Министерство образования Республики Беларусь

Учреждение образования «Белорусский государственный университет информатики и радиоэлектроники»

Факультет компьютерного проектирования

Кафедра инженерной психологии и эргономики

Дисциплина Программирование сетевых приложений

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовому проекту

на тему

**РАЗРАБОТКА СИСТЕМЫ АВТОМАТИЗАЦИИ ОЦЕНКИ ОБЪЕКТОВ НЕДВИЖИМОСТИ**

//БГУИР КП 1-40 05 01-12 029 ПЗ

Студент Ключенко Р.О.

Руководитель Петрович Н.О.

Минск 2019

# СОДЕРЖАНИЕ

[СОДЕРЖАНИЕ 2](#_Toc27155966)

[ВВЕДЕНИЕ 4](#_Toc27155967)

[1 ОПИСАНИЕ СИСТЕМЫ УЧЕТА НОВЫХ ПОСТУПЛЕНИЙ В БИБЛИОТЕКУ 6](#_Toc27155968)

[1.1 Основные понятия и определения системы учета библиотеки 6](#_Toc27155969)

[1.2 Автоматизация системы учета книг в библиотеке 6](#_Toc27155970)

[1.3 Автоматизация бизнес-процессов библиотек 7](#_Toc27155971)

[1.4 Аналоги систем учета библиотек 8](#_Toc27155972)

[2 ОБЗОР МЕТОДОВ РЕШЕНИЯ ПОСТАВЛЕННОЙ ЗАДАЧИ 10](#_Toc27155973)

[2.1 Обзор методов решения поставленной задачи 10](#_Toc27155974)

[2.2 Описание основного процесса системы автоматизации оценки объектов недвижимости 11](#_Toc27155975)

[2.3 Описание функционала разрабатываемого приложения 15](#_Toc27155976)

[3 ИНФОРМАЦИОННАЯ МОДЕЛЬ СИСТЕМЫ И ЕЁ ОПИСАНИЕ 16](#_Toc27155977)

[4 ОПИСАНИЕ АЛГОРИТМОВ, РЕАЛИЗУЮЩИХ БИЗНЕС-ЛОГИКУ СЕРВЕРНОЙ ЧАСТИ ПРОЕКТИРУЕМОЙ СИСТЕМЫ 18](#_Toc27155978)

[5 МОДЕЛИ ПРЕДСТАВЛЕНИЯ СИСТЕМЫ И ИХ ОПИСАНИЕ 20](#_Toc27155979)

[6 РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ 23](#_Toc27155980)

[6.1 Работа сервера 23](#_Toc27155981)

[6.2 Работа приложения в режиме администратора 24](#_Toc27155982)

[6.3 Работа приложения в режиме клиента 27](#_Toc27155983)

[ЗАКЛЮЧЕНИЕ 34](#_Toc27155984)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 35](#_Toc27155985)

[ПРИЛОЖЕНИЕ А 36](#_Toc27155986)

[(Обязательное) 36](#_Toc27155987)

[ПРИЛОЖЕНИЕ Б 37](#_Toc27155988)

[(Обязательное) 37](#_Toc27155989)

[ПРИЛОЖЕНИЕ В 38](#_Toc27155990)

[(Обязательное) 38](#_Toc27155991)

[ПРИЛОЖЕНИЕ Г 39](#_Toc27155992)

[(Обязательное) 39](#_Toc27155993)

[ПРИЛОЖЕНИЕ Д 40](#_Toc27155994)

[(Обязательное) 40](#_Toc27155995)

[40](#_Toc27155996)

[ПРИЛОЖЕНИЕ Е 41](#_Toc27155997)

[(Обязательное) 41](#_Toc27155998)

[ПРИЛОЖЕНИЕ Ж 42](#_Toc27155999)

[(Обязательное) 42](#_Toc27156000)

# ВВЕДЕНИЕ

В данной работе рассматривается разработка программного приложения, которое позволит оптимизировать и автоматизировать систему учета новых поступлений в библиотеку.

Система учета книг – неотъемлемая часть в каждой библиотеке. Она позволяет хранить данные о книгах, которые хранятся, а именно: их название, автор, количество в библиотеке, цена и другое. Можно с уверенностью сказать, что эффективность работы библиотеки зависит от эффективности и доступности ее системы учета. Возьмем в пример заказ новых книг в библиотеку. Для этого нужно найти поставщиков, которые будут предоставлять каталог книг. Затем нужно определить, какие книги уже есть в библиотеке, в каком количестве. Нужно иметь к сведению эту информацию для наиболее выгодного для библиотеки нового заказа книг.

В связи с этим, появилась необходимость в автоматизации этой системы. Необходимо реализовать такое приложение, которое позволит:

* хранить информацию, связанную с информацией о книгах, пользователях и заказах;
* реализовать добавление новых заказов и новых книг, которые в последующем будут доступны библиотекам;
* производить отчётность в виде графиков и документов с необходимой информацией;
* увеличить клиентскую базу поставщиков за счёт оптимизации функции заказов и упростить поиск новых книг библиотекам.

Таким образом, автоматизация системы учета новых поступлений в библиотеку будет решать сразу две основные проблемы: 1) будет уменьшать время, затраченное на оформление нового заказа. 2) будет помогать с наиболее выгодным оформлением заказа, так как появится возможность просмотра и поиска книг в структурированном виде.

Эта автоматизированная система значительно упростит работу сотрудников библиотеки. Более того, поставщику книг также будет удобнее отслеживать заказы и хранить данные о них.

*Объектом исследования для данной курсовой работы является* система учета книг в библиотеке

*Предметом исследования является* автоматизация системы учета

*Целью данного курсового проекта* является

Работа над программным проектом проходит следующие этапы работы:

– рассмотреть существующие аналоги приложений или сайтов по системе учета книг, покупкеновых книг;

– провести анализ ътих систем на предмет недостатков;

– разработать функциональную модель основного процесса;

– разработать приложение системы учета;

– провести тестирование разработанного программного продукта;

– выполнить отладку ошибок, обнаруженных в процессе тестирования.

