Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

ЛЕСОСИБИРСКИЙ ПЕДАГОГИЧЕСКИЙ ИНСТИТУТ –
филиал Сибирского федерального университета

Высшей математики, информатики, экономики и естествознания
кафедра

УТВЕРЖДАЮ
Заведующий кафедрой
Л.Н. Храмова
подпись инициалы, фамилия
«14» 06 2024 г.

БАКАЛАВРСКАЯ РАБОТА

09.03.02 Информационные системы и технологии
код-наименование направления

РАЗРАБОТКА ПРИЛОЖЕНИЯ НА PYTHON ДЛЯ АНАЛИЗА И УПРАВЛЕНИЯ БОЛЬШИМИ ДАННЫМИ В БИБЛИОТЕКЕ

Руководитель 14.06.24 профессор, д-р техн. наук А.П. Мохирев
подпись, дата должность, ученая степень инициалы, фамилия

Выпускник 14.06.2024 г. А.Е. Вольхин
подпись, дата инициалы, фамилия

Нормоконтролер 14.06.2024 г. А.В. Фирер
подпись, дата инициалы, фамилия

Лесосибирск 2024

РЕФЕРАТ

Выпускная квалификационная работа по теме «Разработка приложения на Python для анализа и управления большими данными в библиотеке» содержит 58 страниц текстового документа, 16 иллюстраций, 4 таблицы, 40 использованных источников.

PYTHON, БОЛЬШИЕ ДАННЫЕ, АНАЛИЗ БОЛЬШИХ ДАННЫХ, УПРАВЛЕНИЕ БОЛЬШИМИ ДАННЫМИ

Цель исследования – теоретически обосновать и разработать приложение на Python для анализа и управления большими данными в библиотеке.

Объект исследования – приложения для управления большими данными.

Предмет исследования – процесс проектирования и создания приложения для управления большими данными на Python.

Для реализации поставленной цели необходимо выполнить следующие задачи:

– на основе анализа учебной и технической литературы раскрыть теоретические аспекты, описывающие инструментальные средства разработки приложений для управления и анализа данных, их характеристики, языки разметки и области их применения;

– описать основные функциональные и нефункциональные требования к приложению, определить структуру базы данных, разработать макет приложения;

– разработать приложение для анализа и управления данными.

Во время разработки приложения для управления и анализа больших данных в библиотеке изучены основы языка программирования Python, разметки для сторонних библиотек: tkinker, pandas, matpotlib.

В результате выполнения выпускной квалификационной работы разработано приложение для анализа и управления большими данными в библиотеке с использованием среды программирования Python.

СОДЕРЖАНИЕ

Введение 4
1 Теоретические основы создания приложений для анализа и управления большими данными 6
1.1 Язык программирования для разработки приложения 6
1.2 Инструментальные средства для разработки интерфейса 9
1.3 Средства для управления и анализа данных 13
1.4 Характеристики баз данных 16
2 Разработка приложения для анализа и управления большими данными в библиотеке 20
2.1 Функциональные и нефункциональные требования 20
2.2 Разработка хранилища данных 21
2.3 Разработка графического интерфейса 27
2.4 Разработка функционала приложения 32
Заключение 44
Список использованных источников 46
Приложение А Код приложения 49

ВВЕДЕНИЕ

Создание приложений для управления большими данными остаётся актуальным в современном мире. Количество данных, создаваемых и накапливаемых компаниями и организациями, постоянно увеличивается. Данные позволяют не только облегчить рутину внутри компании, но и анализировать, визуализировать и интерпретировать данные, позволяя выявить тенденции, прогнозировать результаты и определять стратегии развития. Организации стремятся извлечь ценную информацию из своих данных, чтобы принимать более обоснованные решения. Управление такими объемами данных требует специальных инструментов и приложений.

Цель исследования – теоретически обосновать и разработать приложение на Python для анализа и управления большими данными в библиотеке.

Объект исследования – приложения для управления большими данными.

Предмет исследования – процесс проектирования и создания приложения для управления большими данными на Python.

Для реализации поставленной цели необходимо выполнить следующие задачи:

– на основе анализа учебной и технической литературы раскрыть теоретические аспекты, описывающие инструментальные средства разработки приложений для управления и анализа данных, их характеристики, языки разметки и области их применения;

– описать основные функциональные и нефункциональные требования к приложению, определить структуру базы данных, разработать макет приложения;

– разработать приложение для анализа и управления данными.

Методы исследования:

– анализ учебной и научно-технической литературы по теме исследования;

– проектирование, разработка и тестирование приложения.

Результаты исследования представлены на научных мероприятиях:

1. III Всероссийский молодежный научный форум «Современное педагогическое образование: теоретический и прикладной аспекты», ЛПИ – филиал СФУ

2. VII Всероссийская научно-практическая конференция преподавателей, учителей, студентов и молодых ученых «Актуальные проблемы преподавателей дисциплин естественнонаучного цикла», ЛПИ – филиал СФУ

1 Теоретические основы создания приложений для анализа и управления большими данными

Создание приложений для анализа и управления большими данными включает множество теоретических и практических аспектов. Сформулируем основные понятия и определения:

1. «Backend — это часть программного обеспечения, которая работает на сервере и отвечает за управление базой данных, бизнес-логику и обработку запросов от клиента (frontend). Он обеспечивает функциональность, которая не видна и не доступна напрямую пользователю, но необходима для корректной работы приложения.» [22, с. 54].

2. «Frontend — это часть веб-приложения, которая отвечает за взаимодействие с пользователем. Она включает в себя все элементы, с которыми пользователь взаимодействует, такие как кнопки, меню, формы и другие элементы интерфейса, а также визуальное оформление и пользовательский опыт (UI/UX).» [29, с. 32].

3. «Обработка данных — это процесс подготовки необработанных данных для отчетности и анализа. Он включает в себя все этапы, предшествующие анализу, включая структурирование данных, очистку, обогащение и проверку.» [2].

4. «Управление данными — это организация и хранение данных, обработанных на компьютере, организация эффективного доступа к данным и их выборки.» [12, с. 43].

1.1 Язык программирования для разработки приложения

Для разработки приложений для анализа и управления данными существует несколько языков программирования. Выбор языка зависит от конкретных требований проекта, платформы, на которой будет работать приложение.

Языки программирования, используемые для разработки таких приложений:

а) язык программирования Python:

– широко используется для анализа данных, благодаря библиотекам Pandas, NumPy, Matplotlib, Seaborn;

– поддерживает машинное обучение и искусственный интеллект через библиотеки Scikit-learn, TensorFlow, Keras и PyTorch;

– отлично подходит для веб-разработки с использованием фреймворков Django и Flask.

б) язык программирования R:

– специализированный язык для статистики и визуализации данных;

– библиотеки ggplot2, dplyr, tidyr для анализа данных.

в) язык программирования Java:

– используется для создания масштабируемых и надежных приложений;

– большое количество библиотек для работы с данными: Apache Hadoop, Apache Spark.

г) язык программирования JavaScript:

– используется для создания интерактивных веб-приложений;

– библиотеки и фреймворки: D3.js для визуализации данных, Node.js для серверной части, React, Angular, Vue.js для фронтенда.

д) язык программирования C#:

– используется для разработки приложений под платформу NET;

– подходит для создания как настольных, так и веб-приложений.

Для разработки приложения был выбран язык программирования Python.

«Python — это высокоуровневый язык программирования, отличающийся эффективностью, простотой и универсальностью использования. Он широко применяется в разработке веб-приложений и

прикладного программного обеспечения, а также в машинном обучении и обработке больших данных.» [38].

Python является мощным языком программирования с обширной экосистемой библиотек и фреймворков, что делает его популярным выбором для разработки приложений для управления данными. Python обладает богатым набором инструментов для анализа данных, таких как библиотеки Pandas, NumPy, SciPy и scikit-learn. Эти инструменты позволяют производить обработку, анализ и визуализацию данных, что делает Python идеальным выбором для создания приложений для управления данными.

При разработке приложения для анализа и управления данными необходимо учитывать принципы работы с большими данными:

– сбор данных: сбор данных из различных источников, таких как библиотечные базы данных, файлы CSV и другие;

– очистка данных: обработка и очистка данных для устранения ошибок и пропусков;

– анализ данных: применение статистических методов и визуализаций для анализа данных и выявления паттернов;

– хранение данных: эффективное хранение и управление данными для обеспечения быстрого доступа и обработки;

– работа с базами данных: Python поддерживает множество библиотек для работы с различными типами баз данных, включая реляционные (SQLite, MySQL, PostgreSQL) и NoSQL (MongoDB);

– облачные вычисления и Big Data: Python часто используется в облачных вычислениях и обработке Big Data благодаря своей простоте использования и богатой экосистеме библиотек.

В качестве среды для разработки приложения был выбран Visual Studio Code.

«VS Code – популярный редактор с открытым кодом от компании Microsoft с широкими возможностями настройки, что позволяет ему

поддерживать много языков программирования и обеспечивать цветовое выделение синтаксиса, выявление ошибок и форматирование кода.» [9, с. 15].

Ключевые характеристики редактора VS Code:

– легковесность и быстрота: VS Code обладает минимальными требованиями к ресурсам компьютера и быстрым запуском, что позволяет разработчикам мгновенно начать работу с проектом;

– многофункциональность: редактор предоставляет широкий спектр функций для разработки, таких как подсветка синтаксиса, авто дополнение кода, отладчик, система контроля версий (Git), интегрированный терминал и другие инструменты, упрощающие рабочий процесс разработчика;

– расширяемость: VS Code предлагает множество расширений, позволяющих настраивать и расширять функциональность редактора. Расширения могут быть созданы сообществом или индивидуальными разработчиками для улучшения процесса разработки под конкретные нужды;

– интеграция с платформами и инструментами: редактор интегрируется с различными платформами разработки и сервисами;

– поддержка языков программирования: VS Code поддерживает множество языков программирования и обеспечивает инструменты для работы с ними, включая подсветку синтаксиса, автодополнение и проверку ошибок;

– мультиплатформенность: редактор доступен для всех основных операционных систем – Windows, macOS и Linux, обеспечивая консистентный опыт работы для разработчиков на разных платформах.

1.2 Инструментальные средства для разработки интерфейса

Инструменты для разработки графического интерфейса предлагают разные уровни абстракции и функциональные возможности, в зависимости от требований проекта.

Для разработки графического интерфейса на Python можно выделить следующие инструменты:

Существует несколько инструментальных средств которые предлагают разные уровни абстракции и функциональные возможности, в зависимости от требований разрабатываемого проекта. Для разработки графического интерфейса на Python можно выделить следующие инструменты:

а) Tkinter:

– Tkinter является стандартным набором инструментов Python для создания графического интерфейса. Он основан на библиотеке Tk, которая предоставляет элементы управления (widgets) и методы для их организации в приложениях;

– прост в использовании, поставляется вместе с Python, хорошо подходит для простых GUI приложений, кроссплатформенный.

б) PyQt / PySide:

– PyQt и PySide являются мощными инструментами для создания кроссплатформенных GUI приложений на Python. PyQt основан на Qt Framework от Digia, а PySide является его альтернативой с открытым исходным кодом;

– богатые функциональные возможности, кроссплатформенность, поддержка мультимедийных элементов и визуальных эффектов.

в) Kivy:

– Kivy предназначен для создания мультимедийных приложений, включая те, которые требуют мультитач-ввода на устройствах с сенсорным экраном. Это кроссплатформенный фреймворк с открытым исходным кодом;

– поддержка анимации и переходы, возможность разработки для мобильных платформ (Android, iOS).

г) wxPython:

– wxPython предоставляет интерфейс Python для wxWidgets, библиотеки, которая позволяет создавать кроссплатформенные

GUI приложения с использованием нативных элементов интерфейса;

– поддержка кроссплатформенных приложений с использованием нативного внешнего вида, обширные возможности по настройке интерфейса.

д) PyGTK:

– PyGTK представляет собой привязку Python к GTK+, популярной библиотеке для создания графического интерфейса на Linux;

– используется для создания приложений с нативным внешним видом на Linux, хорошо интегрирован с GNOME и другими окружениями рабочего стола;

– необходимость учета совместимости с различными версиями GTK+, поддержка только для Linux и некоторых Unix-подобных систем.

е) PySimpleGUI:

– PySimpleGUI предлагает простой и интуитивно понятный способ создания базовых GUI приложений на Python;

– простота использования, быстрое создание прототипов GUI;

– ограниченные функциональные возможности по сравнению с другими библиотеками.

ж) Toga:

– Toga — это кроссплатформенная библиотека, использующая нативные компоненты для создания GUI на основе инструментария BeeWare;

– поддержка кроссплатформенной разработки, использование нативных компонентов интерфейса.

