | | | | | | |
|  | |  | |  | |  | | | | | | | |  | |  | |  |  | | | |  | |
|  | | | | | | |  | | | |  |  | | | | | | |  | | |  | | |
|  | | | | | | | | |  | | | | | | | | | | |  |  | | | |
|  | | | | | | | |  | | | | | | | | | | | |  |  | | | |
|  | | | | | | |  | | | |  |  | | | | | | |  | | |  | | |
|  | | | | | | | | |  | | | | | | | | | | |  |  | | | |
|  | | | | | | | |  | | | | | | | | | | | |  |  | | | |
|  | | | | | | | | | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | |  | |  | | | | | | | | |

****

СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 3](#_Toc158305178)

[1 АНАЛИТИЧЕСКАЯ ЧАСТЬ 4](#_Toc158305179)

[1.1 Анализ предметной области (концептуальное моделирование) 4](#_Toc158305180)

[1.2 Постановка задачи 4](#_Toc158305181)

[2 РАЗРАБОТКА ЛОГИЧЕСКОЙ МОДЕЛИ БАЗЫ ДАННЫХ 6](#_Toc158305182)

[2.1 Построение диаграммы «сущность-связь» в нотации П. Чена 6](#_Toc158305183)

[2.2 Построение модели, основанной на ключах, и полной атрибутивной модели в нотации IDEF1X 7](#_Toc158305184)

[3 РАЗРАБОТКА ФИЗИЧЕСКОЙ МОДЕЛИ БАЗЫ ДАННЫХ 9](#_Toc158305185)

[3.1 Выбор аппаратной и программной платформы и реализация БД 9](#_Toc158305186)

[3.2 Тестирование базы данных (создание и реализация запросов) 9](#_Toc158305187)

[3.3 Разграничение прав доступа 18](#_Toc158305188)

[4. РАЗРАБОТКА КЛИЕНТСКОГО ПРИЛОЖЕНИЯ 19](#_Toc158305189)

[4.1 Обоснование выбора языка программирования 19](#_Toc158305190)

[4.2 Разработка интерфейса пользователя 19](#_Toc158305191)

[4.3 Алгоритм работы каждого из модулей 19](#_Toc158305192)

[4.4 Тестирование работы приложения 19](#_Toc158305193)

[ЗАКЛЮЧЕНИЕ 20](#_Toc158305194)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 21](#_Toc158305195)

[ПРИЛОЖЕНИЕ А 22](#_Toc158305196)

[ПРИЛОЖЕНИЕ Б 25](#_Toc158305197)

# ВВЕДЕНИЕ

В настоящее время информационные системы являются неотъемлемой частью управления деятельностью множества организаций и учреждений. Создание эффективной базы данных позволяет автоматизировать процессы управления, повысить эффективность работы и сократить время на выполнение рутинных задач. В рамках данной работы была поставлена задача разработки базы данных для библиотеки образовательного учреждения.

Целью работы является проектирование и реализация базы данных, а также разработка пользовательского приложения для её использования. В базе данных должны храниться данные о хранимых книгах, читательских билетах и истории займа и возврата книг. Благодаря этому библиотека сможет эффективнее вести учёт имущества и своих клиентов.

В ходе работы будут рассмотрены логическая и физическая модели базы данных, а также способы реализации связей между таблицами. Будет создано приложение для управления базой данных и предоставления доступа к её функционалу. Результатом работы будет готовая база данных и работающее приложение, которые можно использовать для ведения учёта в библиотеке образовательного учреждения.

# 1 АНАЛИТИЧЕСКАЯ ЧАСТЬ

## Анализ предметной области (концептуальное моделирование)

Студенческая библиотека – это один из видов библиотек, ориентированный на студентов высших учебных заведений. В стандартной библиотеке читатель может бесплатно взять на временное пользование книгу из специального архива, после чего он должен вернуть её обратно в библиотеку. Если читатель не возвращает книгу вовремя, на него накладываются санкции, в том числе материального характера.

Особенностью студенческой библиотеки является то, что её основные клиент – студент, который нуждается не столько в художественной, сколько в научной литературе.

Для разработки базы данных и приложения библиотеки были изучены документы, относящиеся к процессу обучения и организации работы библиотечных учреждений. Кроме того, было проведено общение со специалистами данной предметной области, в результате которого были получены ценные сведения о особенностях функционирования библиотек.

## Постановка задачи

Были определены основные объекты предметной области:

* Книги (название, автор, год, издатель, тематика и др.);
* Люди (ФИО, статус в университете, контакты и др.);
* Выдачи (читатель, книга, дата выдачи, дата возврата и др.);

Связи между объектами:

* Каждая запись о выдаче книги связывает соответствующую книгу с читателем;
* Каждой книге соответствует 1 или более авторов;

Были определены группы пользователей и их набор используемых ими операций:

1. Библиотекарь (администратор):

* Операции: учёт книг в базе данных, регистрация новых пользователей, оформление выдачи и возврата книг;
* Выводимые данные: вся информация о книгах, список читателей и должников, история выдачи и возврата книг;

1. Читатель:

* Операции: поиск книг по различным критериям, запрос на выдачу книги или продление срока возврата;
* Выводимые данные: информация о доступных книгах, информация о выданных читателю книгах, сроках возврата и задолженностях;

Сценарий пользователя: студента регистрирует библиотекарь в системе, после чего он может войти в свой аккаунт и увидеть все доступные в библиотеке книги. После этого он может очно подойти к ответственному за выдачу книг сотруднику библиотеки, который может назначить выдачу необходимой книги на определенный срок, который библиотекарь может скорректировать исходя из обстоятельств. Если за студентом есть долги в виде вовремя не сданных книг, то он не может взять новую. После успешно оформленной выдачи читатель может в любое время до её окончания сдать книгу обратно. Если он не сдает книгу вовремя, то на него накладываются санкции – он не сможет получить новую книгу, а также может получить предупреждение от деканата. На каждую книгу формируется своя выдача, после окончания (закрытия) выдачи, она остается в базе данных, что позволяет отслеживать историю как книг, так и читателей.

# 2 РАЗРАБОТКА ЛОГИЧЕСКОЙ МОДЕЛИ БАЗЫ ДАННЫХ

## 2.1 Построение диаграммы «сущность-связь» в нотации П. Чена

После проведения анализа предметной области и выделения ключевых сущностей стало возможным создание диаграммы «сущность-связь» в нотации П. Чена.

Перед этим были определены отношения между выделенными сущностями. Была разработана сложная сетевая структура, основанная на основных элементах предметной области (Рисунок 2.1).

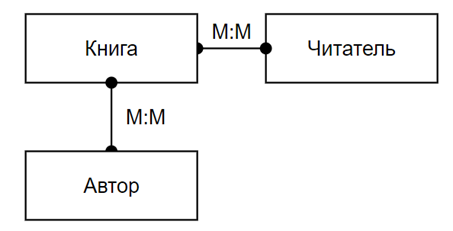


Рисунок 2.1 – Сложная сетевая структура

На основе неё была разработана упрощенная сетевая структура. Она была получена путём введения промежуточных таблиц для преобразования связи «многие ко многим» в связь «один ко многим» (Рисунок 2.2).

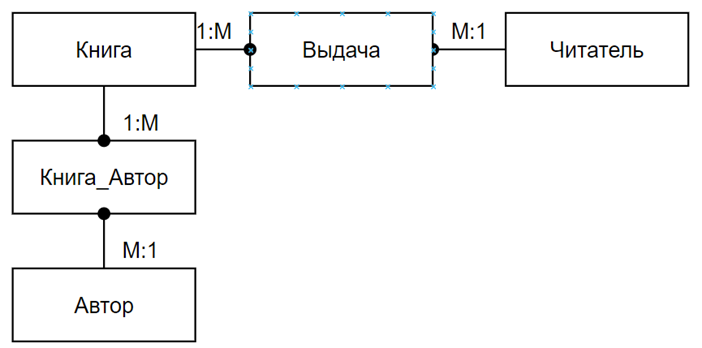


Рисунок 2.2 – Простая сетевая структура

На основе предыдущих таблиц была создана ER-диаграмма (Рисунок 2.3), которая иллюстрирует взаимосвязь между объектами и основные элементы их взаимодействия.

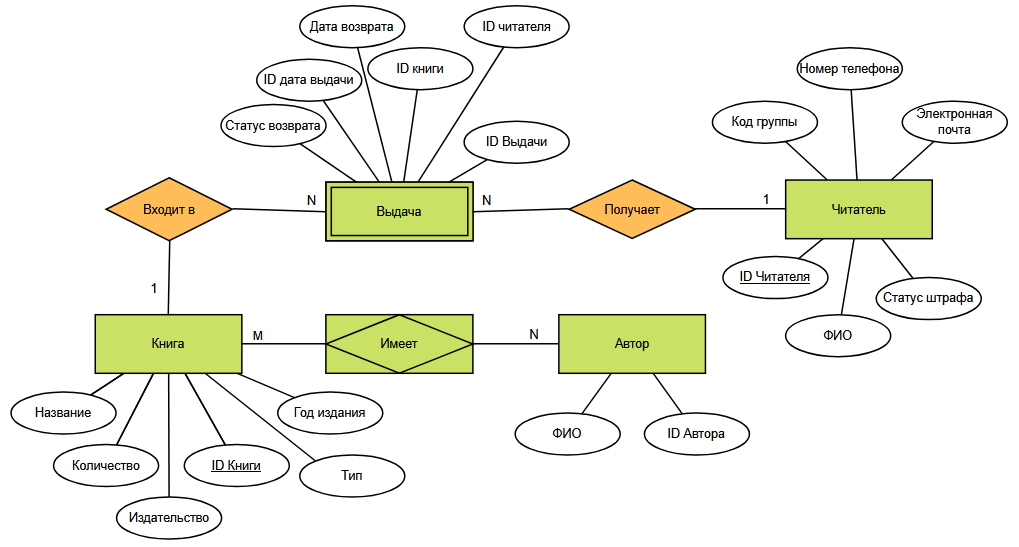


Рисунок 2.3 – ER**-**диаграмма в нотации П. Чена

## 2.2 Построение модели, основанной на ключах, и полной атрибутивной модели в нотации IDEF1X

Основной целью модели, основанной на ключах, является широкий обзор структур данных и ключей, нужных для поддержки определенной предметной области.

В модель были включены все упомянутые сущности, их первичные и вторичные ключи, а также атрибуты (Рисунок 2.4). Для достижения полной атрибутивной модели структура базы данных была приведена к третьей нормальной, была устранена избыточность данных.

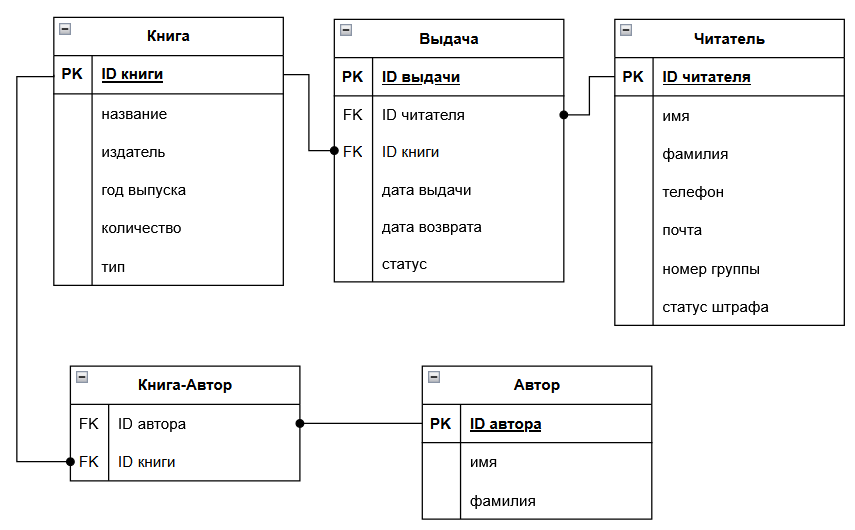


Рисунок 2.4 – Полная атрибутивная модель в нотации IDF1X

Т.к. для первой нормальной формы достаточно, чтобы все атрибуты были атомарными, имеющаяся структура удовлетворяет ей. Всё связи «многие ко многим» выполнены через ассоциативные сущности.

Был проведён анализ на соответствие второй нормальной форме. Первичные ключи в каждой сущности полностью определяют каждый не ключевой атрибут. Таким образом, база данных соответствует второй нормальной форме.

Продолжая анализ в соответствии с третьей нормальной формой, подтверждено отсутствие транзитивных зависимостей в каждой сущности. Это говорит о том, что база данных находится в третьей нормальной форме.

# 3 РАЗРАБОТКА ФИЗИЧЕСКОЙ МОДЕЛИ БАЗЫ ДАННЫХ

## 3.1 Выбор аппаратной и программной платформы и реализация БД

Для реализации базы данных библиотеки была выбрана СУБД PostgreSQL. Данный выбор сделан в результате сравнительного анализа с аналогичными СУБД: Firebird и MySQL.

PostgreSQL была выбрана, так как имеет ряд преимуществ по сравнению с другими СУБД, в частности, более широкие возможности по работе с большими объёмами данных, высокую производительность и продвинутый механизм обработки транзакций. Также был учтён положительный опыт работы в графическом клиенте pgAdmin, облегчающем администрирование и разработку баз данных PostgreSQL.

Для реализации физической схемы базы данных были использованы стандартные типы данных, поддерживаемые PostgreSQL: INTEGER, VARCHAR, DATE и др. Тип данных для каждого атрибута был выбран на основе его предназначения и возможностей, предоставляемых СУБД.

SQL запросы, использованные при создании физической модели базы данных, приведены в приложении А.

## 3.2 Тестирование базы данных (создание и реализация запросов)

В соответствии с предметной областью и структурой разработанной базы данных были составлены тестовые наборы данных (Рисунки 3.1-3.3).

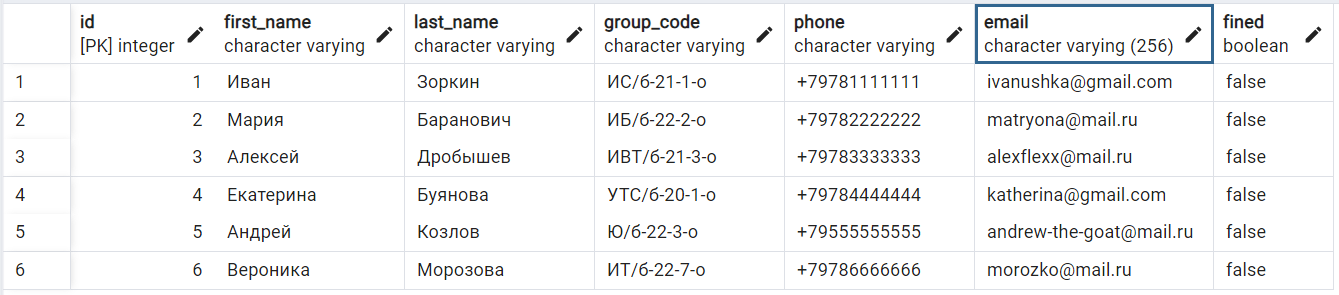


Рисунок 3.1 – Тестовые данные для таблицы «Читатели»

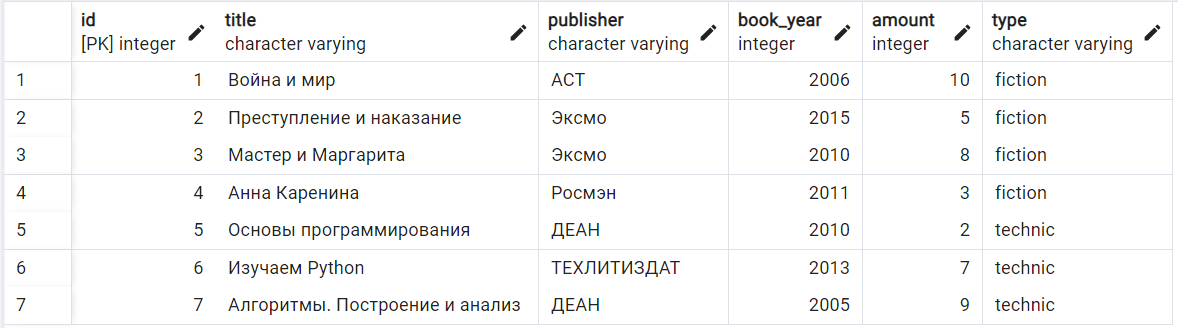


Рисунок 3.2 – Тестовые данные для таблицы «Книги»

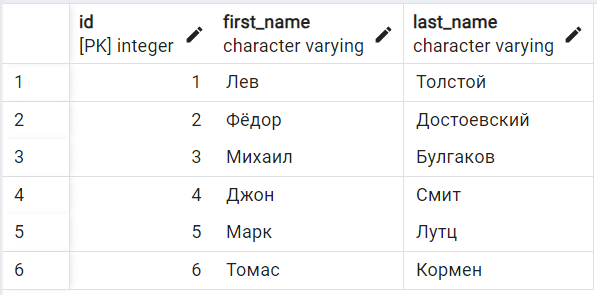


Рисунок 3.3 – Тестовые данные для таблицы «Авторы»

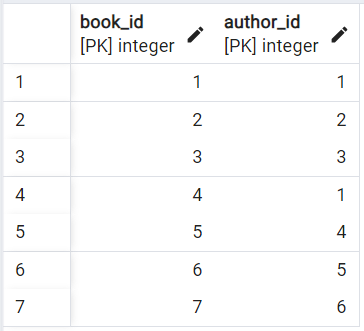


Рисунок 3.4 – Тестовые данные для таблицы «Книга-Автор»

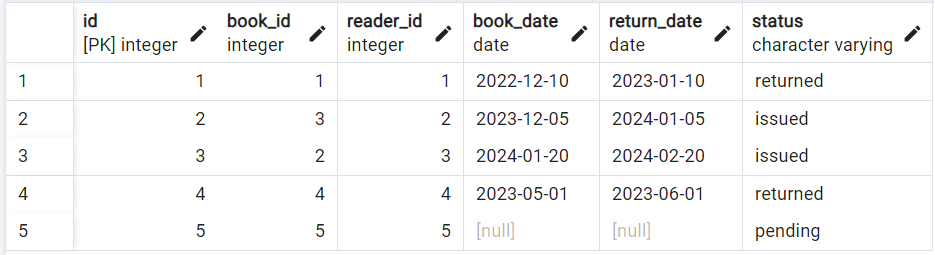


Рисунок 3.5 – Тестовые данные для таблицы «Выдача»

Полученная база данных была протестирована путём проведения 15-ти запросов.