# 1 ОПИСАНИЕ СИСТЕМЫ УЧЕТА НОВЫХ ПОСТУПЛЕНИЙ В БИБЛИОТЕКУ

## Основные понятия и определения системы учета библиотеки

Библиотека – учреждение, собирающее и хранящее книги для общественного пользования.

Библиотечный фонд – основной компонент учреждения, собрание множества документов на различных носителях. От его содержания и качества зависит степень удовлетворенности и развития информационных потребностей пользователей.

Инновационные процессы, происходящие в последние десятилетия, повлияли на традиционные аспекты работы с фондом. Развитие информационных технологий побуждает библиотекарей искать новые подходы к его комплектованию, организации учета и обеспечению сохранности.

Формирование и использование библиотечного фонда является непрерывным процессом. Работа по формированию фонда – основая деятельность библиотечных работников, совокупность многих действий (рис.1).



Рис.1 Формирование и использование библиотечного фонда

Комплектование – важнейшее направление в формировании фонда библиотеки, обеспечивающее книгами пользователей библиотеки.

Основной источник комплектования библиотек – покупка новых книг у различных поставщиков.

Учет фонда библиотеки – это комплекс операций, обеспечивающих фиксацию сведений о величине, составе и движении фонда по установленным правилам.

Система учета библиотеки – упорядоченная совокупность видов, методов и способов учета книг.

Целью учета является получение сведений и данных о(об):

– количестве и стоимости поступающей и выбывающей литературы;

– общеей численности фонда;

– стоимости фонда;

– составе фонда по содержанию;

Решение о том, требуется ли оформлять заказ на покупку новых книг и каких, если требуется, принимается сотрудниками библиотеки на основании анализа учета книг в библиотеке. [1]

## Автоматизация системы учета книг в библиотеке

Автоматизация библиотечного дела призвана дать сотрудникам библиотек удобный инструмент для работы с библиотечным фондом и избавить их от рутинной работы, а читателям — предоставить эффективный и комфортный сервис в поиске и получении книг и изданий. При этом важно найти гармоничное сочетание традиционных и инновационных форм работы, грамотно определить последовательность и возможности практической реализации новых информационных технологий в этой сфере.

Многие библиотеки перевели и переводят свои каталоги в электронный вид. С помощью библиотечных информационных систем ведётся систематизация имеющихся изданий и автоматизированный учёт книжного фонда. Организуется удобный поиск и заказ нужной информации, доступ к ней. Автоматизируются и другие библиотечные функции. При выборе той или иной библиотечной информационной системы решающим фактором являются финансовые возможности библиотек. Приобретая такую систему, необходимо учитывать совокупную стоимость владения, то есть затраты на техническую поддержку, обновление версий системы, обучение персонала, плату за внесение изменений и доработку системы в соответствии с требованиями конкретной библиотеки.

## Автоматизация бизнес-процессов библиотек

**1. Комплектование и заказ литературы в издательствах**

Информацию для заказа книг и журналов библиотекам сейчас нужно искать либо по прайс-листам, либо в Интернете, либо на специализированных книжных выставках. Автоматизированная система должна включать в себя возможность просмотра каталога книг у поставщика для дальнейшего их заказа.

**2. Первичная обработка поступившего в библиотеку издания**

При поступлении какого-либо издания специалист-каталогизатор описывает полученное издание, в определенных полях библиографической записи указывает параметры книги: её автора, название, тип, количество книг, цену и т.д.

**3. Создание электронного каталога**

Библиографические описания вновь поступивших изданий сразу заносятся в систему в электронном виде. А те библиографические описания, которые находились на карточках, также переводятся в электронный вид. Электронный каталог обладает большими возможностями для поиска, нежели печатный. Ведь печатный каталог — это, по сути, три каталога — алфавитный, систематический и предметный. Внутри каталога идет систематизация соответственно или по алфавиту, или по рубрикам, либо по предметам и т.д. При использовании информационной системы электронный каталог позволяет читателю значительно быстрее найти требуемое произведение, предоставляя различные варианты поиска..

**4. Организация хранения документов в электронном формате**

Вопрос организации хранения в автоматизированной библиотечной системе — это вопрос хранения документов в электронном формате. Поэтому развитие автоматизированных библиотечных систем возможно только при условии обеспечения единого библиографического пространства: описания всех ресурсов библиотеки (в традиционном и электронном формате) располагаются в едином каталоге, в котором предусмотрены классификация ресурсов по уровню доступа, а также доступ к ресурсам в электронном формате.

Кроме того, библиотечная система должна обеспечивать выдачу электронных документов с учетом разграничения прав доступа читателей. Таким образом, создается электронная библиотека — упорядоченный фонд электронных документов вместе с соответствующими сервисами их предоставления. Этот фонд хранится, актуализируется, систематизируется, одним словом, с ним можно совершать все те же операции, что и с традиционным библиотечным фондом. [1]

## 1.4 Аналоги систем учета книг

На просторах интернета можно найти множество систем учета книг в библиотеке, но принципы работы их не отличаются. Все они хранят данные о кнгах и читателях (рис. 1.4.1). Также можно найти программы дляя заказа книг в библиотеку (рис. 1.4.2). В основном, программы обладают функционалом заказа разных продуктов, а не только книг.