Для реализации графического интерфейса приложения была выбрана библиотека Tkinter. Tkinter позволяет создавать графический интерфейс без необходимости установки дополнительных пакетов или фреймворков.

«Tkinter – это пакет для Python, предназначенный для работы с библиотекой Tk. Библиотека Tk содержит компоненты графического интерфейса пользователя (graphical user interface – GUI). Эта библиотека написана на языке программирования Tcl.» [33].

Он обладает следующими характеристиками:

– простота использования: Tkinter обладает простым и интуитивно понятным API, что делает его доступным для новичков в области разработки GUI;

– богатый набор виджетов: Tkinter предоставляет широкий набор графических элементов управления (виджетов), таких как кнопки, текстовые поля, метки, списки, радиокнопки и многое другое, что позволяет разработчикам создавать разнообразные интерфейсы;

– кастомизация и стилизация: возможности кастомизации и стилизации виджетов позволяют создавать интерфейсы с учетом индивидуальных дизайнерских предпочтений и требований проекта;

– интеграция с другими библиотеками: Tkinter легко интегрируется с другими библиотеками Python, что позволяет создавать полноценные приложения, используя мощные инструменты и возможности других библиотек;

– обработка событий и взаимодействие: Tkinter предоставляет механизм обработки событий пользовательского взаимодействия, таких как нажатия кнопок, ввод текста и другие, что позволяет создавать интерактивные приложения.

В качестве дополнения к Tkinker была использована библиотека TtkThemes.

«TtkThemes – это расширение для библиотеки tkinter, стандартного инструмента для создания графических пользовательских интерфейсов в Python, которое предоставляет дополнительные темы оформления для виджетов.» [4].

Основные характеристики TtkThemes:

– предлагает широкий выбор тем оформления, которые можно легко применять к приложениям на основе tkinter. Темы включают стили для различных виджетов, таких как кнопки, метки, поля ввода;

– легко интегрируется с tkinter, позволяя разработчикам быстро переключаться между темами. Использование TtkThemes требует минимальных изменений в коде существующих приложений tkinter;

– поддерживает Windows, macOS и Linux, обеспечивая кроссплатформенную совместимость;

– TtkThemes предоставляет более современные и эстетичные стили оформления по сравнению с базовыми темами tkinter. Улучшенная визуализация делает интерфейс приложения более привлекательным и удобным для пользователей;

– позволяет изменять цвета, шрифты и другие параметры оформления виджетов.

1.3 Средства для управления и анализа данных

В данном параграфе опишем средства для управления и анализа данных.

Для разработки основного функционала приложения были использованы библиотеки Python.

Библиотека — набор файлов, модулей, классов, функций, в которых реализован весь функционал решения конкретной проблемы [8].

Для управления данными была выбрана библиотека Pandas.

«Pandas — это библиотека для анализа и обработки данных в языке программирования Python.» [13, с. 46].

Pandas обладает следующими преимуществами:

– удобная работа с данными: pandas предоставляет удобные структуры данных, такие как DataFrame, которые позволяют легко импортировать,

манипулировать и анализировать данные. Это особенно полезно при работе с большими объемами данных;

– мощные функции обработки данных: библиотека Pandas предоставляет множество функций для фильтрации, сортировки, группировки и агрегирования данных. Это позволяет производить различные манипуляции с данными для получения нужной информации;

– интеграция с другими библиотеками: Pandas легко интегрируется с другими популярными библиотеками Python, такими как NumPy (для работы с массивами чисел), Matplotlib (для визуализации данных) и Scikit-learn (для машинного обучения);

– обработка отсутствующих данных: Pandas предоставляет мощные инструменты для обработки отсутствующих данных, таких как заполнение пропущенных значений или удаление строк с отсутствующими данными;

– эффективность и производительность: благодаря оптимизированным алгоритмам и структурам данных, Pandas обеспечивает высокую производительность при работе с данными, даже при больших объемах информации;

– широкий функционал: библиотека Pandas предоставляет множество функций для работы с временными рядами, текстовыми данными, категориальными переменными и многое другое, что делает ее мощным инструментом для анализа различных типов данных.

«Анализ данных — это процесс получения информации из данных путем создания моделей и применения математического аппарата для поиска закономерностей.» [10, с. 22].

К этапам анализа данных относится:

а) сбор данных:

– сбор данных из различных источников;

– проверка данных на наличие ошибок и пропущенных значений.

б) очистка данных:

– удаление или исправление ошибок и пропущенных значений;

– преобразование данных в удобный формат;

– нормализация и стандартизация данных.

в) исследовательский анализ данных:

– использование статистических методов;

– выявление основных характеристик, тенденций и паттернов в данных;

– построение графиков и диаграмм;

г) визуализация данных:

– создание визуальных представлений данных;

– использование графиков, диаграмм;

– интерпретация и представление результатов;

– анализ результатов и формулирование выводов на основе полученных данных.

В качестве библиотеки для реализации анализа данных, представления их в виде диаграмм, была выбрана библиотека Matpotlib.

«Matplotlib — базовый инструмент для создания готового к публикации графика. Он широко применяется сам по себе и в качестве основы для других библиотек с целью построения графиков.» [14, с. 174].

Основными характеристики Matplotlib являются:

а) широкий спектр графиков:

– поддержка различных типов графиков, таких как линейные графики, гистограммы, графики рассеяния, графики с областями, столбчатые диаграммы, круговые диаграммы и другие.

б) высокая настройка:

– возможность настройки практически всех аспектов графика, включая цвета, стили линий, размеры шрифтов, метки осей и легенды.

в) интерактивные графики:

– поддержка интерактивных функций, таких как масштабирование, панорамирование и аннотирование графиков;

– интеграция с IPython и Jupyter Notebook для интерактивной работы.

г) совместимость:

– возможность сохранения графиков в различных форматах, таких как PNG, PDF, SVG и других;

–интеграция с другими библиотеками, такими как pandas, NumPy и seaborn.

д) производительность:

– способность обрабатывать и визуализировать большие объемы данных.

1.4 Характеристики баз данных

В данном параграфе дадим определение базе данных, большим данным. Проведем анализ существующих хранилищ данных, а также их основные характеристики.

«Большие данные — это обозначение совокупности структурированных и неструктурированных данных (информации) сверхбольших объемов и широкого многообразия.» [28, с. 97].

Большие данные характеризуются следующими основными аспектами:

– объем: ключевая характеристика больших данных — их большой объем;

– скорость: большие данные часто генерируются и поступают с высокой скоростью;

– разнообразие: данные могут поступать в различных форматах (структурированные, неструктурированные);

«База данных — это упорядоченный набор структурированной информации или данных, которые обычно хранятся в электронном виде в компьютерной системе.» [31].

Базы данных играют ключевую роль в приложениях для управления и анализа данных по нескольким причинам:

а) структурирование данных:

– позволяют организовывать данные в структурированном виде, для облегчения хранения, поиска и управления.

б) эффективное хранение и доступ:

– обеспечивают быстрый доступ к большим объемам данных, позволяя эффективно управлять данными.

в) целостность данных:

– использование баз данных позволяет гарантировать целостность данных с помощью механизмов транзакций и ограничений целостности.

г) безопасность данных:

– базы данных предоставляют механизмы контроля доступа и аутентификации для защиты данных от несанкционированного доступа.

д) конкурентный доступ:

– базы данных позволяют многопользовательский доступ и управление конкурентными запросами.

е) управление и анализ данных:

– базы данных предлагают инструменты для управления данными (резервное копирование, восстановление, репликация, масштабирование).

– базы данных поддерживают сложные аналитические запросы и операции для анализа данных.

Выбор базы данных для приложения Python зависит от конкретных требований проекта.

Основные популярные базы данных, которые могут использоваться в приложениях для анализа и управления данными на языке программирования Python:

– «SQLite — это встраиваемая СУБД, когда система управления встраивается в саму программу. Это значит, что все запросы и команды идут в базу не через посредника, а напрямую из приложения.» [40];

– «PostgreSQL — это бесплатная СУБД с открытым исходным кодом. С помощью PostgreSQL можно создавать, хранить базы данных и работать с данными с помощью запросов на языке SQL.» [26];

– «MongoDB — система управления базами данных, которая работает с документоориентированной моделью данных. В отличие от реляционных СУБД, MongoDB не требуются таблицы, схемы или отдельный язык запросов. Информация хранится в виде документов либо коллекций.» [37];

– «Redis — система хранения данных в виде структур. Это нереляционная СУБД с открытым исходным кодом, организованная по принципу «ключ – значение». Она является вспомогательной и выполняет функцию хранилища (Redis storage) и кеша для основной, центральной базы данных.» [39];

– «Elasticsearch – это одна из самых популярных поисковых систем в области Big Data, масштабируемое нереляционное хранилище данных с открытым исходным кодом, аналитическая NoSQL-СУБД с широким набором функций полнотекстового поиска.» [32].

В качестве хранилищ данных был выбран csv формат файла.

«CSV — текстовый формат, предназначенный для представления табличных данных. Каждая строка файла – это одна строка таблицы. Значения отдельных колонок разделяются разделительным символом.» [36].

Можно выделить следующие основные характеристики CSV-файлов:

а) простота и универсальность:

– CSV-файлы легко читаются и создаются большинством текстовых редакторов;

– поддерживаются различными программами, включая электронные таблицы и базы данных.

б) структура:

– первая строка обычно содержит заголовки столбцов;

– каждая последующая строка представляет собой одну запись данных, каждое значение разделено запятой или другим разделителем.

в) совместимость:

– «CSV-файлы могут быть импортированы и экспортированы из различных систем управления базами данных и программ анализа данных.» [17, с. 23];

– широко используются для передачи данных между приложениями и системами.

Выводы: в первом параграфе нами были рассмотрены:

– инструментальные средства для разработки графического интерфейса приложения;

– средства для управления и анализа данных в приложении;

– характеристики баз данных.

2 Разработка приложения для анализа и управления большими данными в библиотеке

2.1 Функциональные и нефункциональные требования

Анализ и управление данными в библиотеке необходимы для улучшения ее работы, повышения эффективности использования ресурсов и удовлетворения потребностей пользователей. Внедрение приложения в библиотеку позволит выполнить следующие задачи:

– анализ данных для выявления предпочтений пользователей;

– долгосрочное планирование развития библиотеки, что позволит внедрять новые технологии и услуги;

– отчетность и обоснование данных;

– улучшение существующих сервисов и процедур на основе фидбека от пользователей.

Приложение на Python для управления и анализа больших данных в библиотеке может предоставлять широкий спектр функциональности, чтобы облегчить управление каталогом книг, анализировать данные о читателях и их запросах, а также предоставлять полезную информацию для администраторов.

Разрабатываемое приложение должно иметь функциональные и не функциональные требования.

Функциональные требования:

а) управление каталогом книг:

– добавление новых книг в каталог с указанием информации о книге (название, автор, жанр, год публикации, описание);

– возможность редактирования информации о книге;

– добавление и удаление книг из каталога.

б) учет читателей:

– регистрация и удаление новых читателей с указанием персональных данных (ФИО, контактная информация, адрес).

в) учет авторов:

– возможность добавлять, редактировать и удалять информацию об авторах.

г) учет выдачи:

– регистрация выдачей книг с указанием даты выдачи и даты возврата, а также персональным id читателя, id книги, читательского билета.

д) анализ данных

– определение самых популярных книг на основе количества выдач, популярных жанров книг на основе количества книг в каждом жанре;

– просмотр количества оставшихся книг каждого типа в библиотеке.

е) обработка событий:

– обработка ошибок в коде;

– успешное выполнение операций.

Нефункциональные требования:

а) производительность:

– быстрота и отзывчивость приложения при обработке и анализе больших данных.

б) масштабируемость:

– бесперебойная работа с различными объемами данных и количеством пользователей.

в) надежность:

– стабильность и надежность.

г) удобство использования:

– интуитивно понятный и удобный для пользователей интерфейс.

2.2 Разработка хранилища данных

В данном параграфе разработаем хранилище для данных.

Для реализации основных функций, необходимо заполнить таблицы, в которых будут храниться данные. Были созданы таблицы, и столбцы. Необходимо определить типы данных для каждого столбца этих таблиц. На этапе разработки функционала, типы данных таблиц будут привязаны к коду.

Созданы 4 таблицы: «authors.csv», «books.csv», «readers.csv», «issues.csv».

Таблица 1 содержит информацию об авторах книг.

Таблица 1 – Атрибуты таблицы «Authors»

Название столбца	**Тип данных**
ID Автора	Integer
ФИО	String
Дата рождения	Date
Страна	String

Каждая строка таблицы авторов представляет отдельного автора. Столбцы содержат следующие данные:

– «ID автора»: уникальный идентификатор, используемый для идентификации автора в системе;

– «ФИО»: полное имя автора;

– «Дата рождения»: дата рождения автора;

– «Страна»: страна, в которой автор родился или в которой он проживает на данный момент.