Запрос 1: вывести имена и группы всех читателей библиотеки (текст запроса – Листинг 3.1, результат – Рисунок 3.6).

Листинг 3.1 – Текст запроса 1

select first\_name, last\_name, group\_code

from readers;

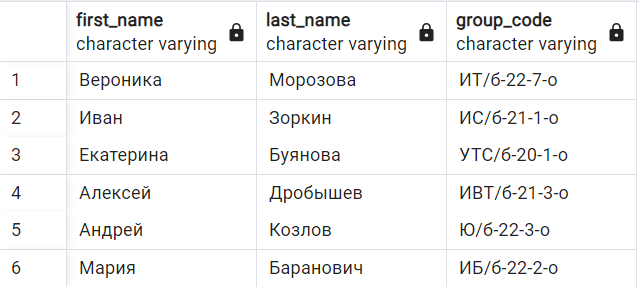


Рисунок 3.6 – Результат запроса 1

Запрос 2: вывести все книги, тип которых – художественная литература (текст запроса – Листинг 3.2, результат – Рисунок 3.7).

Листинг 3.2 – Текст запроса 2

select title, publisher, amount

from books

where type = 'fiction';

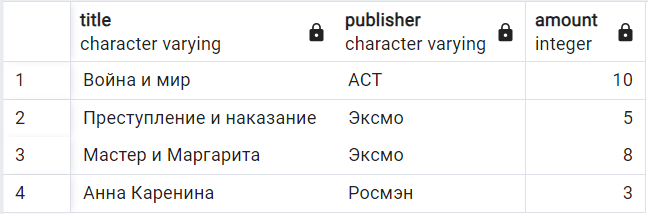


Рисунок 3.7 – Результат запроса 2

Запрос 3: подсчитать число книг, выпущенных до 2010 года (текст запроса – Листинг 3.3, результат – Рисунок 3.8).

Листинг 3.3 – Текст запроса 3

select count(\*) from books

where book\_year < 2010;

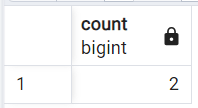


Рисунок 3.8 – Результат запроса 3

Запрос 4: заменить имя четвёртого автора в таблице «Авторы» на ‘Игорь Черпаков’ (текст запроса – Листинг 3.4, результат – Рисунок 3.9).

Листинг 3.4 – Текст запроса 4

update authors

set first\_name = 'Игорь',

last\_name = 'Черпаков'

where id = 4;



Рисунок 3.9 – Результат запроса 4

Запрос 5: вывести все записи, срок возврата которых истёк к моменту запроса (текст запроса – Листинг 3.5, результат – Рисунок 3.10).

Листинг 3.5 – Текст запроса 5

select \* from issuance

where return\_date < current\_date

and status != 'returned';



Рисунок 3.10 – Результат запроса 4

Запрос 6: установить штраф всем читателям, которые на текущий момент не вернули книгу в положенный срок (текст запроса – Листинг 3.6, результат – Рисунок 3.11).

Листинг 3.6 – Текст запроса 4

update readers

set fined = True

where id in

(select reader\_id

from issuance

where return\_date < current\_date

and status != 'returned');

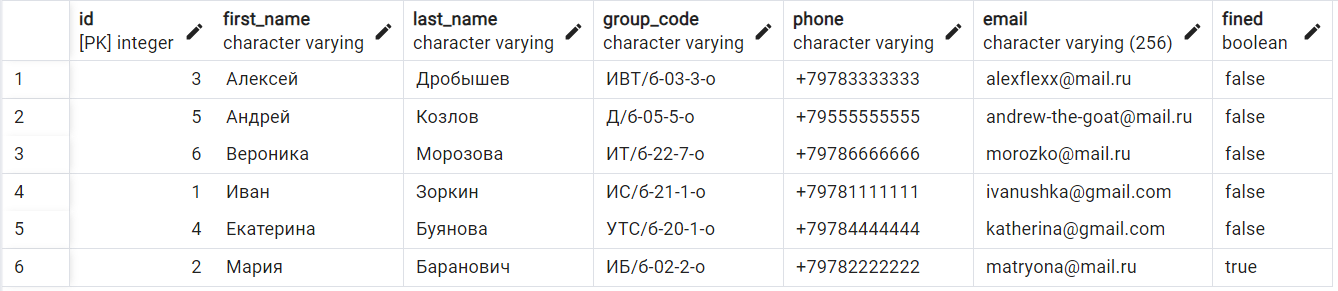


Рисунок 3.11 – Результат запроса 6

Запрос 7: вывести для каждой книги список её авторов (текст запроса – Листинг 3.7, результат – Рисунок 3.12).

Листинг 3.4 – Текст запроса 7

select books.title, string\_agg(authors.last\_name, ', ') as authors

from books

join book\_author on books.id = book\_author.book\_id

join authors on book\_author.author\_id = authors.id

group by books.title;

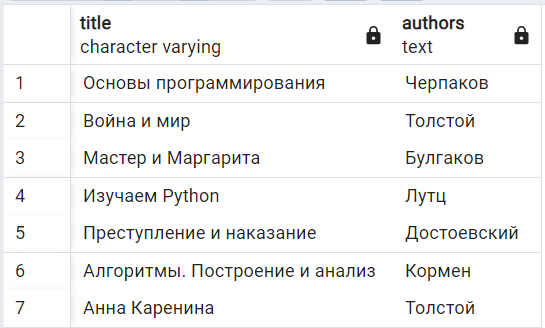


Рисунок 3.12 – Результат запроса 7

Запрос 8: рассмотреть имеющуюся заявку на выдачу – установить дату выдачи, дату возврата и обновить статус (текст запроса – Листинг 3.8, результат – Рисунок 3.13).

Листинг 3.4 – Текст запроса 8

update issuance

set book\_date = current\_date,

return\_date = current\_date + interval '1 month',

status = 'issued'

where id = 5;



Рисунок 3.13 – Результат запроса 8

Запрос 9: добавить в таблицу «Выдачи» заявку на выдачу читателю книги (текст запроса – Листинг 3.9, результат – Рисунок 3.14).

Листинг 3.4 – Текст запроса 9

insert into issuance (book\_id, reader\_id) values (5, 1);

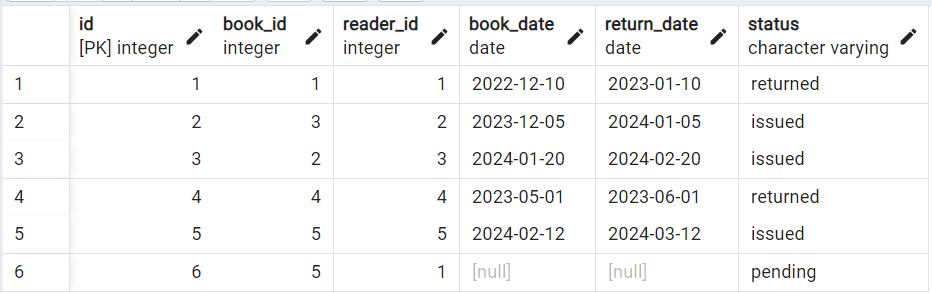


Рисунок 3.14 – Результат запроса 9

Запрос 10: вывести количество выданных книг типа «Художественная литература» (текст запроса – Листинг 3.10, результат – Рисунок 3.15).

Листинг 3.4 – Текст запроса 10

select count(\*)

from issuance

where status = 'issued'

and exists

(select \*

from books

where books.id = issuance.book\_id

and type = 'fiction');

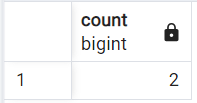


Рисунок 3.15 – Результат запроса 10

Запрос 11: вывести количество книг автора с ID=4, выданных на данный момент (текст запроса – Листинг 3.11, результат – Рисунок 3.16).

Листинг 3.4 – Текст запроса 11

select count(\*)

from issuance

where status = 'issued'

and exists

(select \*

from books

join book\_author as BA

on books.id = BA.book\_id

where BA.author\_id = 4

and books.id = issuance.book\_id

);

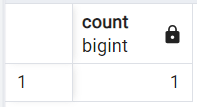


Рисунок 3.16 – Результат запроса 11

Запрос 12: вывести список книг с их общим количеством и количеством книг, доступных для выдачи на данный момент (текст запроса – Листинг 3.12, результат – Рисунок 3.17).

Листинг 3.4 – Текст запроса 12

select books.id, title, amount,

amount - coalesce(issued.total\_count, 0) as available

from books

left join (

select book\_id, count(\*) as total\_count

from issuance

where status = 'issued'

group by book\_id

) as issued

on books.id = issued.book\_id;

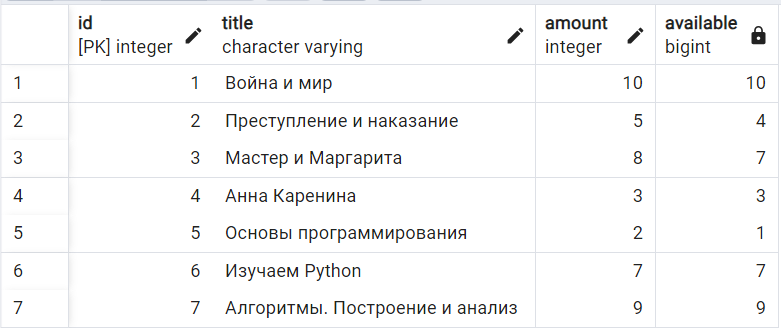


Рисунок 3.17 – Результат запроса 12

Запрос 13: вывести все книги, доступные для выдачи и никем не забронированные (текст запроса – Листинг 3.13, результат – Рисунок 3.18).

Листинг 3.13 – Текст запроса 13

select id, title, book\_year, publisher, type

from books

left join (

select book\_id, count(\*) as total\_count

from issuance

where status in ('issued', 'pending')

group by book\_id

) as issued

on books.id = issued.book\_id

where books.amount - coalesce(issued.total\_count, 0) > 0;



Рисунок 3.18 – Результат запроса 13

Запрос 14: вывести все невозвращённые книги вместе с именами их читателей (текст запроса – Листинг 3.14, результат – Рисунок 3.19).

Листинг 3.14 – Текст запроса 14

select first\_name, last\_name, group\_code,

book\_id, book\_date, return\_date

from issuance

join readers

on readers.id = issuance.reader\_id

where status = 'issued';

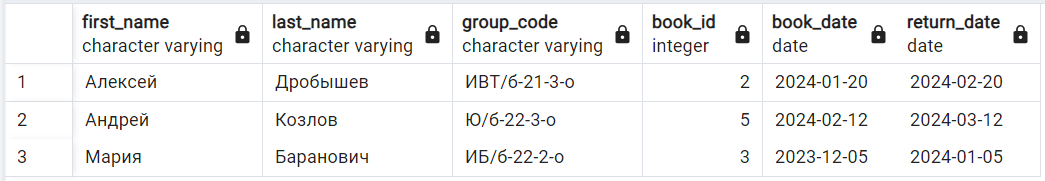


Рисунок 3.19 – Результат запроса 14

Запрос 15: вывести топ авторов по упоминанию их книг в таблице «Выдачи» (текст запроса – Листинг 3.4, результат – Рисунок 3.20).

Листинг 3.15 – Текст запроса 15

select first\_name, last\_name, count(issuance.id) as amount

from authors

join book\_author as ba on ba.author\_id = authors.id

join books on books.id = ba.book\_id

left join issuance on books.id = issuance.book\_id

group by authors.id

order by amount desc;

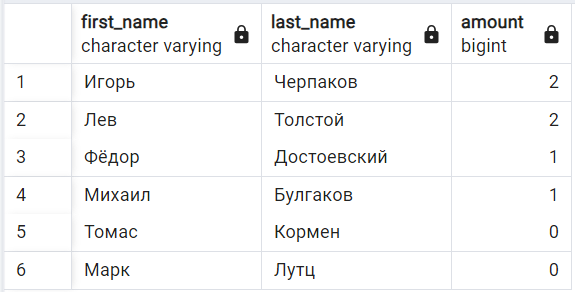


Рисунок 3.20 – Результат запроса 15

По результатам проведённых тестов можно сделать вывод, что таблицы в базе данных созданы правильно и работают корректно.

## 3.3 Разграничение прав доступа

В ходе анализа предметной роли были выделены 2 основные группы пользователей: Сотрудник и Читатель. Сотрудник может просматривать любые данные о книгах и пользователях, добавлять новые данные и обновлять старые. Для него была создана роль STAFF. Читатель может просматривать каталог книг и авторов, делать заявку на выдачу книги, но не имеет доступа к данным других читателей. Для него была создана роль READER.

Листинг 3.16 – Права доступа роли «Сотрудник»

CREATE ROLE staff;

GRANT SELECT ON ALL TABLES IN SCHEMA public TO staff;

GRANT INSERT ON TABLE readers TO staff;

GRANT INSERT ON TABLE books TO staff;

GRANT INSERT ON TABLE authors TO staff;

GRANT INSERT ON TABLE book\_author TO staff;

GRANT UPDATE ON TABLE issuance TO staff;

GRANT UPDATE ON TABLE books TO staff;

GRANT UPDATE ON TABLE authors TO staff;

GRANT DELETE ON TABLE book\_author TO staff;

Листинг 3.17 – Права доступа роли «Читатель»

CREATE ROLE reader;

GRANT SELECT ON TABLE books, authors, book\_author TO reader;

GRANT INSERT ON TABLE issuance TO reader;

# 4. РАЗРАБОТКА КЛИЕНТСКОГО ПРИЛОЖЕНИЯ

## 4.1 Обоснование выбора языка программирования

Для разработки клиентского приложения была выбрана платформа Windows Forms фреймворка .NET. Она предоставляет удобные инструменты для создания интерфейсов клиентских приложений рабочего стола ОС Windows.

Несмотря на то, что платформа не кроссплатформенная, она обладает рядом важных преимуществ: широкий набор компонентов и библиотек, удобный графический интерфейс, большое количество обучающих материалов.

Windows Forms предусматривает возможность написания ПО на различных языках, однако основным для платформы является C#. В отличие от C++, он более высокоуровневый, простой в использовании и сильнее интегрирован с WinForms. Поэтому для разработки приложения был выбран именно он.

## 4.2 Подсистема пользователя

Подсистема пользователя представляет собой основной интерфейс программы, доступный всем её пользователям – читателям, зарегистрированным в системе администраторами. Разработанный интерфейс пользователя включает следующие элементы:

* окно авторизации в системе;
* окно профиля пользователя с функцией смены логина, почты, номера телефона и пароля;
* окно библиотеки со списком доступных для выдачи книг и возможностью подать заявку на выдачу;
* окно формуляра с историей выдач авторизованного пользователя;

Разработанный интерфейс был организован и реализован с учётом требований предметной области и удобства использования.