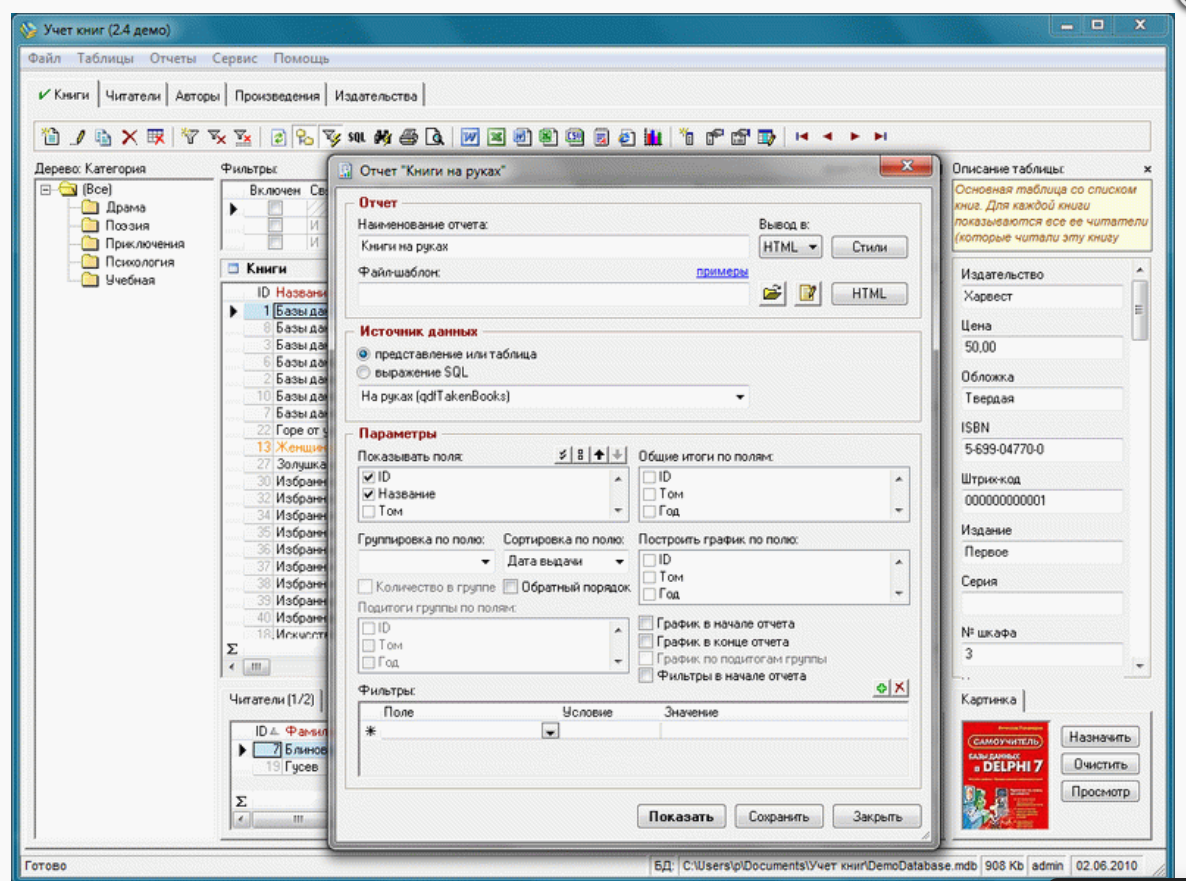


Рисунок 1.4.1 – Пример работы система учета библиотеки

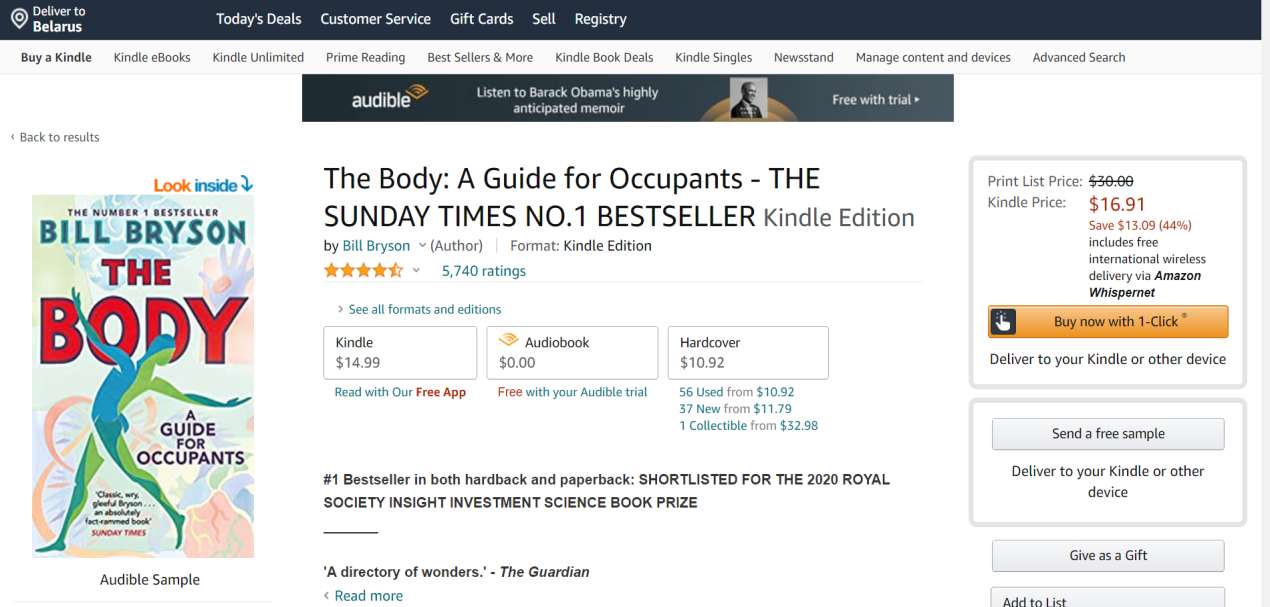


Рисунок 1.4.2 – Пример заказа новых книг

Для работы библиотек не нужна излишняя функциональность. Требуется грамотное и надежное хранение книг и информации о них, возможность поиска по критерию и заказ новых книг, желательно в одном месте. Должен быть удобный просмотр каталога книг у поставщиков, простой в использовании интерфейс.

# 2 ОБЗОР МЕТОДОВ РЕШЕНИЯ ПОСТАВЛЕННОЙ ЗАДАЧИ

## 2.1 Обзор методов решения поставленной задачи

Приложение разработано на языке программирования Java.

Javа – это универсальный компьютерный язык программирования, который является параллельным, объектно-ориентированным и разработанным, чтобы иметь минимум зависимостей реализации. Скомпилированный Jаva – код может отлично работать на всех платформах, поддерживающих Jаva, без необходимости перекомпиляции кода.