Таблица 2 содержит информацию о книгах в библиотеке.

Таблица 2 – Атрибуты таблицы «Books»

Название столбца	**Тип данных**
ID Книги	Integer
Название	String
Автор	String
Год издания	Integer
ISBN	String
Жанр	String
Количество в библиотеке	Integer

Каждая строка представляет отдельную книгу, а столбцы содержат следующие данные:

– «ID Книги»: уникальный идентификатор, используемый для идентификации книги в системе;

– «Название»: название книги;

– «Автор»: имя автора книги. Позволяет связать её с информацией об авторе из таблицы «authors.csv»;

– «Год издания»: год, когда книга была издана;

– «ISBN»: международный стандартный книжный номер, который позволяет идентифицировать книгу в мировом масштабе;

– «Жанр»: жанр или категория, к которой относится книга. Используется для анализа популярности жанров;

– «Количество в библиотеке»: количество экземпляров книги в библиотеке.

Таблица 3 содержит информацию о читателях библиотеки.

Таблица 3 – Атрибуты сущности «Readers»

Название столбца	**Тип данных**
ID Читателя	Integer
ФИО	String
Адрес	String
Email	String
Номер телефона	String

Каждая строка представляет отдельного читателя, а столбцы содержат следующие данные:

– «ID Читателя»: уникальный идентификатор (целое число), используемый для идентификации читателя в системе. Каждому читателю присваивается уникальный ID для удобства управления и связи с другими таблицами;

– «ФИО»: полное имя читателя;

– «Адрес»: адрес проживания читателя;

– «Email»: адрес электронной почты читателя;

– «Номер телефона»: контактный номер телефона читателя;

Таблица 4 содержит информацию о выдаче книг читателям.

Таблица 4 – Атрибуты сущности «Issues»

Название столбца	**Тип данных**
ID Выдачи	Integer
ID Книги	Integer
ID Читателя	Integer
Дата выдачи	Date
Дата возврата	Date
Читательский билет	String

Каждая строка представляет собой отдельную выдачу книги читателю, а столбцы содержат следующие данные:

– «ID Выдачи»: уникальный идентификатор, используемый для идентификации каждой конкретной выдачи книги. Каждая выдача получает свой уникальный ID для отслеживания и управления выдачами в библиотеке;

– «ID Книги»: идентификатор книги, которая была выдана читателю. Этот ID соответствует ID книги из таблицы books.csv и позволяет определить, какая книга была выдана;

– «ID Читателя»: идентификатор читателя, которому была выдана книга. Этот ID соответствует ID читателя из таблицы readers.csv. Позволяет определить, какому читателю была выдана книга;

– «Дата выдачи»: дата, когда книга была выдана читателю. Позволяет отслеживать сроки выдачи и контролировать их соблюдение;

– «Дата возврата»: предполагаемая дата возврата книги. Дата указывается при выдаче книги и используется для определения сроков, по истечении которых книгу необходимо вернуть в библиотеку;

– «Читательский билет»: уникальный идентификатор. Может использоваться вместо ID читателя.

Информация о выдаче книг позволяет библиотеке отслеживать текущее положение книг в использовании, контролировать сроки их возврата, а также предоставлять своевременные уведомления читателям о необходимости возврата книг. Это также может быть использовано для анализа популярности книг, сроков их использования и других аспектов деятельности библиотеки.

Для нормального функционирования программы необходимо перевести наименования столбцов на латинский язык. Столбцы переименованы следующим образом:

– таблица выдач – «issues»: «ID», «BookID», «ReaderID», «IssueDate», «ReturnDate», «Ticket»;

– таблица читателей – «readers»: «ReaderID», «Name», «Address», «Email», «Phone»;

– таблица авторов – «authors»: «ID», «Name», «BirthDate», «Country»;

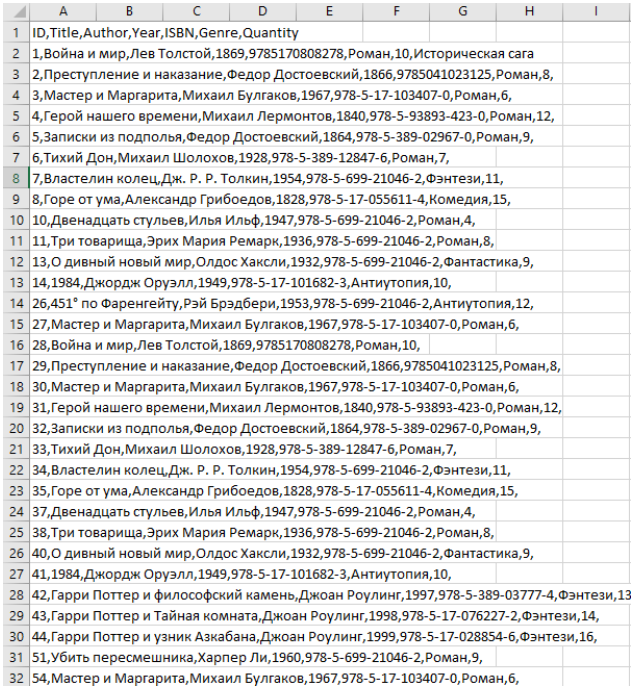
– таблица книг – «books»: «ID», «Title», «Author», «Year», «ISBN», «Genre», «Quantity».

После создания «CSV» файлов, определения типов данных и названий для каждого столбца необходимо заполнить данными эти таблицы. Заполненная таблица «authors» представлена на рисунке 1.

	A	B	C	D	E	F	G	H	I
1	ID,Name,BirthDate,Country								
2	1,Лев Николаевич Толстой,1829-09-09,Россия								
3	2,Петр Петрович Петров,1982-08-25,Россия								
4	3,Сидоров Сидор Сидорович,1990-11-18,Россия								
5	4,Козлова Анна Владимировна,1988-03-03,Россия								
6	5,Смирнов Сергей Васильевич,1979-09-30,Россия								
7	6,Николаева Елена Павловна,1985-06-17,Россия								
8	7,Кузнецов Игорь Александрович,1973-12-08,Россия								
9	8,Ковалев Владимир Дмитриевич,1980-04-21,Россия								
10	9,Морозова Ольга Игоревна,1977-07-14,Россия								
11	10,Белова Марина Сергеевна,1970-10-27,Россия								
12	11,Зайцев Андрей Викторович,1983-01-15,Россия								
13	12,Соколов Виктор Алексеевич,1978-04-08,Россия								
14	13,Григорьева Екатерина Степановна,1986-07-23,Россия								
15	14,Александрова Татьяна Валерьевна,1984-10-06,Россия								
16	15,Дмитриева Лариса Андреевна,1971-12-29,Россия								
17	16,Федорова Евгения Сергеевна,1976-02-20,Россия								
18	17,Максимов Илья Михайлович,1981-05-03,Россия								
19	18,Павлов Василий Викторович,1989-08-16,Россия								
20	19,Андреев Артем Анатольевич,1974-11-09,Россия								
21	20,Галкин Денис Владимирович,1972-02-02,Россия								
22	21,Широков Егор Игоревич,1987-06-25,Россия								
23	22,Куликов Валентин Петрович,1970-09-18,Россия								
24	23,Воронов Антон Семенович,1985-12-11,Россия								
25	24,Тарасова Юлия Федоровна,1978-03-05,Россия								
26	25,Медведев Михаил Иванович,1983-05-28,Россия								
27	26,Романова Анастасия Николаевна,1976-08-21,Россия								
28	27,Захаров Алексей Григорьевич,1980-11-14,Россия								
29	28,Демидова Оксана Сергеевна,1987-02-06,Россия								
30	29,Бобров Владислав Денисович,1979-04-01,Россия								
31	30,Кириллов Екатерина Александровна,1974-06-24,Россия								
32	31,Фомина Светлана Павловна,1981-09-17,Россия								
33	32,Кулаков Игнатий Максимович,1988-12-10,Россия								
34	33,Антонов Владимир Семенович,1977-01-03,Россия								

Рисунок 1 – Содержимое таблицы «Authors»

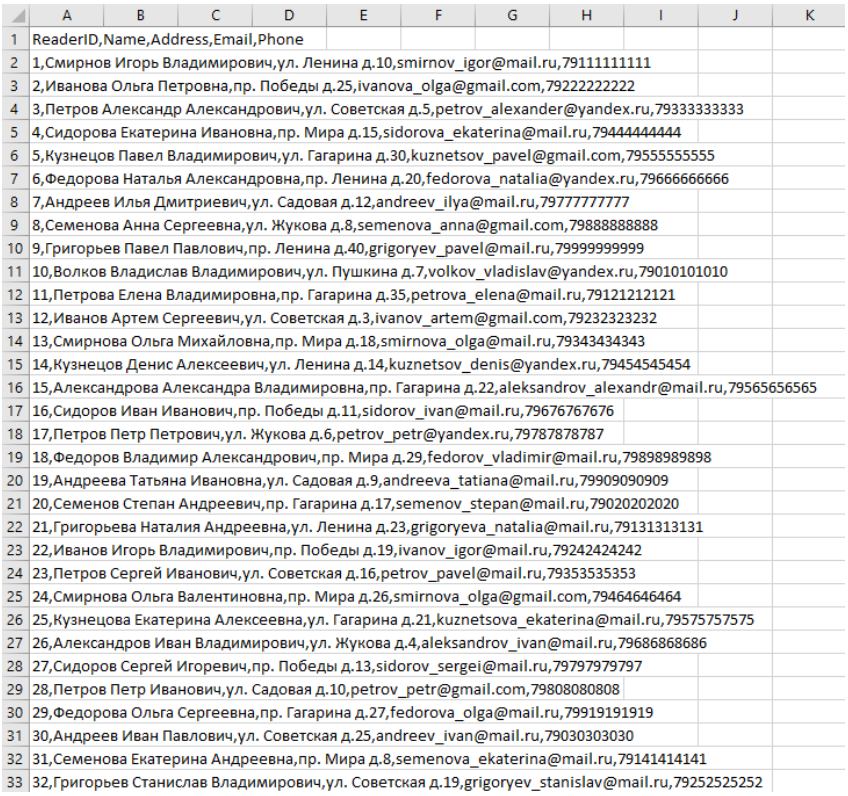
Аналогично нужно заполнить оставшиеся таблицы. Таблица «books» представлена на рисунке 2.



	A	B	C	D	E	F	G	H	I
1	ID,Title,Author,Year,ISBN,Genre,Quantity								
2	1,Война и мир,Лев Толстой,1869,9785170808278,Роман,10,Историческая сага								
3	2,Преступление и наказание,Федор Достоевский,1866,9785041023125,Роман,8,								
4	3,Мастер и Маргарита,Михаил Булгаков,1967,978-5-17-103407-0,Роман,6,								
5	4,Герой нашего времени,Михаил Лермонтов,1840,978-5-93893-423-0,Роман,12,								
6	5,Записки из подполья,Федор Достоевский,1864,978-5-389-02967-0,Роман,9,								
7	6,Тихий Дон,Михаил Шолохов,1928,978-5-389-12847-6,Роман,7,								
8	7,Властелин колец,Дж. Р. Р. Толкин,1954,978-5-699-21046-2,Фэнтези,11,								
9	8,Горе от ума,Александр Грибоедов,1828,978-5-17-055611-4,Комедия,15,								
10	10,Двенадцать стульев,Илья Ильф,1947,978-5-699-21046-2,Роман,4,								
11	11,Три товарища,Эрих Мария Ремарк,1936,978-5-699-21046-2,Роман,8,								
12	13,О дивный новый мир,Олдос Хаксли,1932,978-5-699-21046-2,Фантастика,9,								
13	14,1984,Джордж Оруэлл,1949,978-5-17-101682-3,Антиутопия,10,								
14	26,451° по Фаренгейту,Рэй Брэдбери,1953,978-5-699-21046-2,Антиутопия,12,								
15	27,Мастер и Маргарита,Михаил Булгаков,1967,978-5-17-103407-0,Роман,6,								
16	28,Война и мир,Лев Толстой,1869,9785170808278,Роман,10,								
17	29,Преступление и наказание,Федор Достоевский,1866,9785041023125,Роман,8,								
18	30,Мастер и Маргарита,Михаил Булгаков,1967,978-5-17-103407-0,Роман,6,								
19	31,Герой нашего времени,Михаил Лермонтов,1840,978-5-93893-423-0,Роман,12,								
20	32,Записки из подполья,Федор Достоевский,1864,978-5-389-02967-0,Роман,9,								
21	33,Тихий Дон,Михаил Шолохов,1928,978-5-389-12847-6,Роман,7,								
22	34,Властелин колец,Дж. Р. Р. Толкин,1954,978-5-699-21046-2,Фэнтези,11,								
23	35,Горе от ума,Александр Грибоедов,1828,978-5-17-055611-4,Комедия,15,								
24	37,Двенадцать стульев,Илья Ильф,1947,978-5-699-21046-2,Роман,4,								
25	38,Три товарища,Эрих Мария Ремарк,1936,978-5-699-21046-2,Роман,8,								
26	40,О дивный новый мир,Олдос Хаксли,1932,978-5-699-21046-2,Фантастика,9,								
27	41,1984,Джордж Оруэлл,1949,978-5-17-101682-3,Антиутопия,10,								
28	42,Гарри Поттер и философский камень,Джоан Роулинг,1997,978-5-389-03777-4,Фэнтези,13								
29	43,Гарри Поттер и Тайная комната,Джоан Роулинг,1998,978-5-17-076227-2,Фэнтези,14,								
30	44,Гарри Поттер и узник Азкабана,Джоан Роулинг,1999,978-5-17-028854-6,Фэнтези,16,								
31	51,Убить пересмешника,Харпер Ли,1960,978-5-699-21046-2,Роман,9,								
32	54,Мастер и Маргарита,Михаил Булгаков,1967,978-5-17-103407-0,Роман,6,								

Рисунок 2 – Содержимое таблицы «Books»

Часть заполненной таблицы «readers» представлена на рисунке 3.