## 4.3 Подсистема администратора

В разработанной системе администратору доступен весь функционал пользователя, поэтому интерфейс администратора представляет собой надстройку над интерфейсом пользователя. Помимо вышеуказанных окон для администратора были реализованы:

* окно со списком всех зарегистрированных пользователей библиотеки;
* окно добавления нового пользователя библиотеки;
* окно просмотра имеющихся в библиотеке книг и информации о них;
* окно редактирования информации о книге;
* окно добавления новой книги;
* окно со списком заявок и полной историей выдач с возможностью изменять их статус (принимать и отклонять заявки, оформлять возвраты и фиксировать штрафы);

## 4.4 Тестирование системы

Для тестирования системы таблицы базы данных были заполнены начальными значениями (Рисунки 3.1-3.5).

На рисунке 4.1 изображено окно авторизации в системе библиотеки. Права доступа зависят не от способа входа, а от указанной в базе данных роли пользователя. Окно регистрации отсутствует, так как всех пользователей регистрируют администраторы.

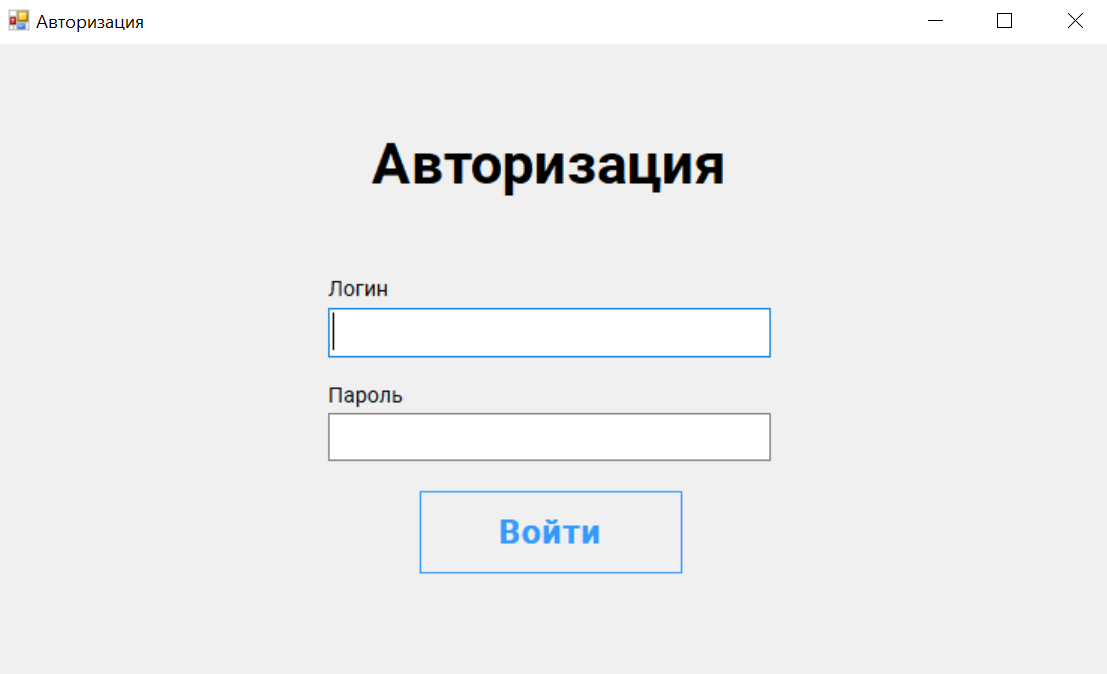


Рисунок 4.1 – Окно авторизации

Вне зависимости от роли пользователю открывается личный профиль с информацией о нём и его аккаунте (Рисунок 4.2). При изменении соответствующих полей становятся доступны функции смены пароля и редактирования данных.

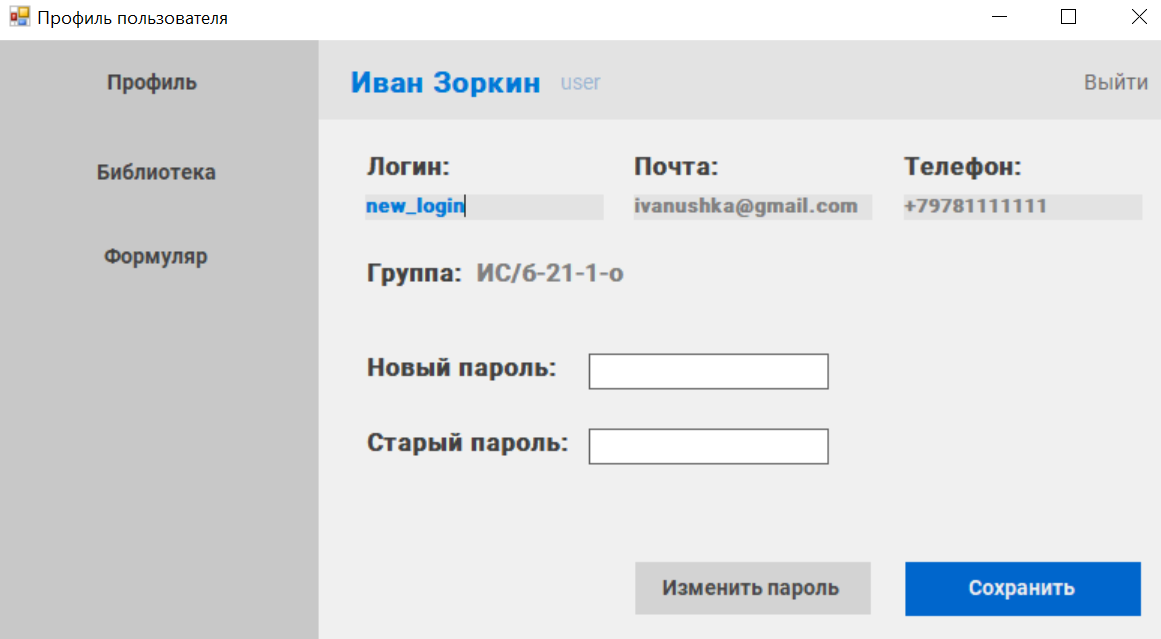


Рисунок 4.2 – Окно профиля (изменение логина)

При редактировании данных подтверждение пароля не требуется, а соответствующие поля отмечаются цветом. При смене пароля происходит сверка введённого пароля с паролём от аккаунта пользователя.

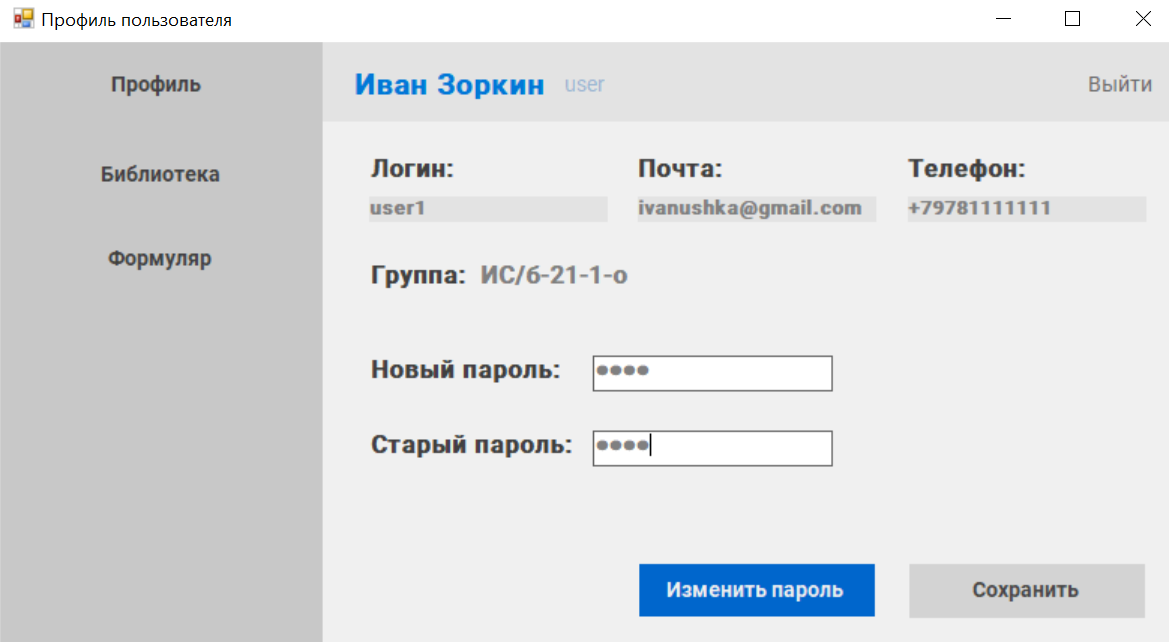


Рисунок 4.3 – Окно профиля (изменение пароля)

Далее следует окно библиотеки (Рисунок 4.4). В нём расположена таблица, содержащая информацию о всех книгах, имеющихся в библиотеке и доступных для выдачи. Через него можно подать заявку на выдачу – пусть это будут «Основы программирования» и «Введение в искусственный интеллект».

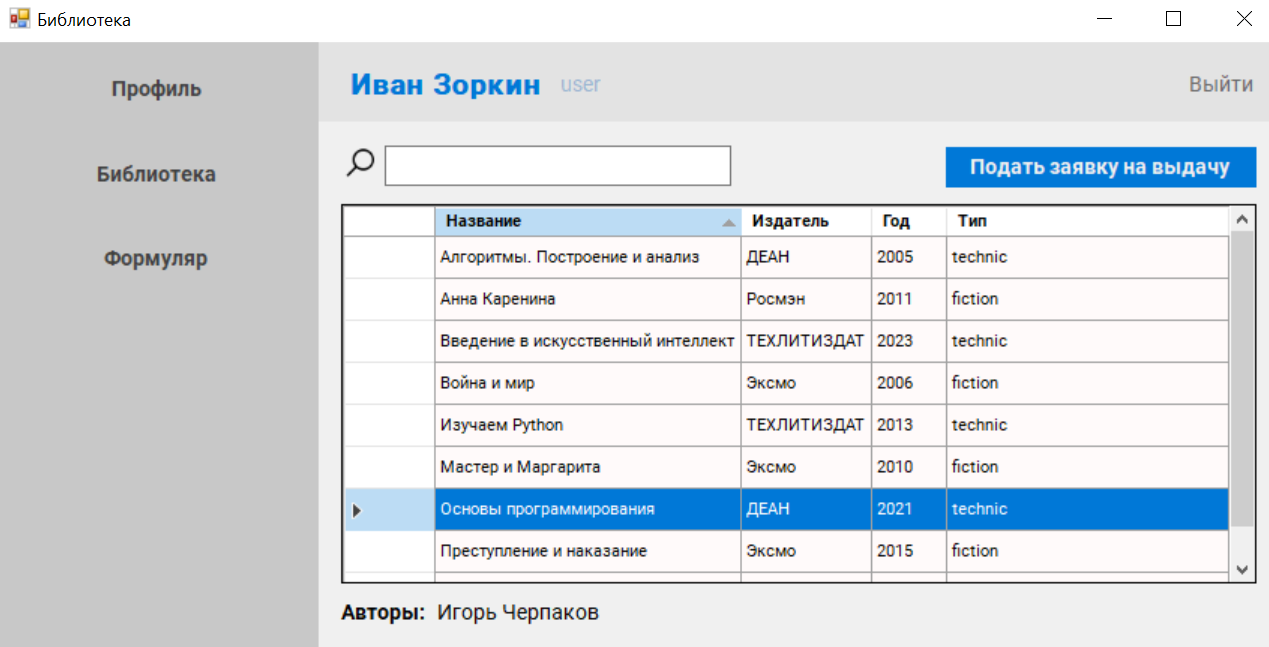


Рисунок 4.4 – Окно библиотеки

Последнее окно интерфейса пользователя – окно формуляра (Рисунок 4.5). Формуляр содержит историю всех выдач и заявок текущего пользователя. Окно не предусматривает никаких взаимодействий помимо просмотра и сортировки данных. Также после работы с предыдущим окном в нём были отражены поданные заявки.

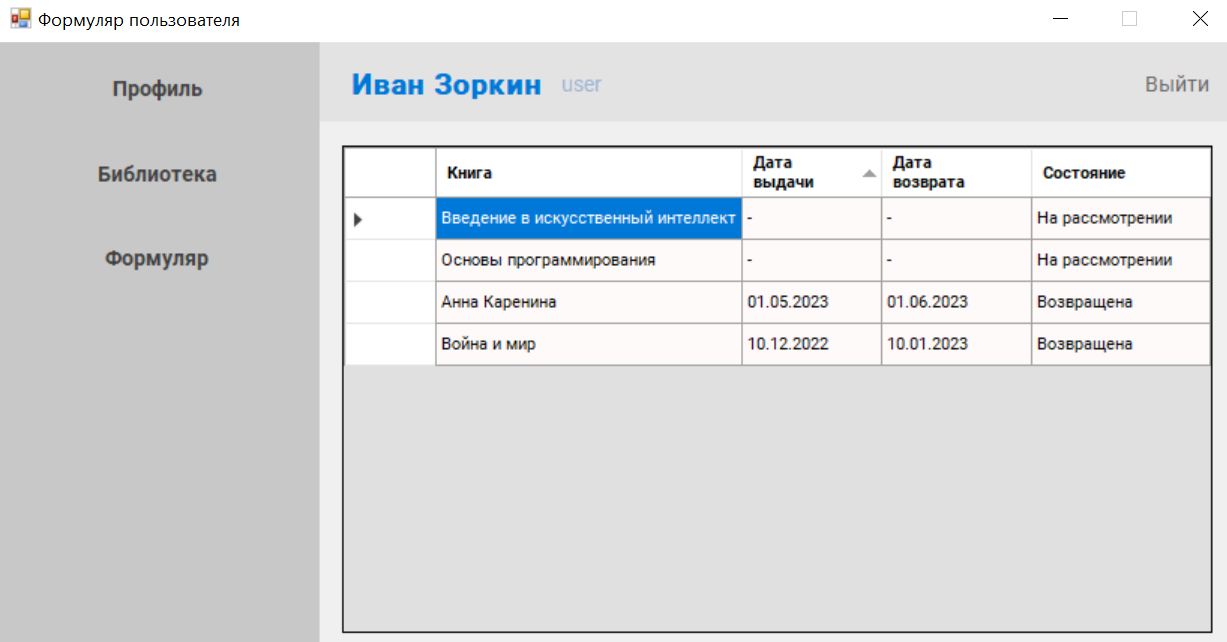


Рисунок 4.5 – Окно формуляра

Через кнопку выйти был осуществлён возврат на окно авторизации. Был осуществлён вход в аккаунт администратора. Через него было открыто окно списка пользователей (Рисунок 4.6).

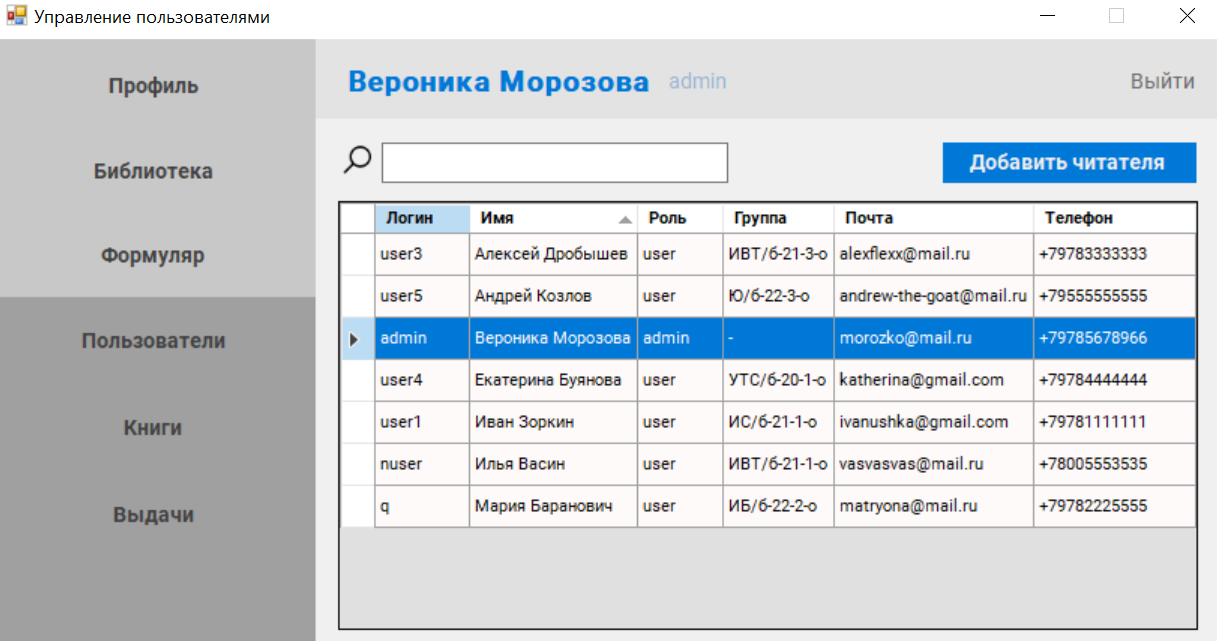


Рисунок 4.6 – Окно списка пользователей

В нём представлена информация обо всех зарегистрированных в системе пользователях. Как и в других окнах, в нём реализована выдача результатов по поисковому запросу (Рисунок 4.7).