Java – так называют не только сам язык, но и платформу для создания приложений уровня предприятий на основе данного языка.

Программы на Java могут быть транслированы в байт-код, выполняемый на виртуальной java-машине (JVM) – программе, обрабатывающей байт-код и передающей инструкции оборудованию, как интерпретатор, но с тем отличием, что байт-код, в отличие от текста, обрабатывается значительно быстрее.

Для реализации программного продукта, разрабатываемого в данном курсовом проекте на языке Jаva, была выбрана среда IntelliJ IDEA – из-за мощного функционала разработки и приятного интерфейса. Эта среда разработки не уступает своим аналогам по функционалу. представляет собой высокотехнологичный комплекс тесно интегрированных инструментов программирования, включающий интеллектуальный редактор исходных текстов с развитыми средствами автоматизации, мощные инструменты рефакторинга кода, встроенную поддержку технологий J2EE, механизмы интеграции со средой тестирования Ant/JUnit и системами управления версиями, уникальный инструмент оптимизации и проверки кода Code Inspection, а также инновационный визуальный конструктор графических интерфейсов.

В данном приложении использована база данных PostgreSQL.

Для создания диаграмм использовалось средство моделирования Draw IO.

Для создания диаграммы классов использовалась интегрированная среда разработки программного обеспечения IntelliJ IDЕA.

## 

## 2.2 Описание основного процесса системы учета новых поступлений в библиотеку

Описание основной работы системы учета новых поступлений осуществляется с помощью контекстных диаграмм и декомпозиций компонентов процесса.

На контекстной диаграмме верхнего уровня (рисунок 2.2.1) представлена функциональная модель «Заказ новых книг», определены потоки входных и выходных данных, механизмы ограничения и управления данными.

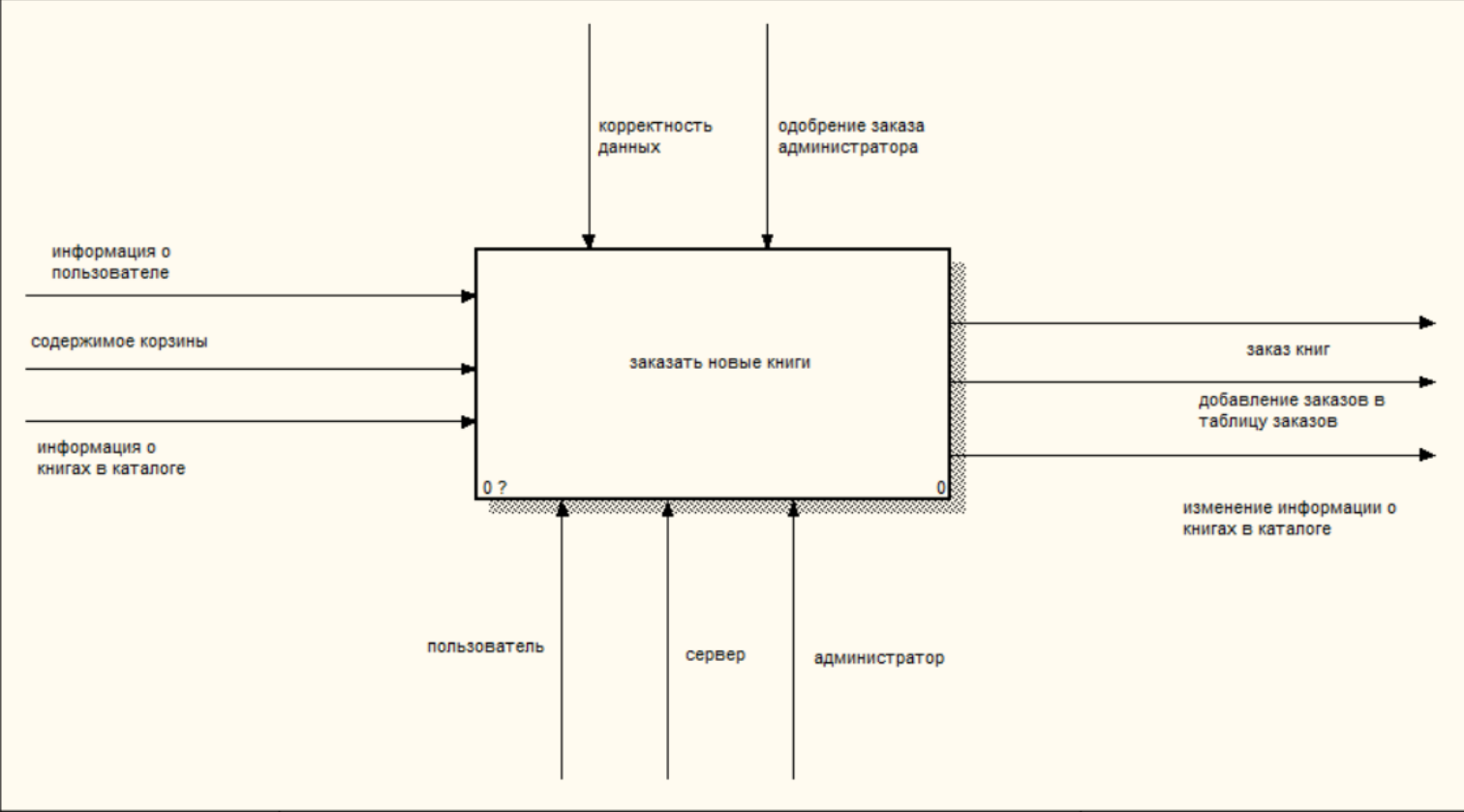


Рисунок 2.2.1 – Контекстная диаграмма верхнего уровня

Декомпозиция диаграммы верхнего уровня содержит четыре компонента.

Первый компонент «Войти в систему» подразумевает, что пользователю нужно авторизоваться в систему. Если пользователь ещё не зарегистрирован в системе, то предварительно следует зарегистрироваться.