	A	B	C	D	E	F	G	H	I	J	K
1	ReaderID,Name,Address,Email,Phone										
2	1,Смирнов Игорь Владимирович,ул. Ленина д.10,smirnov_igor@mail.ru,79111111111										
3	2,Иванова Ольга Петровна,пр. Победы д.25,ivanova_olga@gmail.com,79222222222										
4	3,Петров Александр Александрович,ул. Советская д.5,petrov_alexander@yandex.ru,79333333333										
5	4,Сидорова Екатерина Ивановна,пр. Мира д.15,sidorova_ekaterina@mail.ru,79444444444										
6	5,Кузнецов Павел Владимирович,ул. Гагарина д.30,kuznetsov_pavel@gmail.com,79555555555										
7	6,Федорова Наталья Александровна,пр. Ленина д.20,fedorova_natalia@yandex.ru,79666666666										
8	7,Андреев Илья Дмитриевич,ул. Садовая д.12,andreev_ilya@mail.ru,79777777777										
9	8,Семенова Анна Сергеевна,ул. Жукова д.8,semenova_anna@gmail.com,79888888888										
10	9,Григорьев Павел Павлович,пр. Ленина д.40,grigoryev_pavel@mail.ru,79999999999										
11	10,Волков Владислав Владимирович,ул. Пушкина д.7,volkov_vladislav@yandex.ru,79010101010										
12	11,Петрова Елена Владимировна,пр. Гагарина д.35,petrova_elena@mail.ru,79121212121										
13	12,Иванов Артем Сергеевич,ул. Советская д.3,ivanov_artem@gmail.com,79232323232										
14	13,Смирнова Ольга Михайловна,пр. Мира д.18,smirnova_olga@mail.ru,79343434343										
15	14,Кузнецов Денис Алексеевич,ул. Ленина д.14,kuznetsov_denis@yandex.ru,79454545454										
16	15,Александрова Александра Владимировна,пр. Гагарина д.22,aleksandrov_alexandr@mail.ru,79565656565										
17	16,Сидоров Иван Иванович,пр. Победы д.11,sidorov_ivan@mail.ru,79676767676										
18	17,Петров Петр Петрович,ул. Жукова д.6,petrov_petr@yandex.ru,79787878787										
19	18,Федоров Владимир Александрович,пр. Мира д.29,fedorov_vladimir@mail.ru,79898989898										
20	19,Андреева Татьяна Ивановна,ул. Садовая д.9,andreeva_tatiana@mail.ru,79909090909										
21	20,Семенов Степан Андреевич,пр. Гагарина д.17,semenov_stepan@mail.ru,79020202020										
22	21,Григорьева Наталия Андреевна,ул. Ленина д.23,grigoryeva_natalia@mail.ru,79131313131										
23	22,Иванов Игорь Владимирович,пр. Победы д.19,ivanov_igor@mail.ru,79242424242										
24	23,Петров Сергей Иванович,ул. Советская д.16,petrov_pavel@mail.ru,79353535353										
25	24,Смирнова Ольга Валентиновна,пр. Мира д.26,smirnova_olga@gmail.com,79464646464										
26	25,Кузнецова Екатерина Алексеевна,ул. Гагарина д.21,kuznetsova_ekaterina@mail.ru,79575757575										
27	26,Александров Иван Владимирович,ул. Жукова д.4,aleksandrov_ivan@mail.ru,79686868686										
28	27,Сидоров Сергей Игоревич,пр. Победы д.13,sidorov_sergei@mail.ru,79797979797										
29	28,Петров Петр Иванович,ул. Садовая д.10,petrov_petr@gmail.com,79808080808										
30	29,Федорова Ольга Сергеевна,пр. Гагарина д.27,fedorova_olga@mail.ru,79919191919										
31	30,Андреев Иван Павлович,ул. Советская д.25,andreev_ivan@mail.ru,79030303030										
32	31,Семенова Екатерина Андреевна,пр. Мира д.8,semenova_ekaterina@mail.ru,79141414141										
33	32,Григорьев Станислав Владимирович,ул. Советская д.19,grigoryev_stanislav@mail.ru,79252525252										

Рисунок 3 – Содержимое таблицы «Readers»

Часть заполненной таблицы «issues» представлена на рисунке 4.

	A	B	C	D	E	F
1	ID,BookID,ReaderID,IssueDate,ReturnDate,Ticket					
2	1,12,37,2023-04-10,2023-05-01,Б437					
3	2,98,203,2022-07-15,2022-08-01,А126					
4	3,57,152,2023-09-22,2023-10-10,Г319					
5	4,25,305,2022-11-05,2022-11-20,В204					
6	5,143,78,2024-01-12,2024-02-01,А509					
7	6,76,234,2023-03-17,2023-04-05,Б619					
8	7,33,42,2024-06-30,2024-07-20,В108					
9	8,105,119,2022-09-04,2022-09-25,А731					
10	9,68,295,2023-11-19,2023-12-10,Б812					
11	10,122,377,2024-02-25,2024-03-15,Г218					
12	11,88,144,2022-12-10,2023-01-01,В925					
13	12,49,216,2023-05-31,2023-06-15,Б503					
14	13,136,361,2024-01-02,2024-01-20,А412					
15	14,73,89,2022-10-15,2022-10-30,Г625					
16	15,38,298,2024-04-10,2024-04-25,А814					
17	16,102,196,2023-08-20,2023-09-10,Б317					
18	17,64,373,2022-11-30,2022-12-15,В702					
19	18,17,311,2024-06-10,2024-06-30,А628					
20	19,124,152,2023-02-05,2023-02-25,Б935					
21	20,91,377,2022-12-20,2023-01-10,Г401					
22	21,43,194,2023-10-01,2023-10-20,В802					
23	22,133,268,2024-03-15,2024-04-01,А126					
24	23,82,17,2023-07-05,2023-07-25,Г516					
25	24,59,224,2022-10-01,2022-10-15,В327					
26	25,148,291,2024-05-20,2024-06-05,А915					
27	26,69,49,2023-01-20,2023-02-10,Б709					
28	27,29,212,2024-04-01,2024-04-20,Г819					
29	28,115,112,2022-12-30,2023-01-20,В622					
30	29,78,387,2023-04-20,2023-05-10,А303					
31	30,45,231,2022-11-15,2022-12-05,Г124					
32	31,138,126,2024-02-15,2024-03-01,Р’507					

Рисунок 4 – Содержимое таблицы «Issues»

2.3 Разработка графического интерфейса

В данном параграфе разработаем графический интерфейс приложения.

В процессе разработки макета для приложения были проанализированы существующие приложения для анализа и управления данными в библиотеке.

Функциональные требования, представленные в параграфе 2.1, позволили разработать макет, целью которого является последующее создание привлекательного дизайна и удобной навигации для пользователя.

Для реализации функциональных требований был разработан макет главного окна приложения, в котором отображены основные блоки для реализации функционала приложения. Каждый блок обладает своим функционалом.

Главное окно приложения должно содержать блок для навигации между таблицами, блок для поиска табличных значений, сама таблица, которую можно переключать с помощью блока навигации, блок для управления данными.

Разработанный макет главного окна представлен на рисунке 5.

Рисунок 5 – Макет главного окна

После определения расположения основных блоков на главном окне приложения, были определены размеры блоков, разработан дизайн главного окна приложения, а также были связаны таблицы в формате «CSV» с таблицами приложения.

На главном экране приложения, во вкладке «Управление», представленное на рисунке 6, отображен удобный пользовательский интерфейс для управления базой данных приложения. Основные компоненты этого экрана:

– кнопки категорий: на верхней части экрана были добавлены кнопки для выбора категории данных: «Авторы», «Книги», «Читатели», «Выдачи». При нажатии на каждую из этих кнопок пользователь переходит к просмотру соответствующей таблицы с данными;

– поиск: в верхней части таблицы расположено поле поиска, где пользователь может ввести ключевое слово для поиска определенных данных в текущей таблице. При нажатии на кнопку «Поиск» выполняется поиск и отображение результатов;

– таблица данных: занимает основную часть экрана, которая отображает информацию из выбранной категории. В таблице отображаются различные атрибуты данных из разных таблиц в виде строк и столбцов. Пользователь имеет возможность прокручивать таблицу, для просмотра всех записей, с помощью колеса компьютерной мыши. Каждая строка в таблице представляет собой отдельный элемент данных, которые загружаются из «CSV» файлов;

– кнопки управления строками: в нижней части окна находятся кнопки управления строками: «Добавить строку», «Удалить строку», «Сохранить изменения». Пользователь может добавлять и удалять строки таблиц, сохранять изменения, внесенные в таблицу.

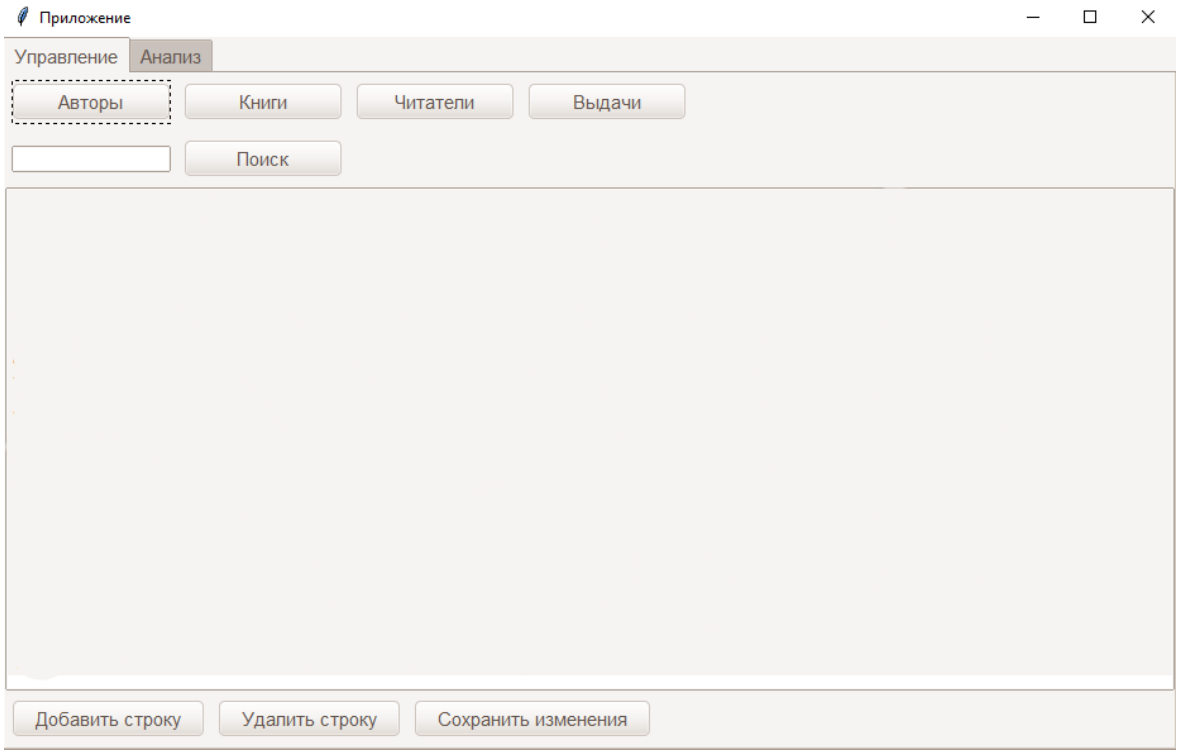


Рисунок 6 – Интерфейс главного окна приложения для управления данными

При двойном нажатии на строку в таблице, добавлено новое окно, представленное на рисунке 7. С помощью данного окна можно редактировать значения строк в таблицах. Это окно имеет 2 дополнительные кнопки: «OK» – сохранить изменения, и «Cancel» – выйти из редактора значений.