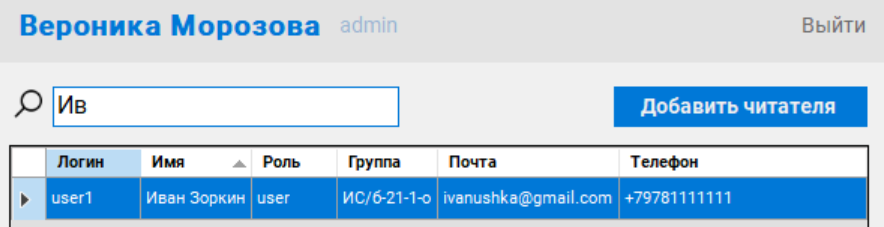


Рисунок 4.7 – Поиск пользователей по запросу

Редактировать имеющихся пользователей нельзя, но можно добавлять новых – для этого при нажатии соответствующей кнопки открывается специальное окно (Рисунок 4.8).

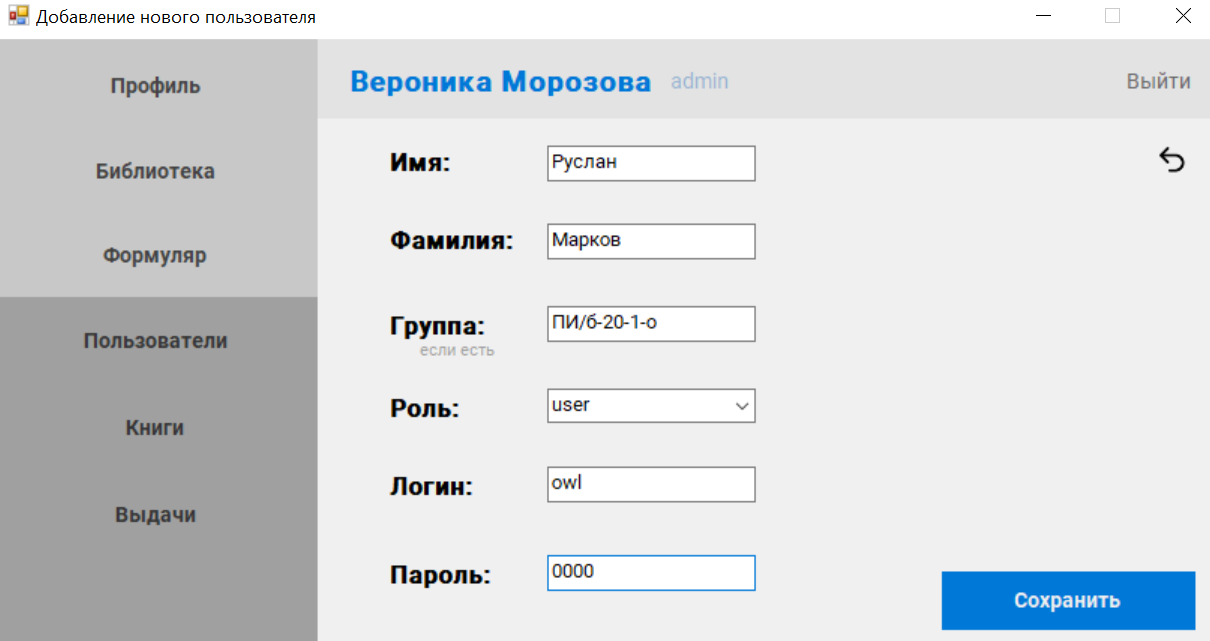


Рисунок 4.8 – Окно добавления пользователя

Также админу доступно окно со списком всех содержащихся в базе данных книг (Рисунок 4.9). В нём доступны поиск по запросам, добавление новых книг и редактирование информации о старых (Рисунок 3.11). Выбор книги осуществляется нажатием на строки таблицы.

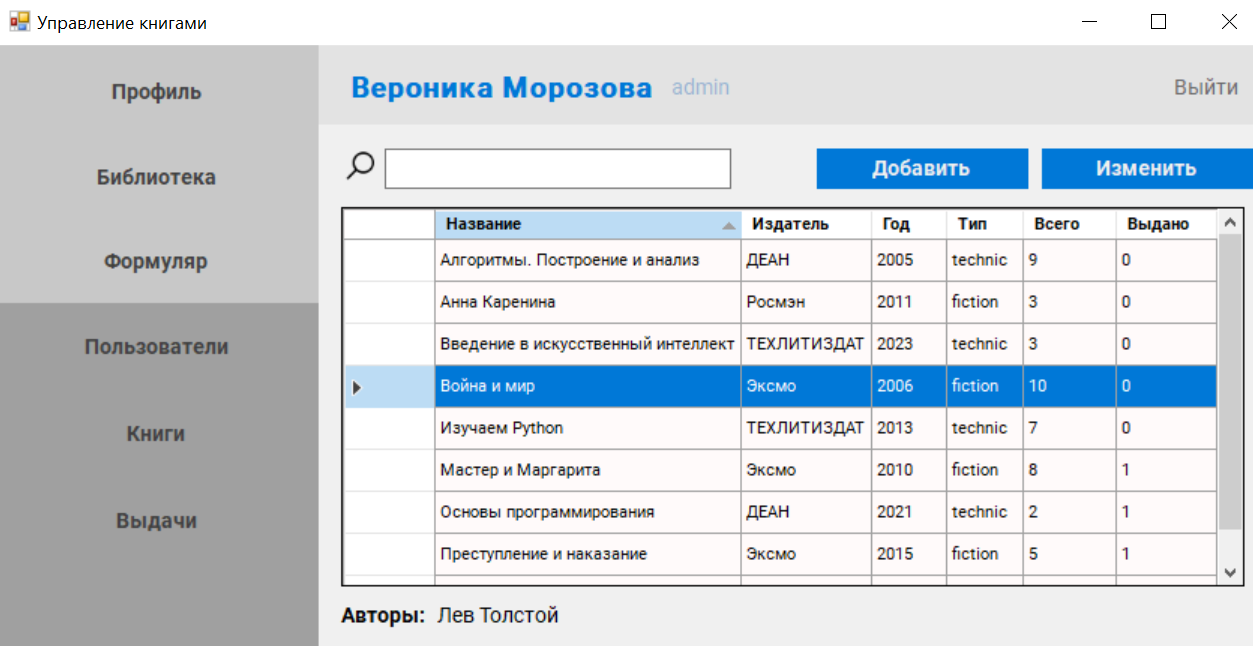


Рисунок 4.9 – Окно информации о книгах

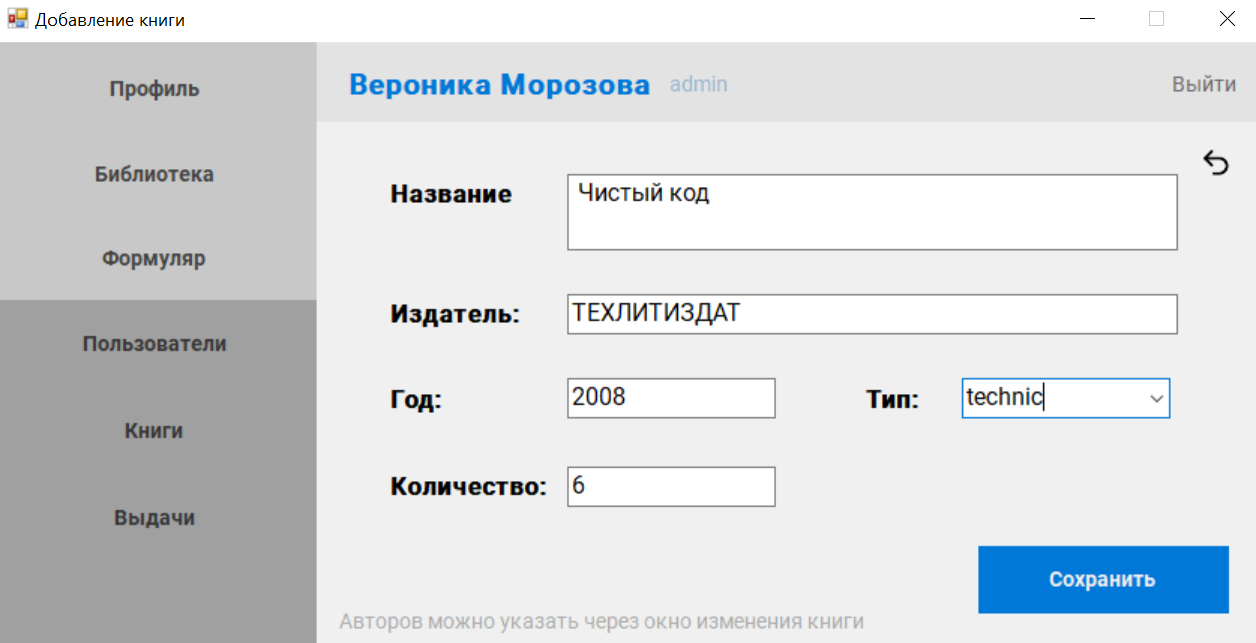


Рисунок 3.10 – Окно добавления книги

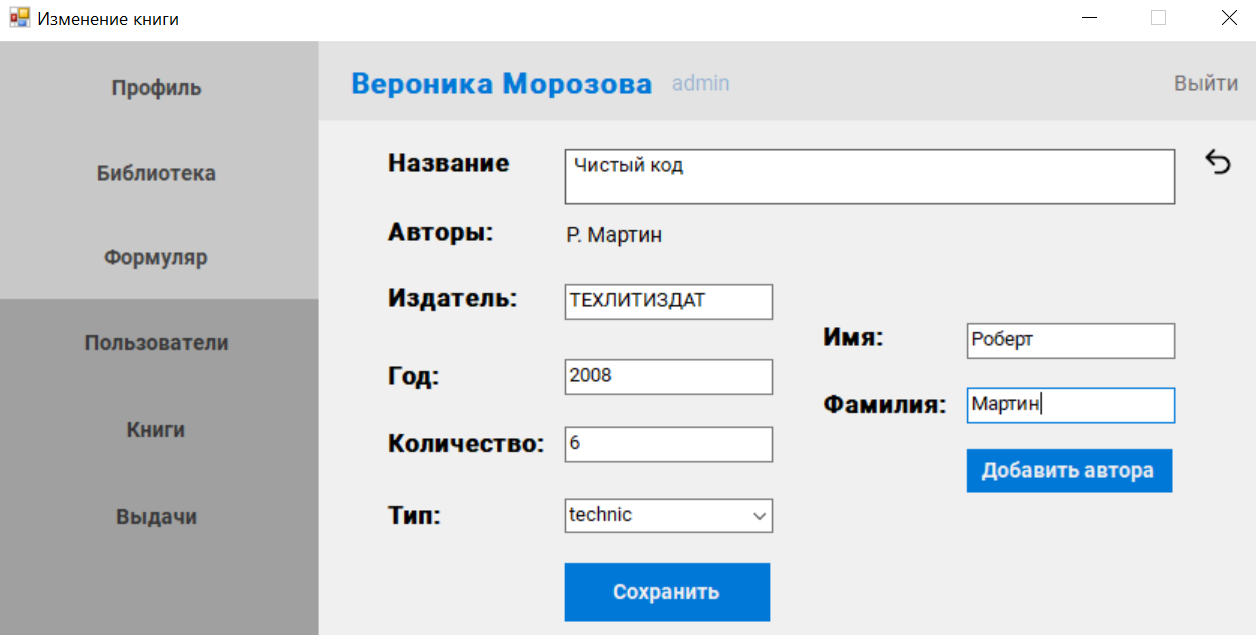


Рисунок 3.11 – Окно редактирования книги

Последнее окно – окно истории выдач (Рисунок 3.12). В нём расположена таблица, содержащая список всех проведённых выдач со всеми пользователями.

Текущие запросы можно отклонять и подтверждать, для выданных книг можно оформить возврат, либо проверить нужно ли выписать читателю штраф за невозврат книги вовремя.

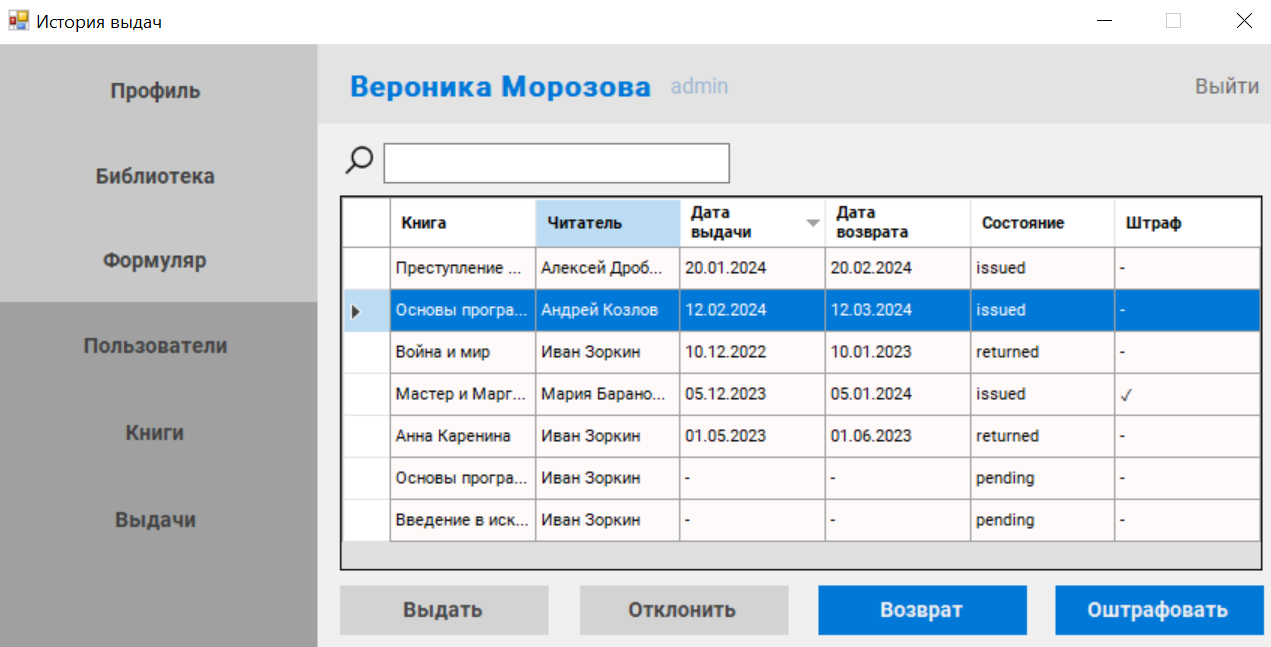


Рисунок 3.12 – Окно истории выдач

По результатам тестирования можно сделать вывод, что программа работает так, как задумывалось на этапе изучения предметной области.

# ЗАКЛЮЧЕНИЕ

В ходе выполнения курсовой работы была разработана базы данных для библиотеки образовательного учреждения (университета). Процесс разработки был разбит на следующие этапы: аналитическая часть (анализ предметной области и постановка задачи), разработка логической модели (построение диаграмм «сущность-связь» и модели на ключах, нормализация), разработка физической модели и создание клиентского приложения с интерфейсами пользователя и администратора.

Физическая модель была реализована на языке SQL через СУБД PostgreSQL. Разработка клиентского приложения была проведена на языке C# через платформу Windows Forms фреймворка .NET. Написание кода и тестирование приложения велись в IDE Visual Studio 2022.