Второй компонент «Просмотреть каталог книг» подразумевает, что пользователь может просмотреть каталог доступных для заказа книг. Он может просмотреть информацию о названиях, авторах, типе, цене и количестве книг.

Третий компонент «Добавить книги в корзину» подразумевает, что в данном процессе пользователь выбирает книги, которые он хочет заказать, и указывает желаемое количество.

Четвертый компонент «Оформить заказ» подразумевает, что заказ отправляется в таблицу заказов со статусом заказа «в обработке». Далее администратор может или изменить статус заказа на «одобрено» или «отказано».

Декомпозиция диаграммы базового процесса представлена на рисунке 2.2.2.

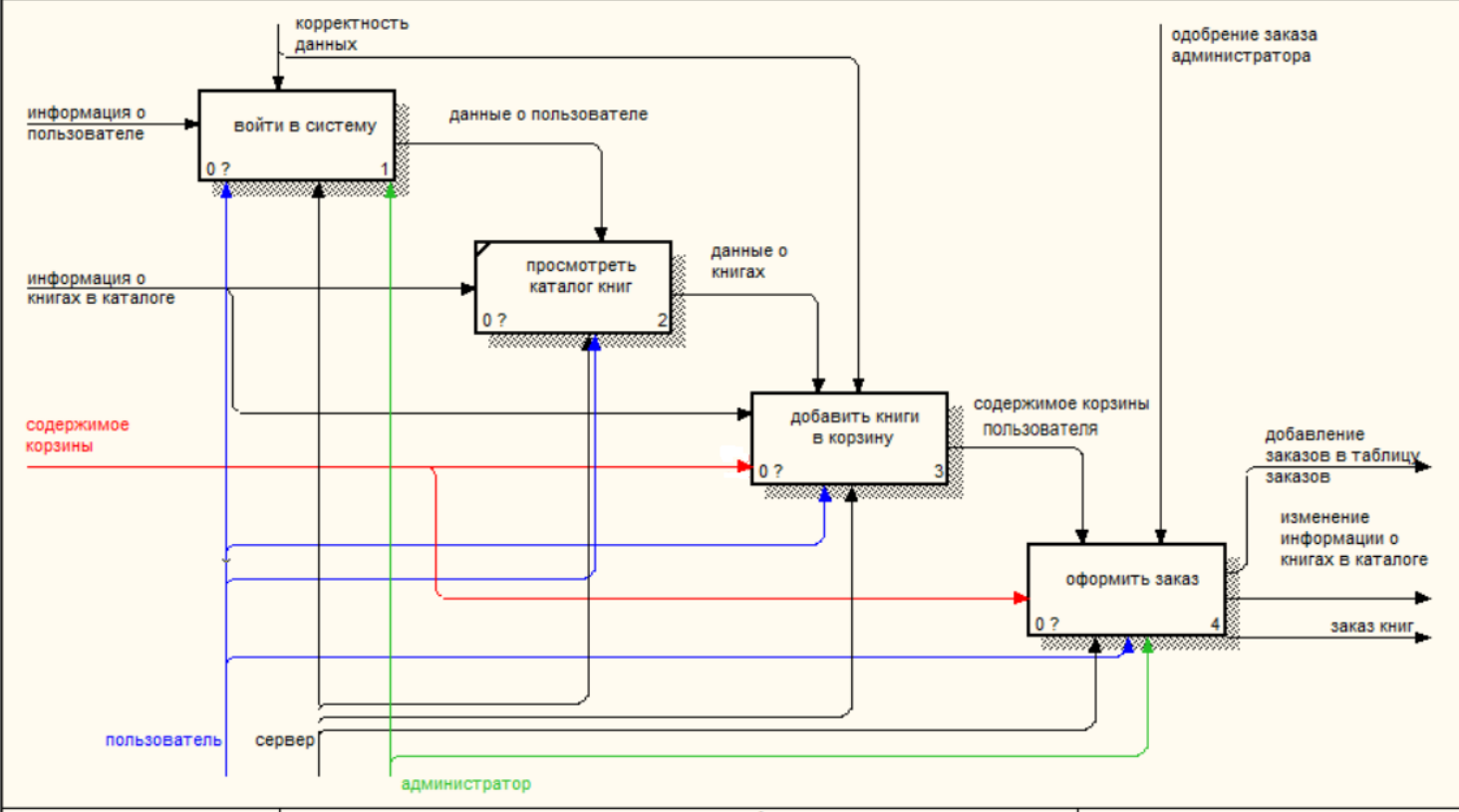


Рисунок 2.2.2 – Декомпозиция диаграммы верхнего уровня

Процесс добавления книг в корзину проходит в два этапа, поэтому декомпозиция блока «Добавить книги в корзину» содержит два компонента.

Первый компонент «Ввести id книг и их количество» подразумевает, что пользователь может выбрать книгу из каталога и указать желаемое количество их штук для заказа.

Второй компонент «Добавить книгу» подразумевает, что сервер проверит введенные данные на корректность: существует ли акая книга и не превышает ли количество, введенное пользователем, количества книг, доступных для заказа, в каталоге. Если все корректно, то книга будет добавлена в корзину.

Декомпозиция компонента «Добавить книги в корзину» представлена на рисунке 2.2.3.