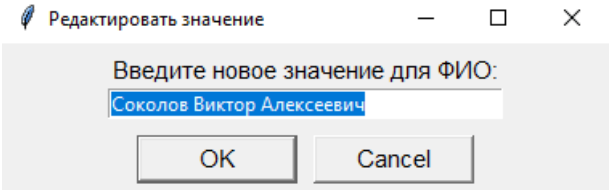


Рисунок 7 – Окно редактирования табличных значений

Для кнопки «Добавить строку» были добавлены всплывающие окна, позволяющие добавлять новые строки в таблицы. Добавлена возможность создать пустую строку, путем нажатия кнопки «OK». Всплывающее окно для кнопки «Добавить строку» представлено на рисунке 8.

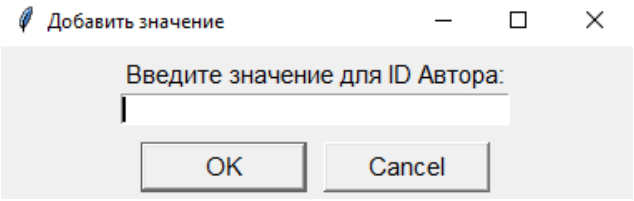


Рисунок 8 – Окно для кнопки «Добавить строку»

Для кнопки «Сохранить изменения» было добавлено всплывающее окно, представленное на рисунке 9. Данное окно информирует о том, что сохранение таблиц после внесения правок прошло успешно.

Для разработки главного окна приложения, отвечающего за анализ, необходимо разработать макет второй основной вкладки приложения – «Анализ». В данной вкладке будут расположены следующие блоки:

– навигационный блок;

– блок выбора типа анализа;

– блок предоставления текстового результата анализа;

– блок предоставления графического результата анализа.

Макет вкладки «Управление» представлен на рисунке 9.

Рисунок 9 – Интерфейс главного окна приложения для анализа данных

С помощью данного макета была разработана вторая основная вкладка приложения – «Анализ», представленная на рисунке 10. На главном экране приложения, во вкладке «Анализ», находятся следующие элементы:

– фрейм анализа: располагается на всю ширину и высоту вкладки. В этом фрейме размещаются кнопки для запуска анализа данных и вывода результатов анализа;

– фрейм кнопок анализа: размещены три кнопки: «Анализ популярности книг», «Анализ популярности жанров», «Анализ остатка книг»;

– кнопка «Анализ популярности книг»: выполняется анализ данных о выдачах книг. Результаты анализа отображаются в виде текста в виджете «Text» под кнопками анализа, а также в виде графика на той же вкладке;

– кнопка «Анализ популярности жанров»: при нажатии на эту кнопку выполняется анализ данных о книгах. Результаты анализа, включая самые популярные жанры и их количество, отображаются в виджете «Text» и на графике;

– кнопка «Анализ остатка книг»: при нажатии на эту кнопку выполняется анализ данных о книгах.

Результаты анализа, включая количество оставшихся книг по каждому их них, отображаются в виджете «Text» и на графике.

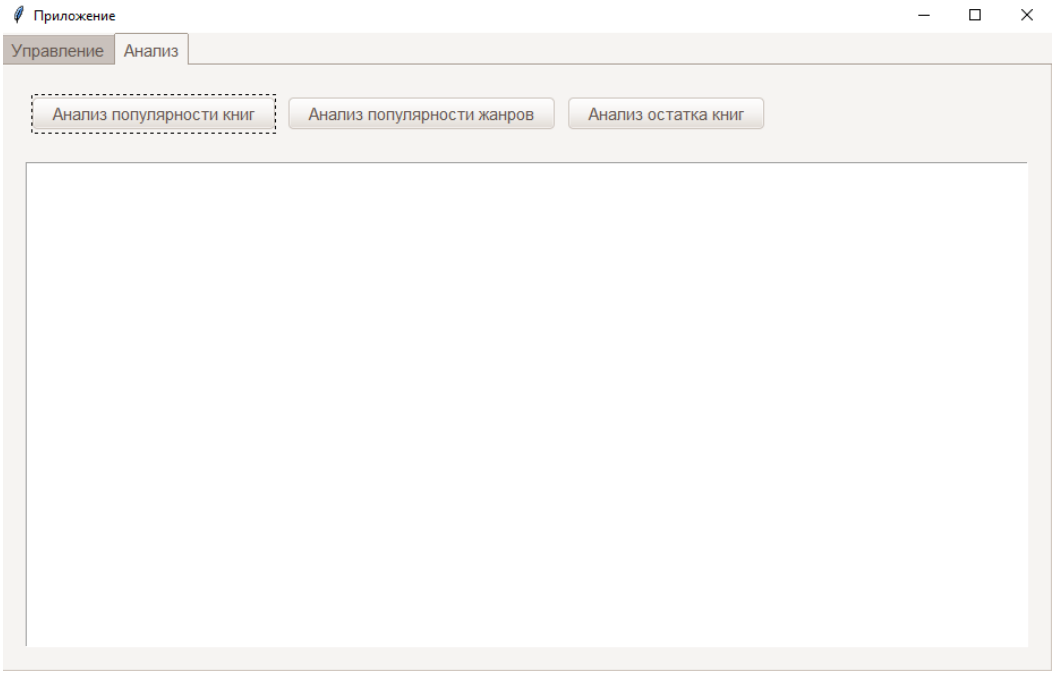


Рисунок 10 – Интерфейс главного окна приложения для анализа данных

2.4 Разработка функционала приложения

Разрабатываемое приложение должно иметь следующий функционал:

а) управление каталогом книг:

– добавление новых книг в каталог с указанием информации о книге (название, автор, жанр, год публикации, описание);

– возможность редактирования информации о книге;

– добавление и удаление книг из каталога.

б) учет читателей:

– регистрация и удаление новых читателей с указанием персональных данных (ФИО, контактная информация, адрес).

в) учет авторов:

– возможность добавлять, редактировать и удалять информацию об авторах (ФИО, дата рождения, страна).

г) учет выдачи:

– регистрация выдачей книг с указанием даты выдачи и даты возврата, персональным id читателя, id книги, а также указанием читательского билета.

д) анализ данных:

– определение самых популярных книг на основе количества выдач;

– определение самых популярных жанров книг на основе количества книг в каждом жанре;

– просмотр количества оставшихся книг каждого типа в библиотеке;

– представление данных в текстовом и графическом вариантах.

е) обработка событий:

– обработка ошибок в коде;

– успешное выполнение операций;

– уведомление пользователя в случае возникновения определенного события.

Для реализации функционала приложения была разработана схема функциональной архитектуры приложения, представленная на рисунке 11.

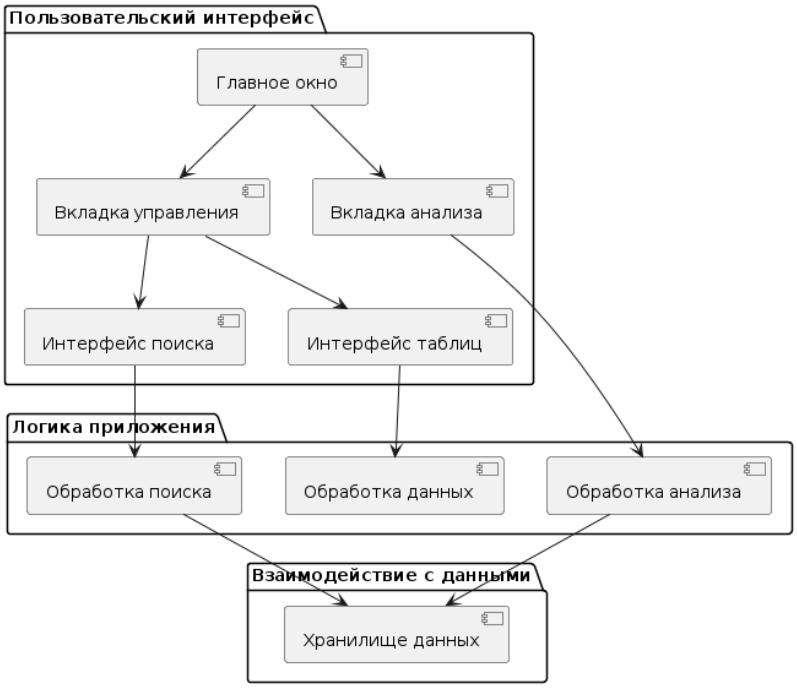


Рисунок 11 – Функциональная архитектура приложения

Схема, представленная выше, даёт представление о функциональной архитектуре приложения, включающая основные компоненты и их взаимодействие:

а) пользовательский интерфейс:

– «Главное окно» – основное окно приложения, содержащее вкладки для управления и анализа;

– «Вкладка управления» – вкладка, предоставляющая интерфейс для управления данными, включая поиск и работу с таблицами;

– «Вкладка анализа» – вкладка, предоставляющая интерфейс для анализа данных;

– «Интерфейс поиска» – компонент для выполнения поисковых операций;

– «Интерфейс таблиц» – компонент для работы с таблицами данных.

б) логика приложения:

– «Обработка данных» – логика обработки данных, включая их загрузку и сохранение;

– «Обработка поиска» – логика обработки поисковых запросов;

– «Обработка анализа» – логика выполнения анализа данных.

в) взаимодействие с данными:

– «Файлы данных» – хранение данных в виде csv файлов.

Взаимодействия:

– главное окно взаимодействует с вкладками управления и анализа;

– интерфейс таблиц, поиска, анализа, взаимодействует с обработчиком данных для выполнения операций с таблицами;

– обработчик данных взаимодействует с базой данных и файлами данных;

– обработчик поиска и обработчик анализа взаимодействуют с файлами данных для выполнения своих функций.

Для реализации функционала управления данными, был разработан алгоритм, изображенный на рисунке 12.

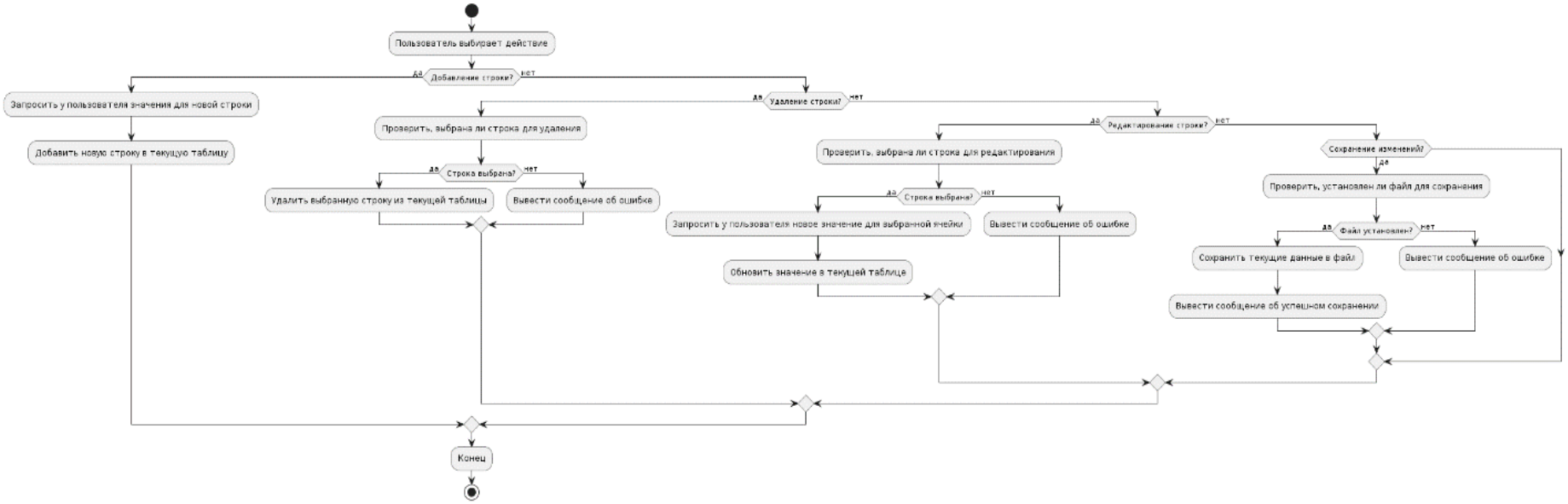


Рисунок 12 – Алгоритм для реализации управления данными

Этот алгоритм охватывает основные операции по управлению данными:

а) начало: пользователь выбирает действие (добавление, удаление, редактирование строки или сохранение изменений).

б) добавление строки:

– запрашиваются значения для новой строки у пользователя;

– новая строка добавляется в текущую таблицу.