В результате проделанной работы была создана база данных для библиотеки и соответствующее приложение для управления ею. Таким образом были получены и отработаны практические и теоретические навыки концептуального, логического и физического проектирования баз данных, а также их интеграции в клиентские приложения.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Воройский Ф.С. Основы проектирования автоматизированных библиотечно-информационных систем. — 2-е изд., доп. и перераб. — М.: ФИЗМАТЛИТ, 2008. — 456 с. — ISBN 978-5-9221-0846-1.
2. Бьюли А. Изучаем SQL. — Пер. с англ. — СПб: Символ-Плюс, 2007. — 312 с., ил. ISBN-13: 978-5-93286-051-9.
3. Шилдс Уолтер. SQL: быстрое погружение. — СПб.: Питер, 2022. — 224 с.: ил. — (Серия «Библиотека программиста»).
4. PostgreSQL 16.2 Documentation. The PostgreSQL Global Development Group (официальная документация языка PostgreSQL).
5. Windows Forms Documentation [Электронный ресурс] — <https://learn.microsoft.com/en-us/dotnet/desktop/winforms/?view=netdesktop-8.0>
6. Методические указания к лабораторным занятиям по дисциплине «Управление днными» / Сост. Ю.В. Доронина, А.В. Волкова – Севастополь: Изд-во СевГУ, 2018 – 24 с.
7. Cистема вопросов и ответов о программировании StackOverFlow (на русском) [Электронный ресурс] — <https://ru.stackoverflow.com/>
8. Вопросы и ответы по IT-темам: Overcoder [Электронный ресурс] — <https://overcoder.net>
9. Система вопросов и ответов о программировании CodeNet [Электронный ресурс] — <http://forum.codenet.ru/>
10. Форум программистов и сисадминов: Киберфорум [Электронный ресурс] — <https://www.cyberforum.ru/cpp-beginners/>

# ПРИЛОЖЕНИЕ А

СОЗДАНИЕ ТАБЛИЦ ДЛЯ БАЗЫ ДАННЫХ

Листинг А.1 – Создание таблицы «Читатели»

CREATE TABLE IF NOT EXISTS public.readers

(

id integer NOT NULL GENERATED ALWAYS AS IDENTITY ( INCREMENT 1 START 1 MINVALUE 1 MAXVALUE 2147483647 CACHE 1 ),

first\_name character varying NOT NULL,

last\_name character varying NOT NULL,

group\_code character varying,

phone character varying COLLATE,

email character varying(256) COLLATE,

fined boolean NOT NULL DEFAULT false,

CONSTRAINT reader\_pkey PRIMARY KEY (id),

CONSTRAINT email CHECK (email ~\* '^[A-Za-z0-9.\_%+-]+@[A-Za-z0-9.-]+\.[A-Za-z]{2,}$'),

CONSTRAINT phone CHECK (phone ~ '^\+[0-9]{11,11}$'::text),

CONSTRAINT group\_code CHECK (group\_code ~\* '[А-Я]{1,3}/б-[0-9]{2}-[0-9]-[а-я]{1,2}$')

)

TABLESPACE pg\_default;

ALTER TABLE IF EXISTS public.readers

OWNER to postgres;

Листинг А.2 – Создание таблицы «Книги»

CREATE TABLE IF NOT EXISTS public.books

(

id integer NOT NULL GENERATED ALWAYS AS IDENTITY ( INCREMENT 1 START 1 MINVALUE 1 MAXVALUE 2147483647 CACHE 1 ),

title character varying COLLATE NOT NULL,

publisher character varying COLLATE,

book\_year integer,

amount integer NOT NULL DEFAULT 0,

type character varying COLLATE,

CONSTRAINT books\_pkey PRIMARY KEY (id)

)

TABLESPACE pg\_default;

ALTER TABLE IF EXISTS public.books

OWNER to postgres;

Листинг А.3 – Создание таблицы «Выдача»

CREATE TABLE IF NOT EXISTS public.issuance

(

id integer NOT NULL GENERATED ALWAYS AS IDENTITY ( INCREMENT 1 START 1 MINVALUE 1 MAXVALUE 2147483647 CACHE 1 ),

book\_id integer NOT NULL,

reader\_id integer NOT NULL,

book\_date date,

return\_date date,

status character varying NOT NULL DEFAULT 'pending',

CONSTRAINT issuance\_pkey PRIMARY KEY (id),

CONSTRAINT issuance\_book\_id\_fkey FOREIGN KEY (book\_id)

REFERENCES public.books (id) MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION,

CONSTRAINT issuance\_reader\_id\_fkey FOREIGN KEY (reader\_id)

REFERENCES public.readers (id) MATCH SIMPLE

ON UPDATE CASCADE

ON DELETE CASCADE

)

TABLESPACE pg\_default;

ALTER TABLE IF EXISTS public.issuance

OWNER to postgres;

Листинг А.4 – Создание таблицы «Авторы»

CREATE TABLE IF NOT EXISTS public.authors

(

id integer NOT NULL GENERATED ALWAYS AS IDENTITY ( INCREMENT 1 START 1 MINVALUE 1 MAXVALUE 2147483647 CACHE 1 ),

first\_name character varying COLLATE pg\_catalog."default",

last\_name character varying COLLATE pg\_catalog."default",

CONSTRAINT authors\_pkey PRIMARY KEY (id)

)

TABLESPACE pg\_default;

ALTER TABLE IF EXISTS public.authors

OWNER to postgres;

Листинг А.5 – Создание таблицы «Книга-Автор»

CREATE TABLE IF NOT EXISTS public.book\_author

(

book\_id integer NOT NULL,

author\_id integer NOT NULL,

CONSTRAINT book\_author\_pkey PRIMARY KEY (book\_id, author\_id),

CONSTRAINT book\_author\_author\_id\_fkey FOREIGN KEY (author\_id)

REFERENCES public.authors (id) MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION,

CONSTRAINT book\_author\_book\_id\_fkey FOREIGN KEY (book\_id)

REFERENCES public.books (id) MATCH SIMPLE

ON UPDATE CASCADE

ON DELETE CASCADE

)

TABLESPACE pg\_default;

ALTER TABLE IF EXISTS public.book\_author

OWNER to postgres;

Листинг А.6 – Представление «Количество книг»

CREATE OR REPLACE VIEW public.books\_amount

AS

SELECT books.id,

books.title,

books.amount AS total\_amount,

COALESCE(issued.amount, 0) AS issued\_amount,

books.amount - COALESCE(issued.amount, 0) AS available\_amount

FROM books

LEFT JOIN ( SELECT issuance.book\_id,

count(\*) AS amount

FROM issuance

WHERE issuance.status = 'issued'

GROUP BY issuance.book\_id) issued ON books.id = issued.book\_id;

ALTER TABLE public.books\_amount

OWNER TO postgres;

# ПРИЛОЖЕНИЕ Б

ИСХОДНЫЙ КОД КЛИЕНТСКОГО ПРИЛОЖЕНИЯ

Листинг Б.1 - Program.cs (точка входа в программу)

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace LibrarySQLApp

{

internal static class Program

{

/// <summary>

/// Главная точка входа для приложения.

/// </summary>

[STAThread]

static void Main()

{

Application.EnableVisualStyles();

Application.SetCompatibleTextRenderingDefault(false);

Application.Run(new MainForm());

}

}

}

Листинг Б.2 - DB.cs (подключение к базе данных)

using System;

using System.Collections.Generic;

using System.Data;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using Npgsql;

namespace LibrarySQLApp

{

public static class DB

{

static NpgsqlConnection conn = new NpgsqlConnection("Server=localhost; Port=5433; User Id=postgres; Password=0000; Database=library;");

public static void openConnection()

{

if (conn.State == ConnectionState.Closed)

{

conn.Open();

}

}

public static void closeConnection()

{

if (conn.State == ConnectionState.Open)

{

conn.Close();

}

}

public static NpgsqlConnection getConnection()

{

return conn;

}

}

}

Листинг Б.3 – Messages.cs (всплывающие сообщения)

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace LibrarySQLApp

{

public static class Messages

{

public static void DisplayErrorMessage(string message)

{

MessageBox.Show(message, "Ошибка", MessageBoxButtons.OK, MessageBoxIcon.Warning);

}

public static void DisplayInfoMessage(string message)

{

MessageBox.Show(message, "Информация", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

public static DialogResult DisplayQuestionMessage(string message)

{

DialogResult dr = MessageBox.Show(message, "Подтверждение", MessageBoxButtons.YesNo, MessageBoxIcon.Question);

return dr;

}

}

}

Листинг Б.4 − User.cs (пользователь системы)

using Npgsql;

using System;

using System.Collections.Generic;

using System.Data;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LibrarySQLApp

{

public class User

{

public string Login { get; set; }

public string Password { get; set; }

public string Role { get; set; }

public int ID { get; set; }

public string FullName { get; set; }

public string GroupCode { get; set; }

public string Phone { get; set; }

public string Email { get; set; }

public bool FineStatus { get; set; }

public User(string login, string password, string role, int readerID)

{

Login = login;

Password = password;

Role = role;

String query = "select \* from readers where id=@id;";

DataTable table = new DataTable();

NpgsqlDataAdapter adapter = new NpgsqlDataAdapter();

NpgsqlCommand command = new NpgsqlCommand(query, DB.getConnection());

command.Parameters.AddWithValue("@id", readerID);

adapter.SelectCommand = command;

adapter.Fill(table);

ID = (int)table.Rows[0]["id"];

FullName =

table.Rows[0]["first\_name"].ToString() + " " +

table.Rows[0]["last\_name"].ToString();

GroupCode = table.Rows[0]["group\_code"].ToString();

Phone = table.Rows[0]["phone"].ToString();

Email = table.Rows[0]["email"].ToString();

FineStatus = Convert.ToBoolean(table.Rows[0]["fined"]);

}

}

}

Листинг Б.5 − Navigation.cs (навигация между окнами)

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LibrarySQLApp

{

public static class Navigation

{

public static MainForm MainForm { get; set; }

public static UserForm UserForm { get; set; }

public static BooksForm BooksForm { get; set; }

public static UserStoryForm UserStoryForm { get; set; }

public static UsersAdminForm UsersAdminForm { get; set; }

public static BooksAdminForm BooksAdminForm { get; set; }

public static IssuanceAdminForm IssuanceAdminForm { get; set; }

public static void CloseUI()

{

UserForm.isClosedByCode = true;

BooksForm.isClosedByCode = true;

UserStoryForm.isClosedByCode = true;

UserForm.Close();

BooksForm.Close();

UserStoryForm.Close();

}

public static void CloseAdmin()

{

UsersAdminForm.isClosedByCode = true;

BooksAdminForm.isClosedByCode = true;

IssuanceAdminForm.isClosedByCode = true;

UsersAdminForm.Close();

BooksAdminForm.Close();

IssuanceAdminForm.Close();

}

}

}

Листинг Б.6 − MainForm.cs (окно авторизации)

using System;

using System.Collections;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Runtime.Remoting.Messaging;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using Npgsql;

using static System.Windows.Forms.VisualStyles.VisualStyleElement.ListView;

namespace LibrarySQLApp

{

public partial class MainForm : Form

{

public MainForm()

{

InitializeComponent();

Navigation.MainForm = this;

}

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

{

String login = loginField.Text;

String password = passField.Text;

if (login == "" || password == "")

{

Messages.DisplayErrorMessage("Заполните все поля!");

return;

}

String query =

"select \* from accounts " +

"where login=@login and password=@password;";

DataTable table = new DataTable();

NpgsqlDataAdapter adapter = new NpgsqlDataAdapter();

NpgsqlCommand command = new NpgsqlCommand(query, DB.getConnection());

command.Parameters.AddWithValue("@login", login);

command.Parameters.AddWithValue("@password", password);

adapter.SelectCommand = command;

adapter.Fill(table);

if (table.Rows.Count > 0)

{

loginField.Text = "";

passField.Text = "";

String role = table.Rows[0]["role"].ToString();

int readerID = (int)table.Rows[0]["reader\_id"];

User user = new User(login, password, role, readerID);

Navigation.UserForm = new UserForm(user);

Navigation.BooksForm = new BooksForm(user);

Navigation.UserStoryForm = new UserStoryForm(user);

if (role == "admin")

{

Navigation.UsersAdminForm = new UsersAdminForm(user);

Navigation.BooksAdminForm = new BooksAdminForm(user);

Navigation.IssuanceAdminForm = new IssuanceAdminForm(user);

}

this.Hide();

Navigation.UserForm.Show();

Navigation.UserForm.Location = this.Location;

}

else

Messages.DisplayErrorMessage("Неверный логин или пароль.");

//

}

}

}

Листинг Б.7 − UserForm.cs (окно профиля)

using Npgsql;

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Text.RegularExpressions;

using System.Threading.Tasks;

using System.Windows.Forms;

using static System.Windows.Forms.VisualStyles.VisualStyleElement;

using static System.Windows.Forms.VisualStyles.VisualStyleElement.ListView;

namespace LibrarySQLApp

{

public partial class UserForm : Form

{

User User { get; set; }

public bool isClosedByCode = false;

public UserForm(User user)

{

this.User = user;

InitializeComponent();

nameLabel.Text = User.FullName;

roleLabel.Text = User.Role;

groupLabel.Text = (User.GroupCode != "") ? User.GroupCode : "-";

loginTextBox.Text = User.Login;

emailTextBox.Text = User.Email;

phoneTextBox.Text = User.Phone;

if (User.Role != "admin")

{

adminPanel.Hide();

}

}

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

{

isClosedByCode = true;

Navigation.CloseUI();

if (User.Role == "admin") Navigation.CloseAdmin();

Navigation.MainForm.Show();

Navigation.MainForm.Location = this.Location;

}

private void exitLable\_MouseEnter(object sender, EventArgs e)

{

exitLable.ForeColor = Color.Black;

}

private void exitLable\_MouseLeave(object sender, EventArgs e)

{

exitLable.ForeColor = Color.DimGray;

}

private void UserForm\_FormClosing(object sender, FormClosingEventArgs e)

{

if (!this.isClosedByCode)

{

Application.Exit();

}

}

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

{

this.Hide();

Navigation.UserForm.Show();

Navigation.UserForm.Location = this.Location;

}

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

{

this.Hide();

Navigation.BooksForm.Show();

Navigation.BooksForm.Location = this.Location;

}

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

{

this.Hide();

Navigation.UserStoryForm.Show();

Navigation.UserStoryForm.Location = this.Location;

}

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

{

this.Hide();

Navigation.UsersAdminForm.Show();

Navigation.UsersAdminForm.Location = this.Location;

}

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

{

this.Hide();

Navigation.BooksAdminForm.Show();

Navigation.BooksAdminForm.Location = this.Location;

}

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

{

this.Hide();

Navigation.IssuanceAdminForm.Show();

Navigation.IssuanceAdminForm.Location = this.Location;

}

private void checkSaveButtonEnability()

{

if (loginTextBox.Text != User.Login ||

emailTextBox.Text != User.Email ||

phoneTextBox.Text != User.Phone)

{

SaveButton.Enabled = true;

SaveButton.BackColor = SystemColors.HotTrack;

}

else

{

SaveButton.Enabled = false;

SaveButton.BackColor = Color.LightGray;

}

}

private void loginTextBox\_TextChanged(object sender, EventArgs e)

{

if (!Regex.IsMatch(loginTextBox.Text, @"^[A-Za-z0-9\_]+$") || loginTextBox.Text == "")

{

loginTextBox.ForeColor = Color.Red;

SaveButton.Enabled = false;

SaveButton.BackColor = Color.LightGray;

}

else if (loginTextBox.Text != User.Login)

{

loginTextBox.ForeColor = SystemColors.Highlight;

checkSaveButtonEnability();

}

else

{

loginTextBox.ForeColor = Color.Gray;

checkSaveButtonEnability();

}

}

private void emailTextBox\_TextChanged(object sender, EventArgs e)

{

if (!Regex.IsMatch(emailTextBox.Text, @"^[A-Za-z0-9.\_%+-]+@[A-Za-z0-9.-]+\.[A-Za-z]{2,}$"))

{

emailTextBox.ForeColor = Color.Red;

SaveButton.Enabled = false;

SaveButton.BackColor = Color.LightGray;

}

else if (emailTextBox.Text != User.Email)

{

emailTextBox.ForeColor = SystemColors.Highlight;

checkSaveButtonEnability();

}

else

{

emailTextBox.ForeColor = Color.Gray;

checkSaveButtonEnability();

}

}

private void phoneTextBox\_TextChanged(object sender, EventArgs e)

{

if (!Regex.IsMatch(phoneTextBox.Text, @"^\+[0-9]{11,11}$"))

{

phoneTextBox.ForeColor = Color.Red;

SaveButton.Enabled = false;

SaveButton.BackColor = Color.LightGray;

}

else if (phoneTextBox.Text != User.Phone)

{

phoneTextBox.ForeColor = SystemColors.Highlight;

checkSaveButtonEnability();

}

else

{

phoneTextBox.ForeColor = Color.Gray;

checkSaveButtonEnability();

}

}

private void newPassTextBox\_TextChanged(object sender, EventArgs e)

{

if (oldPassTextBox.Text != "" && newPassTextBox.Text != "")

{

PassChangeButton.Enabled = true;