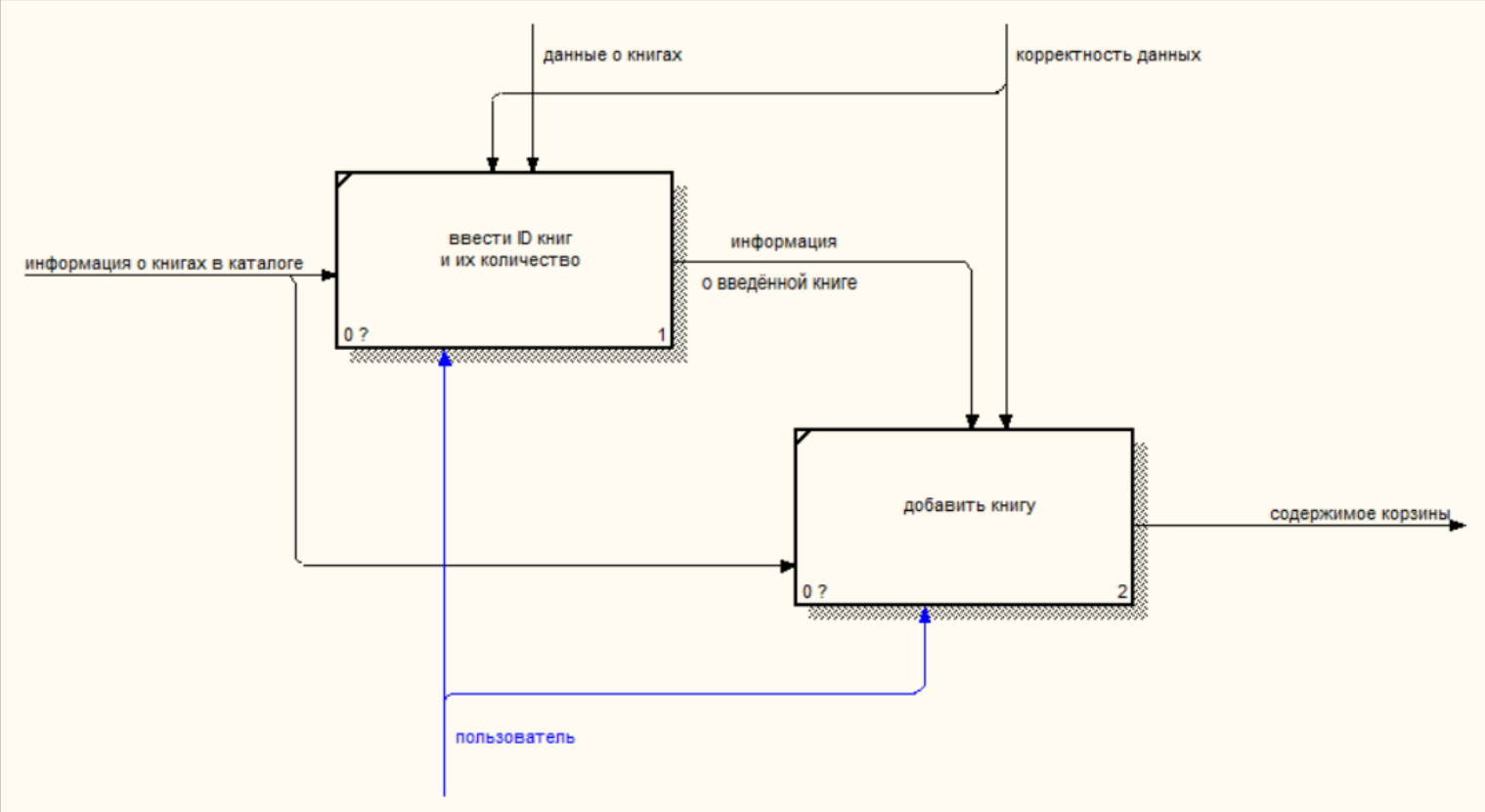


Рисунок 2.2.3 – Декомпозиция компонента «Добавить книги в корзину»

Процесс оформления заказа проходит в три этапа, поэтому декомпозиция блока «Оформить заказ» содержит три компонента.

Первый компонент «Просмотреть содержимое корзины» подрузамевает, что пользователь просматривает содержимое своей корзины на предмет соответствия нужных ему для заказа книг и корректности их количества, а также просмотра итоговых цен книг.

Второй компонент «Добавить заказ в таблицу заказов» подрузамевает, что пользователь убедился в корректности содержимого своей корзины и нажал кнопку «оформить заказ», тем самым поместив заказ в таблицу заказов со статусом «в обработке».

На последнем, третьем, этапе администратор изменяет статус заказа для подтверждения или отклонения. Пользователь может отследить статус своего заказа в таблице заказов.

Декомпозиция компонента «Оформить заказ» представлена на рисунке 2.2.4.