в) Удаление строки:

– проверяется, выбрана ли строка для удаления;

– если строка выбрана, она удаляется из текущей таблицы;

– если строка не выбрана, выводится сообщение об ошибке.

г) редактирование строки:

– проверяется, выбрана ли строка для редактирования;

– если строка выбрана, запрашивается новое значение для выбранной ячейки;

– значение в текущей таблице обновляется;

– если строка не выбрана, выводится сообщение об ошибке.

д) сохранение изменений:

– проверяется, установлен ли файл для сохранения;

– если файл установлен, текущие данные сохраняются в файл;

– выводится сообщение об успешном сохранении;

– если файл не установлен, выводится сообщение об ошибке.

е) конец: завершение процесса.

Для связи между таблицами приложения и csv файлами, используются методы, описанные в приложении А:

а) «create_tables»: этот метод создает таблицы для различных категорий (авторы, книги, читатели, выдачи) и загружает в них данные из соответствующих CSV файлов.

Функции данного метода:

− создание таблицы для авторов, книг, читателей, выдачи;

− загрузка данных в таблицу авторов, книг, читателей, выдачи из csv файла;

− отображение таблиц по умолчанию при запуске программы.

б) «create_treeview»: этот метод создает виджет «Treeview» с указанными столбцами. Функции данного метода:

− создание объекта «Treeview» с заданными столбцами;

− конфигурирование каждого столбца;

− установка заголовка столбца;

− установка ширины столбца;

− привязка события двойного щелчка мыши к методу «on_double_click»;

− возвращение созданного объекта «Treeview».

в) «load_data_to_treeview». Этот метод загружает данные из CSV файла в виджет «Treeview». Функции данного метода:

− загрузка данных из файла csv;

− вставка данных из объекта «DataFrame» в объект «Treeview»;

− сохранение имени файла для последующего использования.

Методы для отображения таблиц: «show_authors_table», «show_books_table», «show_readers_table», «show_issues_table» − отображают таблицы: «Выдачи», «Книги», «Читатели», «Авторы».

Благодаря этим методам, можно обеспечить загрузку данных из файлов CSV и их отображение в таблицах интерфейса приложения.

Внешний вид основной вкладки приложения, представлен на рисунке 13.

На данном рисунке отображено главное окно приложения. Пользователь находится во вкладке «Управление», имеется возможность переключаться между таблицами. В каждой таблице находится окно поиска, кнопки «Добавить строку», «Удалить строку», «Сохранить изменения».

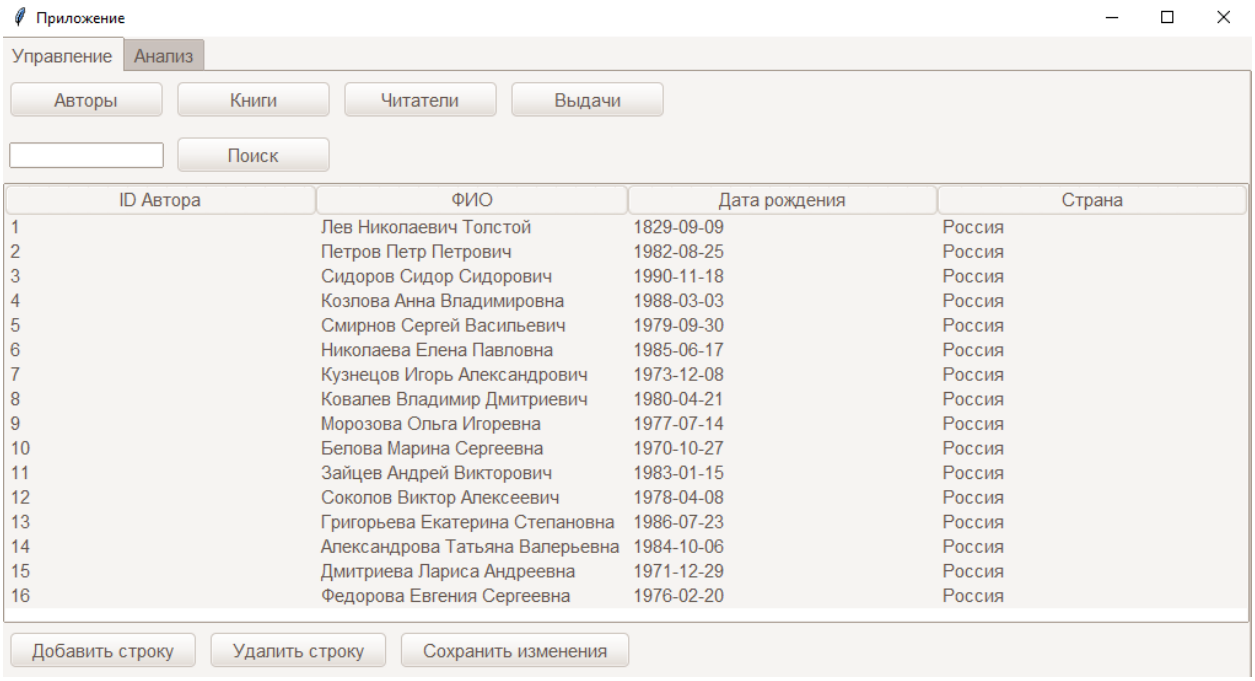


Рисунок 13 – Главная вкладка приложения

Для реализации анализа данных в приложении, был разработан алгоритм, изображенный на рисунке 14.

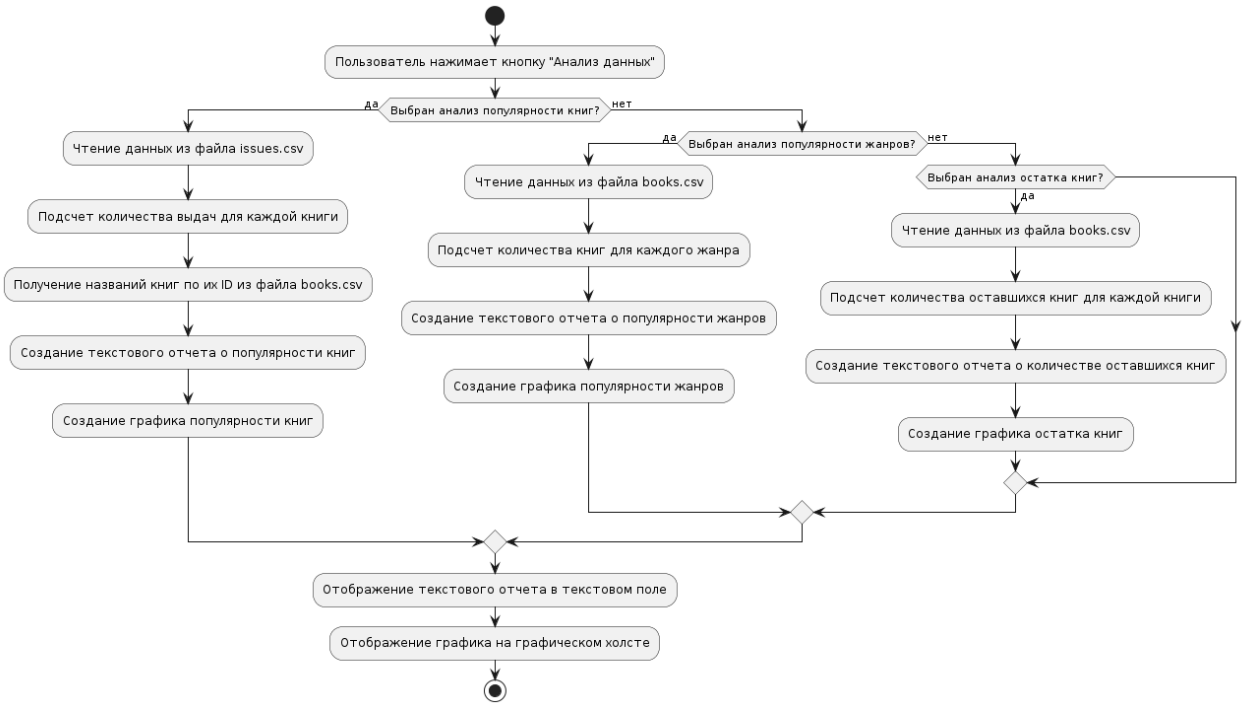


Рисунок 14 – Алгоритм для анализа данных

Этот алгоритм описывает весь процесс анализа данных и отображения результатов в приложении.

а) начало: пользователь нажимает кнопку «Анализ данных».

б) выбор анализа: проверяется, какой тип анализа выбрал пользователь (популярность книг, популярность жанров или остаток книг).

Если выбран анализ популярности книг:

– чтение данных из файла issues.csv;

– подсчет количества выдач для каждой книги;

– получение названий книг по их ID из файла books.csv;

– создание текстового отчета о популярности книг;

– создание графика популярности книг.

Если выбран анализ популярности жанров:

– чтение данных из файла books.csv;

– подсчет количества книг для каждого жанра;

– создание текстового отчета о популярности жанров;

– создание графика популярности жанров.

Если выбран анализ остатка книг:

– чтение данных из файла books.csv;

– подсчет количества оставшихся книг для каждой книги;

– создание текстового отчета о количестве оставшихся книг;

– создание графика остатка книг.

в) отображение результатов:

– отображение текстового отчета в текстовом поле;

– отображение графика на графическом холсте.

На основе данного алгоритма разработаны методы, отвечающие за текстовый и графический варианты представления анализа, изображенных на рисунке 15 и на рисунке 16.

Данные методы позволяют пользователю увидеть результаты анализа. Всего было добавлено 3 варианта анализа: «Анализ популярности книг», «Анализ популярности авторов», «Анализ остатка книг».

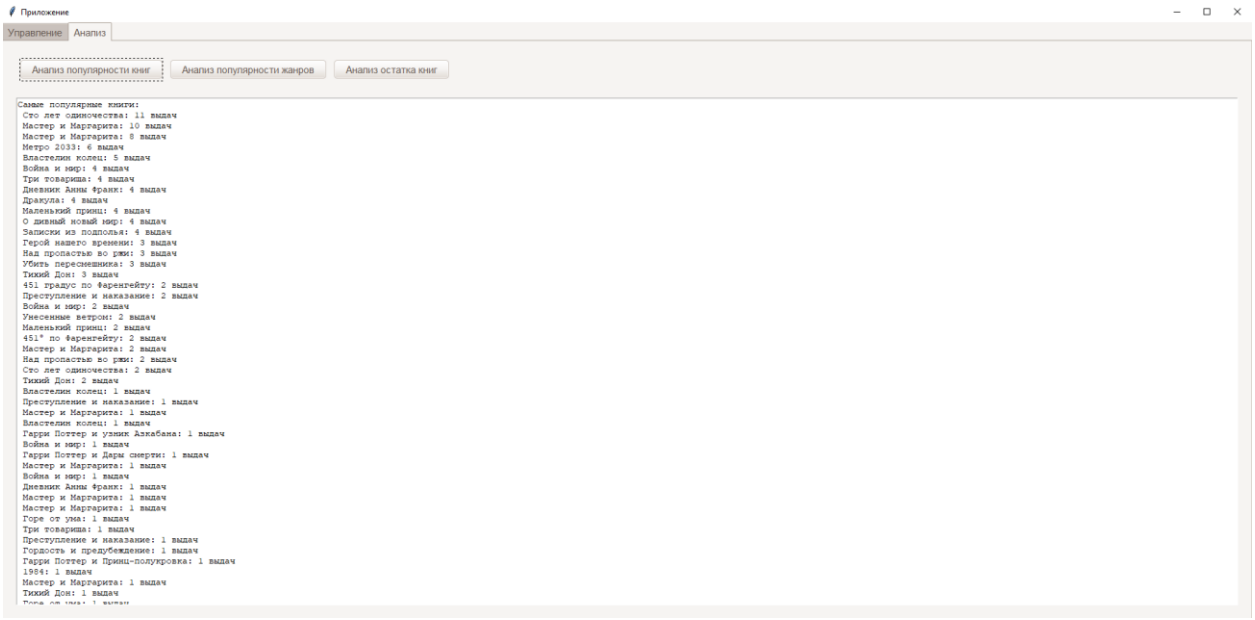


Рисунок 15 – Текстовое представление анализа

Ниже изображен рисунок, отображающий результат работы анализа данных в приложении на примере популярности жанров.