PassChangeButton.BackColor = SystemColors.HotTrack;

}

else

{

PassChangeButton.Enabled = false;

PassChangeButton.BackColor = Color.LightGray;

}

}

private void oldPassTextBox\_TextChanged(object sender, EventArgs e)

{

if (oldPassTextBox.Text != "" && newPassTextBox.Text != "")

{

PassChangeButton.Enabled = true;

PassChangeButton.BackColor = SystemColors.HotTrack;

}

else

{

PassChangeButton.Enabled = false;

PassChangeButton.BackColor = Color.LightGray;

}

}

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

{

if (oldPassTextBox.Text == User.Password)

{

String query =

"update accounts " +

"set password=@password " +

"where reader\_id=@id;";

try

{

DB.openConnection();

NpgsqlCommand command = new NpgsqlCommand(query, DB.getConnection());

command.Parameters.AddWithValue("@id", User.ID);

command.Parameters.AddWithValue("@password", newPassTextBox.Text);

command.ExecuteNonQuery();

Messages.DisplayInfoMessage("Пароль успешно изменён");

DB.closeConnection();

User.Password = newPassTextBox.Text;

newPassTextBox.Text = "";

oldPassTextBox.Text = "";

PassChangeButton.Enabled = false;

PassChangeButton.BackColor = Color.LightGray;

}

catch (Exception ex)

{

Messages.DisplayErrorMessage($"Ошибка: {ex.Message}");

}

finally

{

DB.closeConnection();

}

}

else

{

Messages.DisplayErrorMessage("Неверный пароль");

}

}

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

{

String login = loginTextBox.Text;

String email = emailTextBox.Text;

String phone = phoneTextBox.Text;

String readersQuery =

"update readers " +

"set email=@email, phone=@phone " +

"where id=@id;";

String accountsQuery =

"update accounts " +

"set login=@login " +

"where reader\_id=@id;";

NpgsqlTransaction transaction = null;

try

{

DB.openConnection();

transaction = DB.getConnection().BeginTransaction();

NpgsqlCommand command = new NpgsqlCommand();

command.Connection = DB.getConnection();

command.Transaction = transaction;

command.CommandText = readersQuery;

command.Parameters.AddWithValue("@email", email);

command.Parameters.AddWithValue("@phone", phone);

command.Parameters.AddWithValue("@id", User.ID);

command.ExecuteNonQuery();

command.CommandText = accountsQuery;

command.Parameters.AddWithValue("@login", login);

command.Parameters.AddWithValue("@id", User.ID);

command.ExecuteNonQuery();

transaction.Commit();

Messages.DisplayInfoMessage("Данные сохранены.");

SaveButton.Enabled = false;

SaveButton.BackColor = Color.LightGray;

loginTextBox.ForeColor = Color.Gray;

emailTextBox.ForeColor = Color.Gray;

phoneTextBox.ForeColor = Color.Gray;

User.Login = login;

User.Email = email;

User.Phone = phone;

}

catch (Exception ex)

{

if (transaction != null)

{

transaction.Rollback();

}

Messages.DisplayErrorMessage($"Ошибка: {ex.Message}");

}

finally

{

if (transaction != null)

{

transaction.Dispose();

}

DB.closeConnection();

}

}

}

}

Листинг Б.8 − BooksForm.cs (окно библиотеки)

using Npgsql;

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace LibrarySQLApp

{

public partial class BooksForm : Form

{

User User { get; set; }

public bool isClosedByCode = false;

public BooksForm(User user)

{

this.User = user;

InitializeComponent();

nameLabel.Text = User.FullName;

roleLabel.Text = User.Role;

CreateGridView();

LoadGridView();

if (User.Role != "admin")

{

adminPanel.Hide();

}

}

private void CreateGridView()

{

MainGridView.Columns.Add("title", "Название");

MainGridView.Columns.Add("publisher", "Издатель");

MainGridView.Columns.Add("year", "Год");

MainGridView.Columns.Add("type", "Тип");

MainGridView.Columns.Add("id", String.Empty);

MainGridView.Columns["id"].Visible = false;

}

private void FillGridRow(DataGridView dgv, IDataRecord record)

{

dgv.Rows.Add(

record.GetString(0),

record.GetString(1),

record.GetInt32(2),

record.GetString(3),

record.GetInt32(4));

}

public void LoadGridView()

{

MainGridView.Rows.Clear();

DataTable dataTable = new DataTable();

string query =

"select books.title, publisher, book\_year, type, books.id " +

"from books join books\_amount as ba on ba.id = books.id " +

"where ba.available\_amount > 0;";

DB.openConnection();

NpgsqlCommand command = new NpgsqlCommand(query, DB.getConnection());

NpgsqlDataReader reader = command.ExecuteReader();

while (reader.Read())

{

FillGridRow(MainGridView, reader);

}

DB.closeConnection();

MainGridView.Sort(MainGridView.Columns["title"], ListSortDirection.Ascending);

foreach (DataGridViewColumn column in MainGridView.Columns)

{

column.AutoSizeMode = DataGridViewAutoSizeColumnMode.AllCells;

column.DefaultCellStyle.WrapMode = DataGridViewTriState.True;

}

MainGridView.Columns[3].AutoSizeMode = DataGridViewAutoSizeColumnMode.Fill;

PrintAuthors();

}

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

{

isClosedByCode = true;

Navigation.CloseUI();

if (User.Role == "admin") Navigation.CloseAdmin();

Navigation.MainForm.Show();

Navigation.MainForm.Location = this.Location;

}

private void exitLable\_MouseEnter(object sender, EventArgs e)

{

exitLable.ForeColor = Color.Black;

}

private void exitLable\_MouseLeave(object sender, EventArgs e)

{

exitLable.ForeColor = Color.DimGray;

}

private void UserForm\_FormClosing(object sender, FormClosingEventArgs e)

{

if (!this.isClosedByCode)

{

Application.Exit();

}

}

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

{

this.Hide();

Navigation.UserForm.Show();

Navigation.UserForm.Location = this.Location;

}

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

{

this.Hide();

Navigation.BooksForm.Show();

Navigation.BooksForm.Location = this.Location;

}

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

{

this.Hide();

Navigation.UserStoryForm.Show();

Navigation.UserStoryForm.Location = this.Location;

}

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

{

this.Hide();

Navigation.UsersAdminForm.Show();

Navigation.UsersAdminForm.Location = this.Location;

}

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

{

this.Hide();

Navigation.BooksAdminForm.Show();

Navigation.BooksAdminForm.Location = this.Location;

}

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

{

this.Hide();

Navigation.IssuanceAdminForm.Show();

Navigation.IssuanceAdminForm.Location = this.Location;

}

private void MainGridView\_CellClick(object sender, DataGridViewCellEventArgs e)

{

PrintAuthors();

}

public void PrintAuthors()

{

DataGridViewRow row;

if (MainGridView.CurrentCell != null)

{

row = MainGridView.Rows[MainGridView.CurrentCell.RowIndex];

}

else

{

row = MainGridView.Rows[0];

}

string query = "" +

"select first\_name, last\_name from authors " +

"join book\_author as ba on ba.author\_id = authors.id " +

"join books on books.id = ba.book\_id " +

"where books.id = @id";

List<string> authors = new List<string>();

DB.openConnection();

NpgsqlCommand command = new NpgsqlCommand(query, DB.getConnection());

command.Parameters.AddWithValue("@id", (int)row.Cells[4].Value);

NpgsqlDataReader reader = command.ExecuteReader();

while (reader.Read())

{

authors.Add(reader.GetString(0) + " " + reader.GetString(1));

}

DB.closeConnection();

authorsList.Text = string.Join(", ", authors);

}

private void SearchBox\_TextChanged(object sender, EventArgs e)

{

if (SearchBox.Text.Length <= 0)

{

LoadGridView();

PrintAuthors();

return;

}

MainGridView.Rows.Clear();

DataTable dataTable = new DataTable();

string query =

"select distinct b.title, publisher, book\_year, type, b.id " +

"from books as b " +

"join book\_author as ban on b.id = ban.book\_id " +

"join authors as a on ban.author\_id = a.id " +

"join books\_amount as bam on b.id = bam.id " +

"where b.title like '%" + SearchBox.Text + "%' or " +

"b.publisher like '%" + SearchBox.Text + "%' or " +

"a.first\_name like '%" + SearchBox.Text + "%' or " +

"a.last\_name like '%" + SearchBox.Text + "%';";

DB.openConnection();

NpgsqlCommand command = new NpgsqlCommand(query, DB.getConnection());

NpgsqlDataReader reader = command.ExecuteReader();

while (reader.Read())

{

FillGridRow(MainGridView, reader);

}

DB.closeConnection();

MainGridView.Sort(MainGridView.Columns["title"], ListSortDirection.Ascending);

foreach (DataGridViewColumn column in MainGridView.Columns)

{

column.AutoSizeMode = DataGridViewAutoSizeColumnMode.AllCells;

column.DefaultCellStyle.WrapMode = DataGridViewTriState.True;

}

MainGridView.Columns[3].AutoSizeMode = DataGridViewAutoSizeColumnMode.Fill;

if (MainGridView.CurrentCell != null)

{

PrintAuthors();

return;

}

authorsList.Text = "выберите книгу";

}

private void BooksForm\_Activated(object sender, EventArgs e)

{

LoadGridView();

}

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

{

DataGridViewRow row;

if (MainGridView.CurrentCell == null)

{

Messages.DisplayErrorMessage("Выберите книгу!");

return;

}

row = MainGridView.Rows[MainGridView.CurrentCell.RowIndex];

string countQuery = "" +

"select count(\*) from issuance " +

"where reader\_id = @id and " +

"status = @status;";

DB.openConnection();

NpgsqlCommand command = new NpgsqlCommand(countQuery, DB.getConnection());

command.Parameters.AddWithValue("@id", User.ID);

command.Parameters.AddWithValue("@status", "issued");

object result = command.ExecuteScalar();

int issued = (result != null) ? Convert.ToInt32(result) : 0;

command = new NpgsqlCommand(countQuery, DB.getConnection());

command.Parameters.AddWithValue("@id", User.ID);

command.Parameters.AddWithValue("@status", "pending");

result = command.ExecuteScalar();

int pending = (result != null) ? Convert.ToInt32(result) : 0;

DB.closeConnection();

if (issued != 0)

{

Messages.DisplayErrorMessage("" +

"Невозможно оформить заявку. Имеются невозвращённые книги");

return;

}

if (pending > 2)

{

Messages.DisplayErrorMessage("" +

"Предел заявок от читателя достиг предела. " +

"Дождитесь ответа администратора.");

return;

}

string insertQuery = "" +

"insert into issuance (book\_id, reader\_id, status) " +

"select @book\_id, @reader\_id, 'pending' " +

"where not exists (select 1 from issuance where book\_id = " +

"@book\_id and reader\_id = @reader\_id);";

try

{

DB.openConnection();

command = new NpgsqlCommand(insertQuery, DB.getConnection());

command.Parameters.AddWithValue("@book\_id", (int)row.Cells[4].Value);

command.Parameters.AddWithValue("@reader\_id", User.ID);

command.ExecuteNonQuery();

DB.closeConnection();

Messages.DisplayInfoMessage("Заявка успешно отправлена");

Navigation.UserStoryForm.LoadGridView();

}

catch (Exception ex)

{

Messages.DisplayErrorMessage(ex.Message);

}

}

}

}

Листинг Б.9 − UserStoryForm.cs (окно формуляра)

using Npgsql;

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace LibrarySQLApp

{

public partial class UserStoryForm : Form

{

User User { get; set; }

public bool isClosedByCode = false;

public UserStoryForm(User user)

{

this.User = user;

InitializeComponent();

nameLabel.Text = User.FullName;

roleLabel.Text = User.Role;

if (User.Role != "admin")

{

adminPanel.Hide();

}

CreateGridView();

LoadGridView();

}

private void CreateGridView()

{

MainGridView.Columns.Add("title", "Книга");

MainGridView.Columns.Add("book\_date", "Дата выдачи");

MainGridView.Columns.Add("return\_date", "Дата возврата");

MainGridView.Columns.Add("status", "Состояние");

}

private enum Status

{

Pending,

Issued,

Returned,

Rejected

}

private void FillGridRow(DataGridView dgv, IDataRecord record)

{

Dictionary<Status, string> statusMappings = new Dictionary<Status, string>()

{

{ Status.Pending, "На рассмотрении" },

{ Status.Issued, "Выдана" },

{ Status.Returned, "Возвращена" },

{ Status.Rejected, "Отказано" },

};

Status status = (Status)Enum.Parse(typeof(Status), record.GetString(3), true);

string statusText = statusMappings[status];

dgv.Rows.Add(

record.GetString(0),

record.IsDBNull(1) ? "-" : record.GetDateTime(1).Date.ToString("d"),

record.IsDBNull(2) ? "-": record.GetDateTime(2).Date.ToString("d"),

statusText);

}

public void LoadGridView()

{

MainGridView.Rows.Clear();

DataTable dataTable = new DataTable();

string query = "" +

"SELECT books.title, iss.book\_date, iss.return\_date, iss.status " +

"FROM issuance as iss " +

"join books on iss.book\_id = books.id " +

"where reader\_id=@id;";

DB.openConnection();

NpgsqlCommand command = new NpgsqlCommand(query, DB.getConnection());

command.Parameters.AddWithValue("@id", User.ID);

NpgsqlDataReader reader = command.ExecuteReader();

while (reader.Read())

{

FillGridRow(MainGridView, reader);

}

DB.closeConnection();

MainGridView.Sort(MainGridView.Columns["book\_date"], ListSortDirection.Ascending);

foreach (DataGridViewColumn column in MainGridView.Columns)

{

column.AutoSizeMode = DataGridViewAutoSizeColumnMode.AllCells;

column.DefaultCellStyle.WrapMode = DataGridViewTriState.True;

}

MainGridView.Columns[3].AutoSizeMode = DataGridViewAutoSizeColumnMode.Fill;

}

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

{

isClosedByCode = true;

Navigation.CloseUI();

if (User.Role == "admin") { Navigation.CloseAdmin(); }

Navigation.MainForm.Show();

Navigation.MainForm.Location = this.Location;

}

private void exitLable\_MouseEnter(object sender, EventArgs e)

{

exitLable.ForeColor = Color.Black;

}

private void exitLable\_MouseLeave(object sender, EventArgs e)

{

exitLable.ForeColor = Color.DimGray;

}

private void UserForm\_FormClosing(object sender, FormClosingEventArgs e)

{

if (!this.isClosedByCode)

{

Application.Exit();

}

}

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

{

this.Hide();

Navigation.UserForm.Show();

Navigation.UserForm.Location = this.Location;

}

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

{

this.Hide();

Navigation.BooksForm.Show();

Navigation.BooksForm.Location = this.Location;

}

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

{

this.Hide();

Navigation.UserStoryForm.Show();

Navigation.UserStoryForm.Location = this.Location;

}

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

{

this.Hide();

Navigation.UsersAdminForm.Show();

Navigation.UsersAdminForm.Location = this.Location;

}

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

{

this.Hide();

Navigation.BooksAdminForm.Show();

Navigation.BooksAdminForm.Location = this.Location;

}

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

{

this.Hide();

Navigation.IssuanceAdminForm.Show();

Navigation.IssuanceAdminForm.Location = this.Location;

}

}

}

Листинг Б.10 – UsersAdminForm.cs (список пользователей)

using Npgsql;

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace LibrarySQLApp

{

public partial class UsersAdminForm : Form

{

User User { get; set; }

public bool isClosedByCode = false;

public UsersAdminForm(User user)

{

this.User = user;

InitializeComponent();

nameLabel.Text = User.FullName;

roleLabel.Text = User.Role;

CreateGridView();

LoadGridView();

}

private void CreateGridView()

{

MainGridView.Columns.Add("login", "Логин");

MainGridView.Columns.Add("name", "Имя");

MainGridView.Columns.Add("role", "Роль");

MainGridView.Columns.Add("group\_code", "Группа");

MainGridView.Columns.Add("email", "Почта");

MainGridView.Columns.Add("phone", "Телефон");

}

private void FillGridRow(DataGridView dgv, IDataRecord record)

{

dgv.Rows.Add(

record.GetString(0),

record.GetString(1) + " " + record.GetString(2),

record.GetString(3),

record.IsDBNull(4) ? "-" : record.GetString(4),

record.IsDBNull(5) ? "-" : record.GetString(5),

record.IsDBNull(6) ? "-" : record.GetString(6));

}

public void LoadGridView()

{

MainGridView.Rows.Clear();

DataTable dataTable = new DataTable();

string query = "" +

"select login, first\_name, last\_name, role, group\_code, email, phone " +

"from accounts join readers on " +

"accounts.reader\_id = readers.id;";