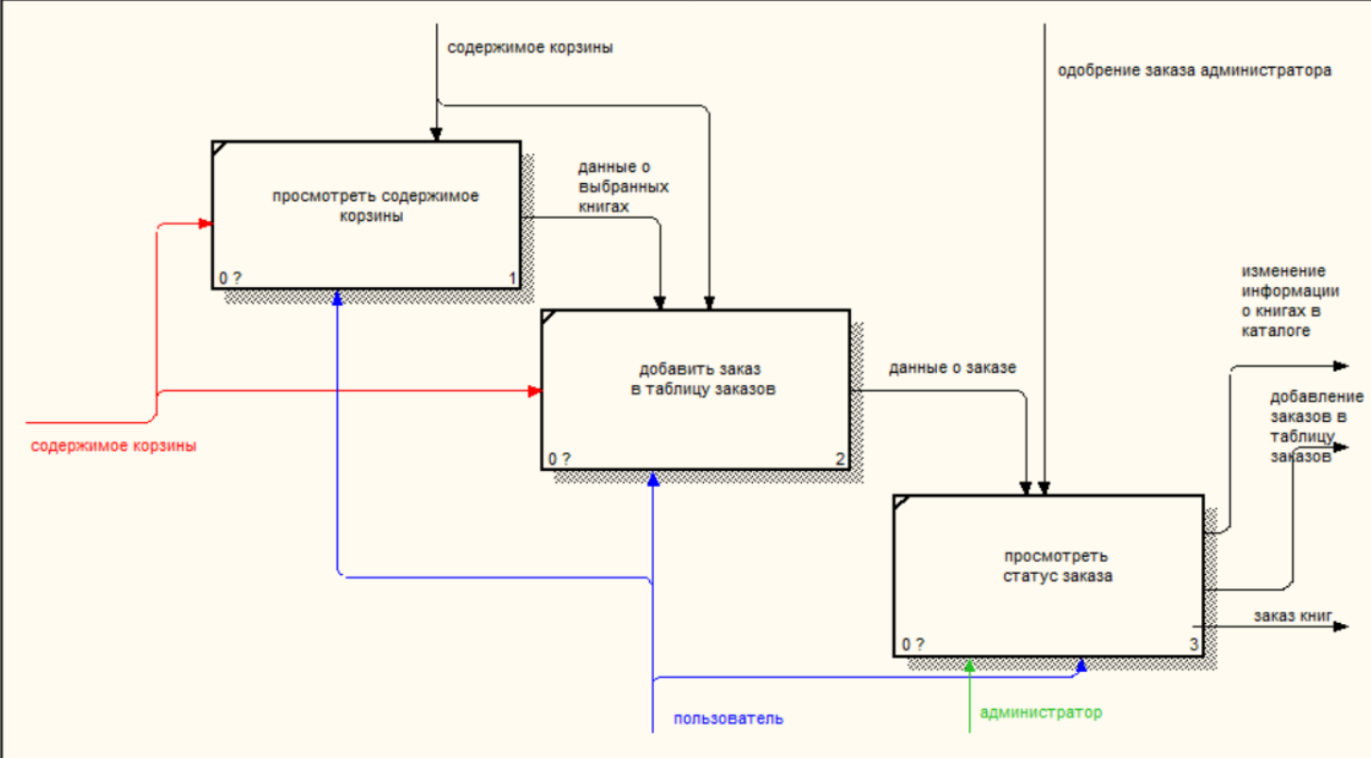


Рисунок 2.2.4 – Декомпозиция компонента «Оформить заказ»

В конце процесса заказа новых книг результатами будут заказ книг, добавление этого заказа в таблицу заказов и изменение информации о книгах в каталоге (уменьшение количества книг).

## 2.3 Описание функционала разрабатываемого приложения

Данные, полученные в результате проведенного анализа существующих аналогов, были взяты за основу при разработке приложения. Программа будет обладать следующим функционалом:

– возможность хранить данные о книгах, данные о пользователях, их личных корзинах покупок и данных о заказах;

– просмотр каталога книг, содержимого корзины, таблицы заказов;

– возможность провести расчёт определённым методом оценки;

– добавление запроса на редактирование книг, корзины покупок и статуса заказа;

– возможность оформления заказов;

– возможность регистрации пользователей;

Для разработки программного приложения системы учета новых поступлений в библиотеку заявленного функционала будет разработана база данных, в которой будет организовано хранение данных пользователей, книг, соедржимого корзин покупок и данных о заказах, что позволит обеспечить быстрый доступ к данным и их систематизацию.

# 3 ИНФОРМАЦИОННАЯ МОДЕЛЬ СИСТЕМЫ И ЕЁ ОПИСАНИЕ

Хранение данных программы организовано в виде базы данных, включающей 5 таблиц: «Пользователи», «Администратор», «Книга», «Корзина», «Заказы». Схема базы данных представлена на рисунке 3.1.

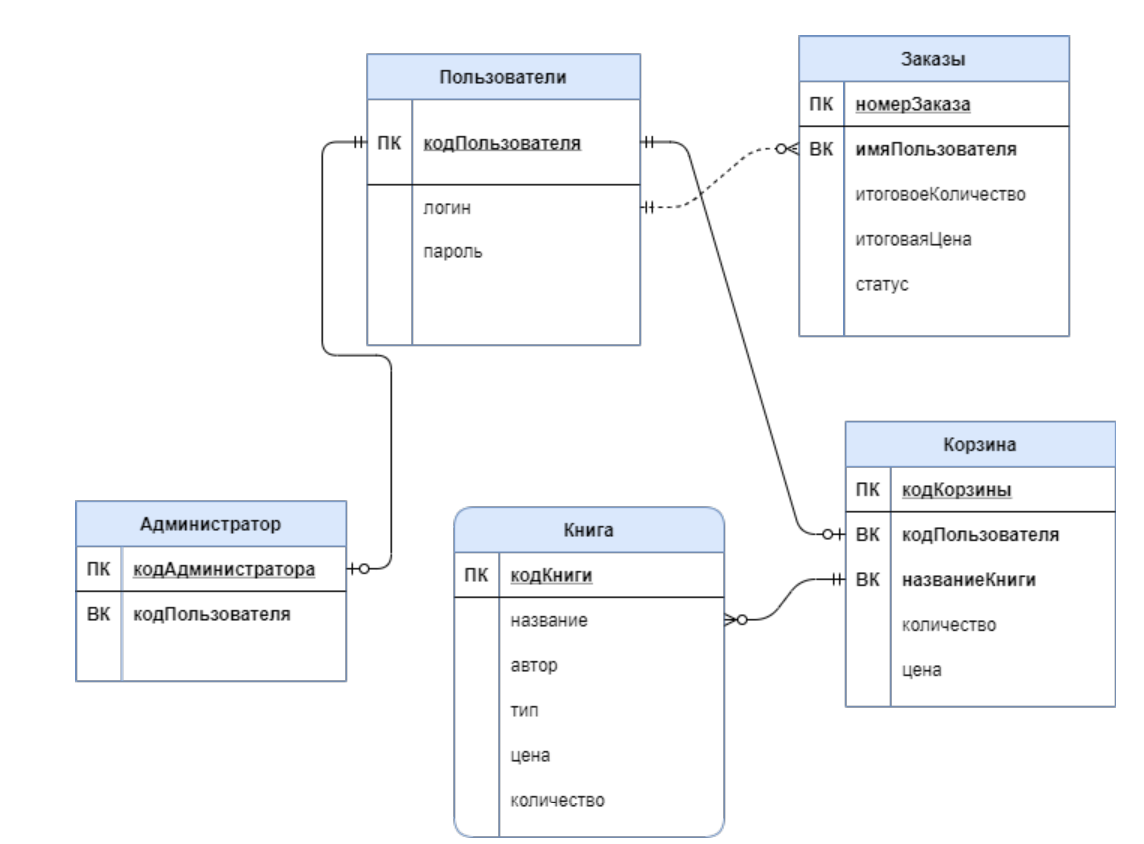


Рисунок 3.1 – База данных системы учета новых поступлений в библиотеку

Таблица «Пользователи» содержит следующие поля:

–кодЗаписи (первчиный ключ), в котором хранится идентификационный номер учётной записи пользователя;

–логин, в котором хранится логин пользователя;

– пароль, в котором хранится пароль для входа в систему.