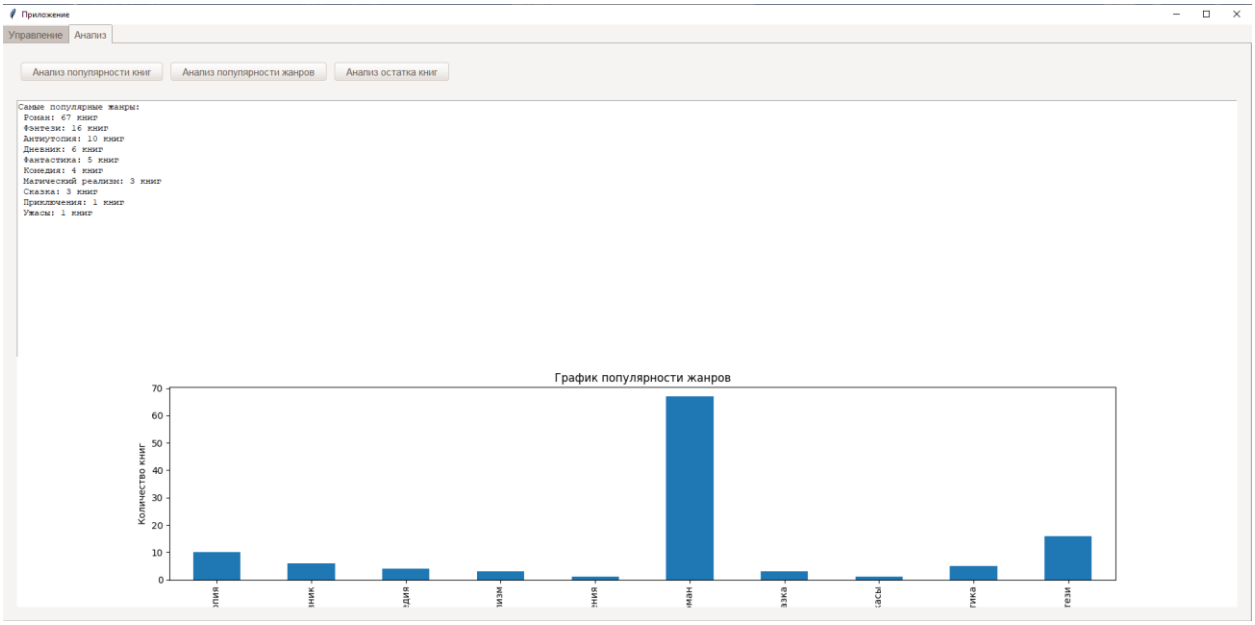


Рисунок 16 – Графическое представление анализа

Методы, представленные в приложении А:

а) метод «analyze_popularity_books» анализирует популярность книг на основе данных из файла issues.csv и выводит результаты в текстовый и графический варианты. Этот метод обладает следующими функциями:

− очистка текстового поля;

− чтение данных из CSV файла;

− подсчет количества выдач для каждой книги;

− получение названия книги по ID;

− добавление информации о книге в текстовый вывод;

− вывод текста в текстовое поле;

− показ графика.

б) метод «analyze_popularity_genres» анализирует популярность жанров на основе данных из файла books.csv и выводит результаты в текстовый и графический варианты. Функции данного метода:

− очистка текстового поля;

− чтение данных из CSV файла;

− подсчет количества книг для каждого жанра;

− добавление информации о жанре в текстовый вывод;

− вывод текста в текстовое поле;

− показ графика.

в) метод «analyze_books_left» анализирует остаток книг на основе данных из файла books.csv и выводит результаты в текстовый и графический варианты. Функции этого метода:

− очистка текстового поля;

− чтение данных из CSV файла;

− добавление информации о книге в текстовый вывод;

− вывод текста в текстовое поле;

− показ графика.

г) метод «show_graph» создаёт график на основе переданных данных и отображает его во вкладке «Анализ». Этот метод отвечает за следующие функции:

− создание новой фигуры для графика;

− построение графика;

− установка заголовка графика;

− установка подписи оси X;

− установка подписи оси Y;

− отрисовка графика;

− отображение графика в интерфейсе.

Таким образом был реализован основной функционал приложения. Для отображения ошибок, были созданы дополнительные методы, отвечающие за обработку исключений:

а) «load_data_to_treeview»:

− ошибка: «FileNotFoundError», если файл не найден;

− ошибка: «EmptyDataError», если файл не содержит данных;

− ошибка: «ParserError, если происходит ошибка при чтении файла.

б) «replace_codes»:

− ошибка: «FileNotFoundError», если файл не найден;

− ошибка: «EmptyDataError», если файл не содержит данных;

− ошибка: «ParserError», если происходит ошибка при чтении файла.

Выводы второй главы:

− описаны функциональные и нефункциональные требования к разрабатываемому приложению;

− разработан макет графического интерфейса, на основе которого был создан основной графический интерфейс приложения;

− разработано собственное приложение для анализа и управления большими данными в библиотеке.

Внедрение разработанного приложения для управления и анализа данных в библиотеку принесет следующие положительные результаты:

− автоматизация рутинных процессов, таких как регистрация новых книг и читателей, выдача и возврат книг;

− быстрый и удобный поиск информации по различным критериям, таким как автор, название книги, жанр и т.д., что облегчает работу с большим объемом данных;

− возможность генерации отчетов и анализа данных позволяет принимать более обоснованные решения на основе статистики и аналитики;

− сокращение бумажного документооборота и переход на электронные записи снижает затраты на материалы и хранение;

− хранение данных в базе данных уменьшает риск потери информации и улучшает безопасность данных.

ЗАКЛЮЧЕНИЕ

В процессе выполнения выпускной квалификационной работы, мной были изучены теоретические основы создания приложений для управления и анализа данных в библиотечном секторе.

В первой главе работы проведен анализ существующих инструментов для реализации приложения, что позволило выявить функциональные требования. Рассмотрено понятие «большие данные» и «базы данных», описаны выбранные инструменты разработки и область их применения.

Во второй главе разработано приложение для управления библиотечными данными. Проведено проектирование дизайна интерфейса с учетом требований пользователей и современных технологий дизайна графического интерфейса. Разработана база данных, которая обеспечивает хранение и управление информацией о книгах, авторах, читателях и выдачах.

Был разработан функционал приложения что позволило обеспечить взаимодействие с базой данных, анализ и управление данными. В результате работы создано полноценное приложение для управления библиотечными данными, которое предоставляет пользователям удобный интерфейс, позволяет осуществлять поиск, добавление, редактирование и удаление информации о книгах, авторах, читателях и выдачах, а также проводить анализ данных и визуализировать результаты в графическом виде. Разработанное приложение соответствует современным стандартам для приложений этой сферы и может успешно использоваться в практических условиях. Внедрение данного приложения в библиотеку приведет к значительному улучшению эффективности управления библиотечными процессами, повысит эффективность работы библиотеки, качество обслуживания пользователей, позволит сэкономить ресурсы.

Выполнение выпускной работы позволило расширить знания о создании и сопровождении приложений для управления и анализа данных в

библиотечном секторе, а также приобрести навыки разработки приложении для анализа и управления данных на Python.

Результаты исследования представлены на научных мероприятиях:

1. III Всероссийский молодежный научный форум «Современное педагогическое образование: теоретический и прикладной аспекты», ЛПИ – филиал СФУ

2. VII Всероссийская научно-практическая конференция преподавателей, учителей, студентов и молодых ученых «Актуальные проблемы преподавателей дисциплин естественнонаучного цикла», ЛПИ – филиал СФУ

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Альсведи, Б. Python. Учебник для начинающих / Б. Альсведи. – ДМК Пресс, 2017. – 496 с.

2. Аммар, А. Что такое обработка данных и почему это важно? Полное руководство / URL: https://www.astera.com/ru/type/blog/what-is-data-munging/ (дата обращения: 17.02.2024).

3. Бейдер, Л. Питон. Чистый код / Л. Бейдер. – ДМК Пресс, 2018. – 484 с.

4. Библиотеки Python. Большая подборка с описанием / URL: https://bangbangeducation.ru/point/razrabotka/biblioteki-dlya-python (дата обращения: 20.02.2024).

5. Блейз, Д. Программирование на Python для начинающих / Д. Блейз. – Питер, 2017. – 400 с.

6. Бэнкс, Дж. Основы программирования на Python / Дж. Бэнкс. – Питер, 2017. – 320 с.

7. Ван Россум, Г. Дрейк, Ф. Python Tutorial / Г. Ван Россум, Ф. Дрейк. – Создатель Python, 2016. – 147 с.

8. Вандерплас, Д. Python для сложных анализов данных / Д. Вандерплас. – ДМК Пресс, 2017. – 548 с.

9. Волкманн, М. Svelte и Sapper в действии / М. Волкманн. – Питер, 2022. – 512 с.

10. Груздев, А. Изучаем pandas. Высокопроизводительная обработка и анализ данных в Python / А. Груздев. – ДМК Пресс, 2019. – 684 с.

11. Дауни, А. Как думать как программист. Изучаем Python / А. Дауни. – Питер, 2016. – 256 с.

12. Иопа, Н. И. Информатика. Конспект лекций. Учебное пособие / Н. И. Иопа. – Питер, 2016. – 193 с.

13. Картер, Д. Python Библиотеки / Д. Картер. – Питер, 2024. – 365 с.

14. Кеннеди, Б. Основы Python для Data Science / Б. Кеннеди. – Питер,2023. – 272 с.

15. Крейг, Д. Искусство программирования на Python / Д. Крейг. – ДМК Пресс, 2020. – 416 с.

16. Ланцерони, М. Python. Практика программирования / М. Ланцерони. – Символ-Плюс, 2022. – 384 с.

17. Лутц, М. Изучаем Python / М. Лутц. – ДМК Пресс, 2019. – 1600 с.

18. Маккини, У. Python для анализа данных / У. Маккини. – ДМК Пресс, 2018. – 544 с.

19. Марк, С. Python. Эффективное программирование / С. Марк. – ДМК Пресс, 2017. – 432 с.

20. Мюллер, А. Введение в машинное обучение с Python / А. Мюллер. – ДМК Пресс, 2017. – 400 с.

21. Рамадильо, К. Python. К вершинам мастерства / К. Рамадильо. – Питер, 2017. – 792 с.

22. Рамалхо, Л. Python. Карманный справочник / Л. Рамалхо. – Вильямс, 2018. – 792 с. – С. 215.

23. Расмуссен, М. Python и анализ данных / М. Расмуссен. – Питер, 2018. – 320 с.

24. Ротеринд, Л. Python. К вершинам мастерства / Л. Ротеринд. – Питер, 2017. – 792 с.

25. Саммерфилд, М. Python. Эффективное программирование. Том 1 / М. Саммерфилд. – Питер, 2022. – 736 с.

26. СУБД PostgreSQL: почему её стоит выбрать для работы с данными и как установить / URL: https://practicum.yandex.ru/blog/chto-takoe-subd-postgresql (дата обращения: 24.02.2024)

27. Тейлор, Д. Автоматизация задач с Python / Д. Тейлор. – Вильямс, 2020. – 352 с.

28. Тузиков, А. В. Цифровая трансформация. Основные понятия и терминология / А. В. Тузиков. – Минск, 2020. – 267 с.

29. Труханов, Д. Python и анализ данных. Обработка данных с использованием Pandas / Д. Труханов. – Питер, 2020. – 320 с.

30. Хетланд, М. Программирование на Python / М. Хетланд. – Символ-Плюс, 2019. – 480 с.

31. Что такое база данных. Oracle СНГ / URL: https://www.oracle.com/cis/database/what-is-database (дата обращения: 20.02.2024).

32. Что такое Elasticsearch: история, архитектура, применение в Big Data / URL: https://bigdataschool.ru/wiki/elasticsearch (дата обращения: 11.01.2024)

33. Что такое Tkinter. Создание графического интерфейса на Python / URL: https://younglinux.info/tkinter/tkinter (дата обращения: 18.02.2024).

34. Шавиш, М. Python для анализа данных. Эффективная обработка информации / М. Шавиш. – ДМК Пресс, 2021. – 416 с.

35. Шапиро, Л. Программирование на Python. Глубокое погружение / Л. Шапиро. – Питер, 2019. – 528 с.

36. CSV формат файла: описание, использование, представление / URL: https://topexpert.digital/wiki/csv/ (дата обращения: 03.01.2024)

37. MongoDB: что это, для чего используется, особенности / URL: https://itglobal.com/ru-ru/company/glossary/mongodb/ (дата обращения: 24.02.2024)

38. Python. Глоссарий. / URL: https://blog.skillfactory.ru/glossary/python (дата обращения: 18.02.2024).

39. Redis – как работает и зачем нужна база данных / URL: https://selectel.ru/blog/how-redis-works/ (дата обращения: 24.02.2024)

40. SQLLite – самая простая база данных, которая работает везде / URL: https://thecode.media/sqlite (дата обращения: 20.02.2024).