DB.openConnection();

NpgsqlCommand command = new NpgsqlCommand(query, DB.getConnection());

NpgsqlDataReader reader = command.ExecuteReader();

while (reader.Read())

{

FillGridRow(MainGridView, reader);

}

DB.closeConnection();

MainGridView.Sort(MainGridView.Columns["name"], ListSortDirection.Ascending);

foreach (DataGridViewColumn column in MainGridView.Columns)

{

column.AutoSizeMode = DataGridViewAutoSizeColumnMode.AllCells;

column.DefaultCellStyle.WrapMode = DataGridViewTriState.True;

}

MainGridView.Columns[5].AutoSizeMode = DataGridViewAutoSizeColumnMode.Fill;

}

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

{

isClosedByCode = true;

Navigation.CloseUI();

if (User.Role == "admin") Navigation.CloseAdmin();

Navigation.MainForm.Show();

Navigation.MainForm.Location = this.Location;

}

private void exitLable\_MouseEnter(object sender, EventArgs e)

{

exitLable.ForeColor = Color.Black;

}

private void exitLable\_MouseLeave(object sender, EventArgs e)

{

exitLable.ForeColor = Color.DimGray;

}

private void UserForm\_FormClosing(object sender, FormClosingEventArgs e)

{

if (!this.isClosedByCode)

{

Application.Exit();

}

}

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

{

this.Hide();

Navigation.UserForm.Show();

Navigation.UserForm.Location = this.Location;

}

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

{

this.Hide();

Navigation.BooksForm.Show();

Navigation.BooksForm.Location = this.Location;

}

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

{

this.Hide();

Navigation.UserStoryForm.Show();

Navigation.UserStoryForm.Location = this.Location;

}

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

{

this.Hide();

Navigation.UsersAdminForm.Show();

Navigation.UsersAdminForm.Location = this.Location;

}

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

{

this.Hide();

Navigation.BooksAdminForm.Show();

Navigation.BooksAdminForm.Location = this.Location;

}

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

{

this.Hide();

Navigation.IssuanceAdminForm.Show();

Navigation.IssuanceAdminForm.Location = this.Location;

}

private void SearchBox\_TextChanged(object sender, EventArgs e)

{

if (SearchBox.Text.Length > 0)

{

MainGridView.Rows.Clear();

DataTable dataTable = new DataTable();

string query = $"SELECT login, first\_name, last\_name, role, group\_code, email, phone " +

"FROM accounts JOIN readers " +

"ON accounts.reader\_id = readers.id " +

"WHERE accounts.login LIKE '%" + SearchBox.Text + "%' " +

"OR accounts.role LIKE '%" + SearchBox.Text + "%' " +

"OR readers.first\_name LIKE '%" + SearchBox.Text + "%' " +

"OR readers.last\_name LIKE '%" + SearchBox.Text + "%' " +

"OR readers.group\_code LIKE '%" + SearchBox.Text + "%' " +

"OR readers.email LIKE '%" + SearchBox.Text + "%' " +

"OR readers.phone LIKE '%" + SearchBox.Text + "%'";

DB.openConnection();

NpgsqlCommand command = new NpgsqlCommand(query, DB.getConnection());

NpgsqlDataReader reader = command.ExecuteReader();

command.Parameters.AddWithValue("@text", SearchBox.Text);

while (reader.Read())

{

FillGridRow(MainGridView, reader);

}

DB.closeConnection();

MainGridView.Sort(MainGridView.Columns["name"], ListSortDirection.Ascending);

foreach (DataGridViewColumn column in MainGridView.Columns)

{

column.AutoSizeMode = DataGridViewAutoSizeColumnMode.AllCells;

column.DefaultCellStyle.WrapMode = DataGridViewTriState.True;

}

MainGridView.Columns[5].AutoSizeMode = DataGridViewAutoSizeColumnMode.Fill;

}

else

{

LoadGridView();

}

}

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

{

AddUserAdminForm AddUserForm = new AddUserAdminForm(User);

this.Hide();

AddUserForm.Show();

AddUserForm.Location = this.Location;

}

}

}

Листинг Б.11 – AddUserAdminForm.cs (добавление пользователя)

using Npgsql;

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace LibrarySQLApp

{

public partial class AddUserAdminForm : Form

{

User User { get; set; }

public bool isClosedByCode = false;

public AddUserAdminForm(User user)

{

this.User = user;

InitializeComponent();

nameLabel.Text = User.FullName;

roleLabel.Text = User.Role;

}

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

{

isClosedByCode = true;

this.Close();

Navigation.CloseUI();

if (User.Role == "admin") Navigation.CloseAdmin();

Navigation.MainForm.Show();

Navigation.MainForm.Location = this.Location;

}

private void exitLable\_MouseEnter(object sender, EventArgs e)

{

exitLable.ForeColor = Color.Black;

}

private void exitLable\_MouseLeave(object sender, EventArgs e)

{

exitLable.ForeColor = Color.DimGray;

}

private void UserForm\_FormClosing(object sender, FormClosingEventArgs e)

{

if (!this.isClosedByCode)

{

Application.Exit();

}

}

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

{

String login = loginTextBox.Text;

String pass = passTextBox.Text;

String first\_name = fnameTextBox.Text;

String last\_name = lnameTextBox.Text;

String role = roleComboBox.Text;

String group = groupTextBox.Text;

if (string.IsNullOrWhiteSpace(login) ||

string.IsNullOrWhiteSpace(pass) ||

string.IsNullOrWhiteSpace(first\_name) ||

string.IsNullOrWhiteSpace(last\_name) ||

string.IsNullOrWhiteSpace(role))

{

Messages.DisplayErrorMessage("Заполните все поля");

return;

}

String readersQuery =

"insert into readers (first\_name, last\_name, group\_code, email, phone) " +

"values (@first\_name, @last\_name, @group, null, null);";

String accountsQuery =

"insert into accounts (login, password, role, reader\_id) " +

"values (@login, @pass, @role, @id);";

NpgsqlTransaction transaction = null;

try

{

DB.openConnection();

transaction = DB.getConnection().BeginTransaction();

NpgsqlCommand command = new NpgsqlCommand();

command.Connection = DB.getConnection();

command.Transaction = transaction;

command.CommandText = readersQuery;

command.Parameters.AddWithValue("@first\_name", first\_name);

command.Parameters.AddWithValue("@last\_name", last\_name);

if (string.IsNullOrEmpty(group))

{

command.Parameters.AddWithValue("@group", DBNull.Value);

}

else

{

command.Parameters.AddWithValue("@group", group);

}

command.ExecuteNonQuery();

command.CommandText = "select lastval();";

int newUserID = Convert.ToInt32(command.ExecuteScalar());

command.CommandText = accountsQuery;

command.Parameters.AddWithValue("@login", login);

command.Parameters.AddWithValue("@pass", pass);

command.Parameters.AddWithValue("@role", role);

command.Parameters.AddWithValue("@id", newUserID);

command.ExecuteNonQuery();

transaction.Commit();

Messages.DisplayInfoMessage("Пользователь добавлен");

Navigation.UsersAdminForm.LoadGridView();

loginTextBox.Text = "";

passTextBox.Text = "";

fnameTextBox.Text = "";

lnameTextBox.Text = "";

groupTextBox.Text = "";

}

catch (Exception ex)

{

if (transaction != null)

{

transaction.Rollback();

}

Messages.DisplayErrorMessage($"Ошибка: {ex.Message}");

}

finally

{

if (transaction != null)

{

transaction.Dispose();

}

DB.closeConnection();

}

}

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

{

String login = loginTextBox.Text;

String pass = passTextBox.Text;

String first\_name = fnameTextBox.Text;

String last\_name = lnameTextBox.Text;

String group = groupTextBox.Text;

if (!string.IsNullOrWhiteSpace(login) ||

!string.IsNullOrWhiteSpace(pass) ||

!string.IsNullOrWhiteSpace(first\_name) ||

!string.IsNullOrWhiteSpace(last\_name) ||

!string.IsNullOrWhiteSpace(group))

{

DialogResult result = Messages.DisplayQuestionMessage("При выходе данные не сохранятся. Продолжить?");

if (result == DialogResult.No)

{

return;

}

}

isClosedByCode = true;

this.Close();

Navigation.UsersAdminForm.Show();

Navigation.UsersAdminForm.Location = this.Location;

}

private void getBack\_MouseEnter(object sender, EventArgs e)

{

getBack.BackColor = Color.LightGray;

}

private void getBack\_MouseLeave(object sender, EventArgs e)

{

getBack.BackColor = SystemColors.Control;

}

}

}

Листинг Б.12 – BooksAdminForm.cs (список книг)

using Npgsql;

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace LibrarySQLApp

{

public partial class BooksAdminForm : Form

{

User User { get; set; }

public bool isClosedByCode = false;

public BooksAdminForm(User user)

{

this.User = user;

InitializeComponent();

nameLabel.Text = User.FullName;

roleLabel.Text = User.Role;

CreateGridView();

LoadGridView();

}

private void CreateGridView()

{

MainGridView.Columns.Add("title", "Название");

MainGridView.Columns.Add("publisher", "Издатель");

MainGridView.Columns.Add("year", "Год");

MainGridView.Columns.Add("type", "Тип");

MainGridView.Columns.Add("amount", "Всего");

MainGridView.Columns.Add("issued", "Выдано");

MainGridView.Columns.Add("id", String.Empty);

MainGridView.Columns["id"].Visible = false;

}

private void FillGridRow(DataGridView dgv, IDataRecord record)

{

dgv.Rows.Add(

record.GetString(0),

record.GetString(1),

record.GetInt32(2),

record.GetString(3),

record.GetInt32(4),

record.GetInt32(5),

record.GetInt32(6));

}

public void LoadGridView()

{

MainGridView.Rows.Clear();

DataTable dataTable = new DataTable();

string query =

"select books.title, publisher, book\_year, type, amount, issued\_amount, books.id " +

"from books join books\_amount as ba on ba.id = books.id;";

DB.openConnection();

NpgsqlCommand command = new NpgsqlCommand(query, DB.getConnection());

NpgsqlDataReader reader = command.ExecuteReader();

while (reader.Read())

{

FillGridRow(MainGridView, reader);

}

DB.closeConnection();

MainGridView.Sort(MainGridView.Columns["title"], ListSortDirection.Ascending);

foreach (DataGridViewColumn column in MainGridView.Columns)

{

column.AutoSizeMode = DataGridViewAutoSizeColumnMode.AllCells;

column.DefaultCellStyle.WrapMode = DataGridViewTriState.True;

}

MainGridView.Columns[5].AutoSizeMode = DataGridViewAutoSizeColumnMode.Fill;

PrintAuthors();

}

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

{

isClosedByCode = true;

Navigation.CloseUI();

if (User.Role == "admin") Navigation.CloseAdmin();

Navigation.MainForm.Show();

Navigation.MainForm.Location = this.Location;

}

private void exitLable\_MouseEnter(object sender, EventArgs e)

{

exitLable.ForeColor = Color.Black;

}

private void exitLable\_MouseLeave(object sender, EventArgs e)

{

exitLable.ForeColor = Color.DimGray;

}

private void UserForm\_FormClosing(object sender, FormClosingEventArgs e)

{

if (!this.isClosedByCode)

{

Application.Exit();

}

}

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

{

this.Hide();

Navigation.UserForm.Show();

Navigation.UserForm.Location = this.Location;

}

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

{

this.Hide();

Navigation.BooksForm.Show();

Navigation.BooksForm.Location = this.Location;

}

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

{

this.Hide();

Navigation.UserStoryForm.Show();

Navigation.UserStoryForm.Location = this.Location;

}

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

{

this.Hide();

Navigation.UsersAdminForm.Show();

Navigation.UsersAdminForm.Location = this.Location;

}

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

{

this.Hide();

Navigation.BooksAdminForm.Show();

Navigation.BooksAdminForm.Location = this.Location;

}

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

{

this.Hide();

Navigation.IssuanceAdminForm.Show();

Navigation.IssuanceAdminForm.Location = this.Location;

}

private void MainGridView\_CellClick(object sender, DataGridViewCellEventArgs e)

{

PrintAuthors();

}

public void PrintAuthors()

{

DataGridViewRow row;

if (MainGridView.CurrentCell != null)

{

row = MainGridView.Rows[MainGridView.CurrentCell.RowIndex];

}

else

{

row = MainGridView.Rows[0];

}

string query = "" +

"select first\_name, last\_name from authors " +

"join book\_author as ba on ba.author\_id = authors.id " +

"join books on books.id = ba.book\_id " +

"where books.id = @id";

List<string> authors = new List<string>();

DB.openConnection();

NpgsqlCommand command = new NpgsqlCommand(query, DB.getConnection());

command.Parameters.AddWithValue("@id", (int)row.Cells[6].Value);

NpgsqlDataReader reader = command.ExecuteReader();

while (reader.Read())

{

authors.Add(reader.GetString(0) + " " + reader.GetString(1));

}

DB.closeConnection();

authorsList.Text = string.Join(", ", authors);

}

private void SearchBox\_TextChanged(object sender, EventArgs e)

{

if (SearchBox.Text.Length <= 0)

{

LoadGridView();

return;

}

MainGridView.Rows.Clear();

DataTable dataTable = new DataTable();

string query =

"select distinct b.title, publisher, book\_year, " +

"type, amount, issued\_amount, b.id " +

"from books as b " +

"join book\_author as ban on b.id = ban.book\_id " +

"join authors as a on ban.author\_id = a.id " +

"join books\_amount as bam on b.id = bam.id " +

"where b.title like '%" + SearchBox.Text + "%' or " +

"b.publisher like '%" + SearchBox.Text + "%' or " +

"a.first\_name like '%" + SearchBox.Text + "%' or " +

"a.last\_name like '%" + SearchBox.Text + "%';";

DB.openConnection();

NpgsqlCommand command = new NpgsqlCommand(query, DB.getConnection());

NpgsqlDataReader reader = command.ExecuteReader();

command.Parameters.AddWithValue("@text", SearchBox.Text);

while (reader.Read())

{

FillGridRow(MainGridView, reader);

}

DB.closeConnection();

MainGridView.Sort(MainGridView.Columns["title"], ListSortDirection.Ascending);

foreach (DataGridViewColumn column in MainGridView.Columns)

{

column.AutoSizeMode = DataGridViewAutoSizeColumnMode.AllCells;

column.DefaultCellStyle.WrapMode = DataGridViewTriState.True;

}

MainGridView.Columns[5].AutoSizeMode = DataGridViewAutoSizeColumnMode.Fill;

if (MainGridView.CurrentCell != null)

{

PrintAuthors();

return;

}

authorsList.Text = "выберите книгу";

}

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

{

this.Hide();

AddBookForm addBookForm = new AddBookForm(User);

addBookForm.Show();

addBookForm.Location = this.Location;

}

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

{

if (MainGridView.CurrentCell == null)

{

Messages.DisplayErrorMessage("Выберите книгу для изменения");

return;

}

DataGridViewRow row = MainGridView.Rows[MainGridView.CurrentCell.RowIndex];

this.Hide();

AlterBookForm AlterBookForm = new AlterBookForm(User, row);

AlterBookForm.Show();

AlterBookForm.Location = this.Location;

}

private void BooksAdminForm\_Activated(object sender, EventArgs e)

{

LoadGridView();

}

}

}

Листинг Б.13 – AlterBookForm.cs (редактирование книги)

using Npgsql;

using System;

using System.Collections;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Security.Policy;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace LibrarySQLApp

{

public partial class AlterBookForm : Form

{

User User { get; set; }

DataGridViewRow Row { get; set; }

public bool isClosedByCode = false;

public AlterBookForm(User user, DataGridViewRow row)

{

this.User = user;

this.Row = row;

InitializeComponent();

nameLabel.Text = User.FullName;

roleLabel.Text = User.Role;

FillBookData();

PrintAuthors();

}

public void FillBookData()

{

titleBox.Text = Row.Cells[0].Value.ToString();

publisherBox.Text = Row.Cells[1].Value.ToString();

yearBox.Text = Row.Cells[2].Value.ToString();

typeBox.Text = Row.Cells[3].Value.ToString();

amountBox.Text = Row.Cells[4].Value.ToString();

}

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

{

isClosedByCode = true;

this.Close();

Navigation.CloseUI();

if (User.Role == "admin") Navigation.CloseAdmin();

Navigation.MainForm.Show();

Navigation.MainForm.Location = this.Location;

}

private void exitLable\_MouseEnter(object sender, EventArgs e)

{

exitLable.ForeColor = Color.Black;

}

private void exitLable\_MouseLeave(object sender, EventArgs e)

{

exitLable.ForeColor = Color.DimGray;

}

private void UserForm\_FormClosing(object sender, FormClosingEventArgs e)

{

if (!this.isClosedByCode)

{

Application.Exit();

}

}

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

{

try

{

String title = titleBox.Text;

String publisher = publisherBox.Text;

int year = int.Parse(yearBox.Text);

int amount = int.Parse(amountBox.Text);

String type = typeBox.Text;

if (string.IsNullOrWhiteSpace(title) ||

string.IsNullOrWhiteSpace(publisher) ||

string.IsNullOrWhiteSpace(type))

{

Messages.DisplayErrorMessage("Заполните все поля");

return;

}

String query =

"update books set " +

"title = @title, " +

"publisher = @publisher, " +

"book\_year = @year, " +

"amount = @amount, " +

"type = @type " +

"where id = @id";

DB.openConnection();

NpgsqlCommand command = new NpgsqlCommand();

command.Connection = DB.getConnection();

command.CommandText = query;

command.Parameters.AddWithValue("@title", title);

command.Parameters.AddWithValue("@publisher", publisher);

command.Parameters.AddWithValue("@amount", amount);

command.Parameters.AddWithValue("@year", year);

command.Parameters.AddWithValue("@type", type);

command.Parameters.AddWithValue("@id", (int)Row.Cells[6].Value);

command.ExecuteNonQuery();

Messages.DisplayInfoMessage("Данные обновлены");

Navigation.BooksAdminForm.LoadGridView();

}

catch (Exception ex)

{

Messages.DisplayErrorMessage($"Ошибка: {ex.Message}");

}

finally

{

DB.closeConnection();

}

}

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

{

DialogResult result = Messages.DisplayQuestionMessage(

"Возможно, при выходе некоторые данные не сохранятся. Продолжить?");

if (result == DialogResult.No)

{

return;

}

isClosedByCode = true;

this.Close();

Navigation.BooksAdminForm.Show();

Navigation.BooksAdminForm.Location = this.Location;

}

private void getBack\_MouseEnter(object sender, EventArgs e)

{

getBack.BackColor = Color.LightGray;

}

private void getBack\_MouseLeave(object sender, EventArgs e)

{

getBack.BackColor = SystemColors.Control;

}

public void PrintAuthors()

{

string query = "" +

"select first\_name, last\_name from authors " +

"join book\_author as ba on ba.author\_id = authors.id " +

"join books on books.id = ba.book\_id " +

"where books.id = @id";

List<string> authors = new List<string>();

DB.openConnection();

NpgsqlCommand command = new NpgsqlCommand(query, DB.getConnection());

command.Parameters.AddWithValue("@id", (int)Row.Cells[6].Value);

NpgsqlDataReader reader = command.ExecuteReader();

while (reader.Read())

{

authors.Add(reader.GetString(0)[0] + ". " + reader.GetString(1));

}

DB.closeConnection();

authorsList.Text = string.Join(", ", authors);

}

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

{

String first\_name = firstNameBox.Text;

String last\_name = lastNameBox.Text;

String authorsQuery =

"insert into authors (first\_name, last\_name) " +

"select @fname, @lname where not exists " +

"(select 1 from authors " +

"where first\_name = @fname and " +

"last\_name = @lname);";

String searchQuery =

"select id from authors " +

"where first\_name = @fname and " +

"last\_name = @lname";

String book\_authorQuery =

"insert into book\_author " +

"values (@b\_id, @a\_id)";

if (string.IsNullOrWhiteSpace(first\_name) ||

string.IsNullOrWhiteSpace(last\_name))

{

Messages.DisplayErrorMessage("Заполните все поля");

return;

}

try

{

DB.openConnection();

NpgsqlCommand command = new NpgsqlCommand();

command.Connection = DB.getConnection();

command.CommandText = authorsQuery;

command.Parameters.AddWithValue("@fname", first\_name);

command.Parameters.AddWithValue("@lname", last\_name);

command.ExecuteNonQuery();

command.CommandText = searchQuery;

command.Parameters.AddWithValue("@fname", first\_name);

command.Parameters.AddWithValue("@lname", last\_name);

int newAuthorID = Convert.ToInt32(command.ExecuteScalar());

command.CommandText = book\_authorQuery;

command.Parameters.AddWithValue("@b\_id", (int)Row.Cells[6].Value);

command.Parameters.AddWithValue("@a\_id", newAuthorID);

command.ExecuteNonQuery();

PrintAuthors();

Navigation.BooksAdminForm.PrintAuthors();

firstNameBox.Text = string.Empty;

lastNameBox.Text = string.Empty;

Messages.DisplayInfoMessage("Автор добавлен");

}

catch (Exception ex)

{

Messages.DisplayErrorMessage($"Ошибка: {ex.Message}");

}

finally

{

DB.closeConnection();

}

}

}

}

Листинг Б.14 – AddBookForm.cs (добавление книги)

using Npgsql;

using System;

using System.Collections;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Security.Policy;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace LibrarySQLApp

{

public partial class AddBookForm : Form

{

User User { get; set; }

public bool isClosedByCode = false;

public AddBookForm(User user)

{

this.User = user;

InitializeComponent();

nameLabel.Text = User.FullName;

roleLabel.Text = User.Role;

}

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

{

isClosedByCode = true;

this.Close();

Navigation.CloseUI();

if (User.Role == "admin") Navigation.CloseAdmin();

Navigation.MainForm.Show();

Navigation.MainForm.Location = this.Location;

}

private void exitLable\_MouseEnter(object sender, EventArgs e)

{

exitLable.ForeColor = Color.Black;

}

private void exitLable\_MouseLeave(object sender, EventArgs e)

{

exitLable.ForeColor = Color.DimGray;

}

private void UserForm\_FormClosing(object sender, FormClosingEventArgs e)

{

if (!this.isClosedByCode)

{

Application.Exit();

}

}

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

{

try

{

String title = titleBox.Text;

String publisher = publisherBox.Text;

int year = int.Parse(yearBox.Text);

int amount = int.Parse(amountBox.Text);

String type = typeBox.Text;

if (string.IsNullOrWhiteSpace(title) ||

string.IsNullOrWhiteSpace(publisher) ||

string.IsNullOrWhiteSpace(type))

{

Messages.DisplayErrorMessage("Заполните все поля");

return;

}

String query =

"insert into books " +

"(title, publisher, book\_year, amount, type) " +

"values (@title, @publisher, @year, @amount, @type)";

DB.openConnection();

NpgsqlCommand command = new NpgsqlCommand();

command.Connection = DB.getConnection();

command.CommandText = query;

command.Parameters.AddWithValue("@title", title);

command.Parameters.AddWithValue("@publisher", publisher);

command.Parameters.AddWithValue("@amount", amount);

command.Parameters.AddWithValue("@year", year);

command.Parameters.AddWithValue("@type", type);

command.ExecuteNonQuery();

Messages.DisplayInfoMessage("Книга добавлена");

Navigation.BooksAdminForm.LoadGridView();

titleBox.Text = string.Empty;

publisherBox.Text = string.Empty;

amountBox.Text = string.Empty;

typeBox.Text = string.Empty;

yearBox.Text = string.Empty;

}

catch (Exception ex)

{

Messages.DisplayErrorMessage($"Ошибка: {ex.Message}");

}

finally

{

DB.closeConnection();

}

}

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

{

DialogResult result = Messages.DisplayQuestionMessage(

"Возможно, при выходе некоторые данные не сохранятся. Продолжить?");

if (result == DialogResult.No) { return; }

isClosedByCode = true;

this.Close();

Navigation.BooksAdminForm.Show();

Navigation.BooksAdminForm.Location = this.Location;

}

private void getBack\_MouseEnter(object sender, EventArgs e)

{

getBack.BackColor = Color.LightGray;

}

private void getBack\_MouseLeave(object sender, EventArgs e)

{

getBack.BackColor = SystemColors.Control;

}

}

}

Листинг Б.15 – IssuanceAdminForm.cs (история выдач)

using Npgsql;

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using static System.Windows.Forms.VisualStyles.VisualStyleElement;

namespace LibrarySQLApp

{

public partial class IssuanceAdminForm : Form

{

User User { get; set; }

public bool isClosedByCode = false;

public IssuanceAdminForm(User user)

{

this.User = user;

InitializeComponent();

nameLabel.Text = User.FullName;

roleLabel.Text = User.Role;

if (User.Role != "admin")

{

adminPanel.Hide();

}

CreateGridView();

LoadGridView();

}

private void CreateGridView()

{

MainGridView.Columns.Add("title", "Книга");

MainGridView.Columns.Add("reader", "Читатель");

MainGridView.Columns.Add("book\_date", "Дата выдачи");

MainGridView.Columns.Add("return\_date", "Дата возврата");

MainGridView.Columns.Add("status", "Состояние");

MainGridView.Columns.Add("fine", "Штраф");

MainGridView.Columns.Add("id", "ID");

MainGridView.Columns["id"].Visible = false;

}

private void FillGridRow(DataGridView dgv, IDataRecord record)

{

dgv.Rows.Add(

record.GetString(0),

record.GetString(1) + " " + record.GetString(2),

record.IsDBNull(3) ? "-" : record.GetDateTime(3).Date.ToString("d"),

record.IsDBNull(4) ? "-": record.GetDateTime(4).Date.ToString("d"),

record.GetString(5),

record.GetBoolean(6) ? "✓" : "-",

record.GetInt32(7));

}

public void LoadGridView()

{

MainGridView.Rows.Clear();

DataTable dataTable = new DataTable();

string query = "" +

"SELECT title, first\_name, last\_name, iss.book\_date, " +

"iss.return\_date, iss.status, fined, iss.id " +

"FROM issuance as iss " +

"join books on iss.book\_id = books.id " +

"join readers on iss.reader\_id = readers.id";

DB.openConnection();

NpgsqlCommand command = new NpgsqlCommand(query, DB.getConnection());

NpgsqlDataReader reader = command.ExecuteReader();

while (reader.Read())

{

FillGridRow(MainGridView, reader);

}

DB.closeConnection();

MainGridView.Sort(MainGridView.Columns["book\_date"], ListSortDirection.Descending);

}

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

{

isClosedByCode = true;

Navigation.CloseUI();

if (User.Role == "admin") { Navigation.CloseAdmin(); }

Navigation.MainForm.Show();

Navigation.MainForm.Location = this.Location;

}

private void exitLable\_MouseEnter(object sender, EventArgs e)

{

exitLable.ForeColor = Color.Black;

}

private void exitLable\_MouseLeave(object sender, EventArgs e)

{

exitLable.ForeColor = Color.DimGray;

}

private void UserForm\_FormClosing(object sender, FormClosingEventArgs e)

{

if (!this.isClosedByCode)

{

Application.Exit();

}

}

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

{

this.Hide();

Navigation.UserForm.Show();

Navigation.UserForm.Location = this.Location;

}

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

{

this.Hide();

Navigation.BooksForm.Show();

Navigation.BooksForm.Location = this.Location;

}

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

{

this.Hide();

Navigation.UserStoryForm.Show();

Navigation.UserStoryForm.Location = this.Location;

}

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

{

this.Hide();

Navigation.UsersAdminForm.Show();

Navigation.UsersAdminForm.Location = this.Location;

}

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

{

this.Hide();

Navigation.BooksAdminForm.Show();

Navigation.BooksAdminForm.Location = this.Location;

}

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

{

this.Hide();

Navigation.IssuanceAdminForm.Show();

Navigation.IssuanceAdminForm.Location = this.Location;

}

private void SearchBox\_TextChanged(object sender, EventArgs e)

{

if (SearchBox.Text.Length <= 0)

{

LoadGridView();

return;

}

MainGridView.Rows.Clear();

DataTable dataTable = new DataTable();

string query =

"select b.title, r.first\_name, r.last\_name, " +

"iss.book\_date, iss.return\_date, iss.status, r.fined, iss.id " +

"from books as b " +

"join issuance as iss on b.id = iss.book\_id " +

"join readers as r on iss.reader\_id = r.id " +

"where b.title like '%" + SearchBox.Text + "%' or " +

"r.first\_name like '%" + SearchBox.Text + "%' or " +

"iss.status like '%" + SearchBox.Text + "%' or " +

"r.last\_name like '%" + SearchBox.Text + "%';";

DB.openConnection();

NpgsqlCommand command = new NpgsqlCommand(query, DB.getConnection());

NpgsqlDataReader reader = command.ExecuteReader();

while (reader.Read())

{

FillGridRow(MainGridView, reader);

}

DB.closeConnection();

MainGridView.Sort(MainGridView.Columns["book\_date"], ListSortDirection.Descending);

}

private void SetButton(System.Windows.Forms.Button button, Boolean IsEnabled)

{

if (IsEnabled)

{

button.Enabled = true;

button.BackColor = SystemColors.Highlight;

}

else

{

button.Enabled = false;

button.BackColor = Color.LightGray;

}

}

private void UpdateButtons()

{

SetButton(rejectButton, false);

SetButton(returnButton, false);

SetButton(fineButton, false);

SetButton(issueButton, false);

if (MainGridView.CurrentCell == null)

{

return;

}

DataGridViewRow row = MainGridView.Rows[MainGridView.CurrentCell.RowIndex];

String status = row.Cells[4].Value.ToString();

if (status == "pending")

{

SetButton(rejectButton, true);

SetButton(issueButton, true);

}

else if (status == "issued")

{

SetButton(returnButton, true);

SetButton(fineButton, true);

}

return;

}

private void MainGridView\_CellContentClick(object sender, DataGridViewCellEventArgs e)

{

UpdateButtons();

}

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

{

NpgsqlTransaction transaction = null;

try

{

DataGridViewRow row = MainGridView.Rows[MainGridView.CurrentCell.RowIndex];

int id = (int)row.Cells[6].Value;

string issueQuery = "" +

"update issuance " +

"set status = 'returned', " +

"return\_date = CURRENT\_DATE " +

"where id = @id";

string readerQuery = "" +

"update readers " +

"set fined = false " +

"WHERE id = " +

"(select reader\_id from issuance " +

"where id = @id); ";

DB.openConnection();

NpgsqlCommand command1 = new NpgsqlCommand(issueQuery, DB.getConnection());

transaction = DB.getConnection().BeginTransaction(); ;

command1.Parameters.AddWithValue("@id", id);

command1.ExecuteNonQuery();

NpgsqlCommand command = new NpgsqlCommand(readerQuery, DB.getConnection());

command.Parameters.AddWithValue("@id", id);

command.ExecuteNonQuery();

transaction.Commit();

DB.closeConnection();

LoadGridView();

UpdateButtons();

}

catch (Exception ex)

{

transaction.Rollback();

Messages.DisplayErrorMessage(ex.Message);

}

transaction.Dispose();

}

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

{

DataGridViewRow row = MainGridView.Rows[MainGridView.CurrentCell.RowIndex];

int id = (int)row.Cells[6].Value;

string query = "" +

"update issuance " +

"set status = 'rejected' " +

"where id = @id";

DB.openConnection();

NpgsqlCommand command = new NpgsqlCommand(query, DB.getConnection());

command.Parameters.AddWithValue("@id", id);

command.ExecuteNonQuery();

DB.closeConnection();

LoadGridView();

UpdateButtons();

}

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

{

DataGridViewRow row = MainGridView.Rows[MainGridView.CurrentCell.RowIndex];

int id = (int)row.Cells[6].Value;

string query = "" +

"update issuance " +

"set status = 'issued', " +

"book\_date = CURRENT\_DATE, " +

"return\_date = CURRENT\_DATE + INTERVAL '1 month' " +

"where id = @id";

DB.openConnection();

NpgsqlCommand command = new NpgsqlCommand(query, DB.getConnection());

command.Parameters.AddWithValue("@id", id);

command.ExecuteNonQuery();

DB.closeConnection();

LoadGridView();

UpdateButtons();

}

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

{

DataGridViewRow row = MainGridView.Rows[MainGridView.CurrentCell.RowIndex];

int id = (int)row.Cells[6].Value;

DateTime endDate = DateTime.Parse(row.Cells[3].Value.ToString());

DateTime currentDate = DateTime.Now;

if (currentDate < endDate)

{

Messages.DisplayErrorMessage("Срок выдачи ещё не истёк!");

return;

}

string readerQuery = "" +

"update readers " +

"set fined = true " +

"WHERE id = " +

"(select reader\_id from issuance " +

"where id = @id); ";

DB.openConnection();

NpgsqlCommand command = new NpgsqlCommand(readerQuery, DB.getConnection());

command.Parameters.AddWithValue("@id", id);

command.ExecuteNonQuery();

DB.closeConnection();

LoadGridView();

UpdateButtons();

}

private void IssuanceAdminForm\_Activated(object sender, EventArgs e)

{

LoadGridView();

}

}

}