ПРИЛОЖЕНИЕ А

Код приложения

```
import tkinter as tk
from tkinter import ttk, filedialog, messagebox, simpledialog
import pandas as pd
from ttkthemes import ThemedTk
import matplotlib.pyplot as plt
from matplotlib.backends.backend_tkagg import FigureCanvasTkAgg
class ManagementWindow:
    def __init__(self, root):
        self.root = root
        self.root.title("Приложение")
        self.notebook = ttk.Notebook(self.root)
        self.notebook.pack(fill="both", expand=True)
        self.management_tab = ttk.Frame(self.notebook)
        self.notebook.add(self.management_tab, text="Управление")
        self.category_frame = ttk.Frame(self.management_tab)
        self.category_frame.pack(side="top", fill="x")
        self.create_category_buttons()
        self.main_frame = ttk.Frame(self.management_tab)
        self.main_frame.pack(side="top", fill="both", expand=True)
        self.search_frame = ttk.Frame(self.main_frame)
        self.search_frame.pack(side="top", fill="x", pady=5)
        self.search_var = tk.StringVar()
        self.search_entry = ttk.Entry(self.search_frame, textvariable=self.search_var)
        self.search_entry.pack(side="left", padx=5)
        self.search_button = ttk.Button(self.search_frame, text="Поиск",
command=self.search_data)
        self.search_button.pack(side="left", padx=5)
```

```
        self.table_frame = ttk.Frame(self.main_frame)
        self.table_frame.pack(side="top", fill="both", expand=True)
        self.create_tables()
        self.analysis_tab = ttk.Frame(self.notebook)
        self.notebook.add(self.analysis_tab, text="Анализ")
        self.create_analysis_widgets()
        self.canvas = None
        self.button_frame = ttk.Frame(self.main_frame)
        self.button_frame.pack(side="bottom", fill="x", pady=5)
        self.add_row_button = ttk.Button(self.button_frame, text="Добавить
строку", command=self.add_row)
        self.add_row_button.pack(side="left", padx=5)
        self.delete_row_button = ttk.Button(self.button_frame, text="Удалить
строку", command=self.delete_row)
        self.delete_row_button.pack(side="left", padx=5)
        self.save_changes_button = ttk.Button(self.button_frame, text="Сохранить
изменения", command=self.save_changes)
        self.save_changes_button.pack(side="left", padx=5)
    def create_category_buttons(self):
        self.authors_button = ttk.Button(self.category_frame, text="Авторы",
command=self.show_authors_table)
        self.authors_button.pack(side="left", padx=5, pady=5)
        self.books_button = ttk.Button(self.category_frame, text="Книги",
command=self.show_books_table)
        self.books_button.pack(side="left", padx=5, pady=5)
        self.readers_button = ttk.Button(self.category_frame, text="Читатели",
command=self.show_readers_table)
        self.readers_button.pack(side="left", padx=5, pady=5)
        self.issues_button = ttk.Button(self.category_frame, text="Выдачи",
command=self.show_issues_table)
```

```
        self.issues_button.pack(side="left", padx=5, pady=5)
    def create_tables(self):
        self.authors_treeview = self.create_treeview(self.table_frame, ["ID Автора",
"ФИО", "Дата рождения", "Страна"])
        self.load_data_to_treeview(self.authors_treeview, "authors.csv")
        self.books_treeview = self.create_treeview(self.table_frame, ["ID Книги",
"Название", "Автор", "Год издания", "ISBN", "Жанр", "Количество в
библиотеке", "Описание"])
        self.load_data_to_treeview(self.books_treeview, "books.csv")
        self.readers_treeview = self.create_treeview(self.table_frame, ["ID
Читателя", "ФИО", "Адрес", "Email", "Номер телефона"])
        self.load_data_to_treeview(self.readers_treeview, "readers.csv")
        self.issues_treeview = self.create_treeview(self.table_frame, ["ID Выдачи",
"ID Книги", "ID Читателя", "Дата выдачи", "Дата возврата", "Читательский
билет"])
        self.load_data_to_treeview(self.issues_treeview, "issues.csv")
        self.show_authors_table()
    def create_treeview(self, parent, columns):
        treeview = ttk.Treeview(parent, columns=columns, show='headings')
        for col in columns:
            treeview.heading(col, text=col)
            treeview.column(col, width=100)
        treeview.bind("<Double-1>", self.on_double_click)
        return treeview
    def load_data_to_treeview(self, treeview, filename):
        try:
            df = pd.read_csv(filename)
            for _, row in df.iterrows():
                treeview.insert("", "end", values=tuple(row))
            treeview.filename = filename
```

```
            except FileNotFoundError:
                messagebox.showwarning("Ошибка", f"Файл {filename} не найден!")
            except pd.errors.EmptyDataError:
                messagebox.showwarning("Ошибка", f"В файле {filename} не найдено данных!")
            except pd.errors.ParserError:
                messagebox.showerror("Ошибка", f"Ошибка чтения файла {filename}!")
    def show_authors_table(self):
        self.hide_all_tables()
        self.authors_treeview.pack(fill="both", expand=True)
        self.current_treeview = self.authors_treeview
    def show_books_table(self):
        self.hide_all_tables()
        self.books_treeview.pack(fill="both", expand=True)
        self.current_treeview = self.books_treeview
    def show_readers_table(self):
        self.hide_all_tables()
        self.readers_treeview.pack(fill="both", expand=True)
        self.current_treeview = self.readers_treeview
    def show_issues_table(self):
        self.hide_all_tables()
        self.issues_treeview.pack(fill="both", expand=True)
        self.current_treeview = self.issues_treeview
    def hide_all_tables(self):
        for child in self.table_frame.winfo_children():
            child.pack_forget()
    def create_analysis_widgets(self):
        self.analysis_frame = ttk.Frame(self.analysis_tab)
        self.analysis_frame.pack(fill="both", expand=True, padx=10, pady=10)
        self.analysis_button_frame = ttk.Frame(self.analysis_frame)
```

```
        self.analysis_button_frame.pack(fill="x", expand=False, padx=10, pady=10)
        self.popularity_books_button = ttk.Button(self.analysis_button_frame,
text="Анализ популярности книг", command=self.analyze_popularity_books)
        self.popularity_books_button.pack(side="left", padx=5, pady=5)
        self.popularity_genres_button = ttk.Button(self.analysis_button_frame,
text="Анализ популярности жанров",
command=self.analyze_popularity_genres)
        self.popularity_genres_button.pack(side="left", padx=5, pady=5)
        self.books_left_button = ttk.Button(self.analysis_button_frame, text="Анализ
остатка книг", command=self.analyze_books_left)
        self.books_left_button.pack(side="left", padx=5, pady=5)
        self.analysis_output_frame = ttk.Frame(self.analysis_frame)
        self.analysis_output_frame.pack(fill="both", expand=True, padx=10,
pady=10)
        self.analysis_output_text = tk.Text(self.analysis_output_frame)
        self.analysis_output_text.pack(fill="both", expand=True)
    def analyze_popularity_books(self):
        self.analysis_output_text.delete(1.0, "end")
        df = pd.read_csv("issues.csv")
        book_counts = df["BookID"].value_counts()
        output = "Самые популярные книги:\n"
        for book_id, count in book_counts.items():
            book_title = self.get_book_title(book_id)
            if book_title != "Unknown Title":
                output += f"{book_title}: {count} выдач\n"
        self.analysis_output_text.insert("end", output)
        df["BookID"] = df["BookID"].astype(str)
        df = df.rename(columns={"BookID": "ID"})
        df["BookTitle"] = df["ID"].apply(lambda x: self.get_book_title(x))
        df = df.groupby("BookTitle")["ID"].count().reset_index(name="Count")
```

```
        df = df[df["BookTitle"] != "Unknown Title"]
        self.show_graph(df=df, kind="bar", x="ID", y="Count", title="График популярности книг", xlabel="ID книги", ylabel="Количество выдач")
    def analyze_popularity_genres(self):
        self.analysis_output_text.delete(1.0, "end")
        df = pd.read_csv("books.csv")
        genre_counts = df["Genre"].value_counts()
        output = "Самые популярные жанры:\n"
        for genre, count in genre_counts.items():
            output += f"{genre}: {count} книг\n"
        self.analysis_output_text.insert("end", output)
        df = df.groupby("Genre")["ID"].count().reset_index(name="Count")
        self.show_graph(df=df, kind="bar", x="Genre", y="Count", title="График популярности жанров", xlabel="Жанр", ylabel="Количество книг")
    def analyze_books_left(self):
        self.analysis_output_text.delete(1.0, "end")
        df = pd.read_csv("books.csv")
        df = df.drop_duplicates(subset=['ID'])
        output = "Количество оставшихся книг:\n"
        for _, row in df.iterrows():
            output += f"ID {row['ID']} - {row['Title']}: {row['Quantity']} книг\n"
        self.analysis_output_text.insert("end", output)
        df = df.rename(columns={"ID": "BookID", "Quantity": "Count"})
        self.show_graph(df=df, kind="bar", x="BookID", y="Count", title="График остатка книг", xlabel="ID книги", ylabel="Остаток книг")
    def get_book_title(self, book_id):
        df = pd.read_csv("books.csv")
        book_row = df[df["ID"] == int(book_id)]
        if not book_row.empty:
            return book_row.iloc[0]["Title"]
```

```
        else:
            return "Unknown Title"
    def show_graph(self, df, kind, x, y, title, xlabel, ylabel):
        plt.figure()
        ax = df.plot(kind=kind, x=x, y=y, legend=False)
        ax.set_title(title)
        ax.set_xlabel(xlabel)
        ax.set_ylabel(ylabel)
        if self.canvas:
            self.canvas.get_tk_widget().pack_forget()
        self.canvas = FigureCanvasTkAgg(ax.figure,
master=self.analysis_output_frame)
        self.canvas.draw()
        self.canvas.get_tk_widget().pack(fill="both", expand=True)
    def on_double_click(self, event):
        item = self.current_treeview.selection()[0]
        column = self.current_treeview.identify_column(event.x)
        column_number = int(column[1:]) - 1
        old_value = self.current_treeview.item(item, "values")[column_number]
        new_value = simpledialog.askstring("Редактировать значение", f"Введите
новое значение для {self.current_treeview.heading(column)['text']}:",
initialvalue=old_value)
        if new_value is not None:
            self.current_treeview.set(item, column=column, value=new_value)
    def add_row(self):
        columns = self.current_treeview["columns"]
        new_values = []
        for column in columns:
            new_value = simpledialog.askstring("Добавить значение", f"Введите
значение для {column}:")
```

```
            new_values.append(new_value)
        if all(value is not None for value in new_values):
            self.current_treeview.insert("", "end", values=new_values)
    def delete_row(self):
        selected_items = self.current_treeview.selection()
        if selected_items:
            for item in selected_items:
                self.current_treeview.delete(item)
        else:
            messagebox.showwarning("Ошибка", "Пожалуйста, выберите строку
для удаления.")
    def save_changes(self):
        filename = self.current_treeview.filename
        if not filename:
            messagebox.showerror("Ошибка", "Не удалось определить файл для
сохранения.")
            return
        data = []
        for row_id in self.current_treeview.get_children():
            row = self.current_treeview.item(row_id)["values"]
            data.append(row)
        df = pd.DataFrame(data, columns=self.current_treeview["columns"])
        df.to_csv(filename, index=False)
        messagebox.showinfo("Сохранено", f"Изменения успешно сохранены в
{filename}.")
    def search_data(self):
        search_text = self.search_var.get().lower()
        for child in self.current_treeview.get_children():
            self.current_treeview.delete(child)
        filename = self.current_treeview.filename
```

```
        df = pd.read_csv(filename)
        filtered_df = df[df.apply(lambda row: search_text in
row.astype(str).str.lower().values, axis=1)]
        for _, row in filtered_df.iterrows():
            self.current_treeview.insert("", "end", values=tuple(row))
    def replace_codes(self):
        files = ["authors.csv",
"books.csv",
"readers.csv",
"issues.csv"]
        code_mapping = {"missing": "не указано", "N/A": "не указано"}
        for file in files:
            try:
                df = pd.read_csv(file)
                df.replace(code_mapping, inplace=True)
                df.to_csv(file, index=False)
                print(f"Успешно обработан файл: {file}")  # Диагностическое
сообщение
            except FileNotFoundError:
                messagebox.showwarning("Ошибка", f"Файл {file} не найден!")
            except pd.errors.EmptyDataError:
                messagebox.showwarning("Ошибка", f"В файле {file} не найдено
данных!")
            except pd.errors.ParserError:
                messagebox.showerror("Ошибка", f"Ошибка чтения файла {file}!")
            except Exception as e:
                messagebox.showerror("Ошибка", f"Неизвестная ошибка при
обработке файла {file}: {str(e)}")
                print(f"Неизвестная ошибка при обработке файла {file}: {str(e)}")  #
Диагностическое сообщение
```

```
        messagebox.showinfo("Успех", "Коды успешно заменены во всех
файлах.")
if __name__ == "__main__":
    root = ThemedTk(theme="radiance")
    app = ManagementWindow(root)
    root.mainloop()
```