Таблица «Администратор» содержит следующие поля:

–кодЗаписи (первчиный ключ), в котором хранится идентификационный номер учётной записи администратора;

–кодПользователя (внешний ключ), в котором хранится идентификационный номер учётной записи пользователя.

Таблица «Книга» содержит следующие поля:

– кодКниги (первичный ключ), в котором хранится идентификационный номер книги;

– название, в котором хранится название книги;

– автор, в котором хранится имя и фамилия автора книги;

– тип, в котором хранится тип книги;

– цена, в котором хранится цена книги;

– количество, в котором хранится количество штук данной книги.

Таблица «Корзина» содержит внешние ключи «кодПользователя» и «названиеКниги» и следующие поля:

– количество, в котором хранится количество штук данной книги для заказа;

– цена, в котором хранится высчитанная цена данной книги для заказа (в зависимости от количества).

Таблица «Заказы» содержит следующие поля:

– номерЗаказа (первичный ключ), в котором хранится идентификацонный номер заказа;

– имяПользователя (внешний ключ), в котором хранится логин пользователя, который оформил заказ;

– итоговаяЦена, в котором хранится итоговая цена всех книг в заказе;

– итоговоеКоличество, в котором хранится итоговое количество всех книг, которые заказали;

– статус, в котором хранится статус заказа (по умолчанию – «в обработке»).

# 4 ОПИСАНИЕ АЛГОРИТМОВ, РЕАЛИЗУЮЩИХ БИЗНЕС-ЛОГИКУ

Для построения блок-схем использовался веб-сервис DRAW IO, предназначенный для формирования диаграмм и схем.

При запуске программа предлагает пользователю войти или зарегистрироваться. При регистрации пользователю необходимо подобрать логин, который ещё не использован, и придумать надёжный пароль. При авторизации пользователю предлагается ввести логин и пароль, после чего происходит их проверка на соответствие в базе данных. Если вход выполнен, открывается меню пользователя либо администратора, в зависимости от входных данных, в противном случае предлагается повторить попытку входа. Прекращение выполнения программы происходит при закрытии её нажатием на крестик.

Схема алгоритма работы программы приведена на рисунке 4.1.

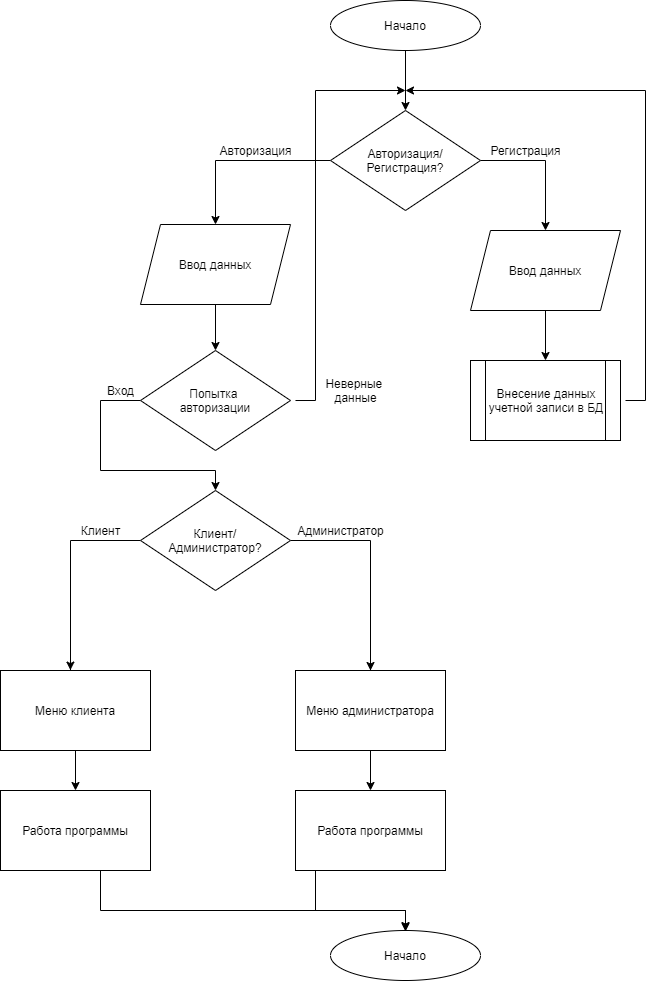


Рисунок 4.1 – Блок-схема работы программы

При добавлении книг в корзину пользователь вводит id книги, ее желаемое количество и нажимает на кнопку «добавить в корзину». При наличие такой книги и такого количества книг будут добавлены в корзину и будет выведено соответсвующее сообщения. В противном случае же выведется сообщение об ошибке. Блок-схема данного процесса представлена на рисунке 4.2.

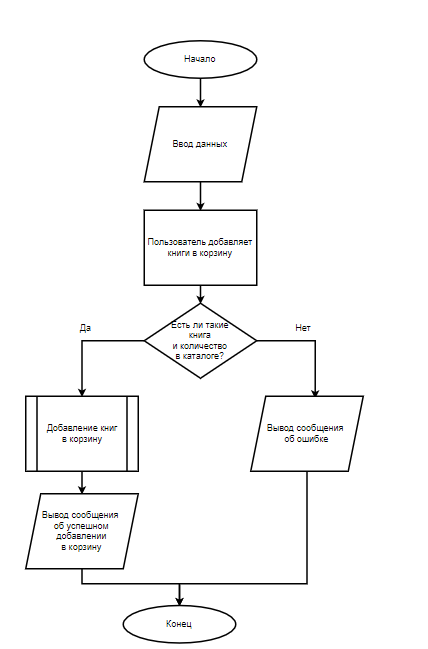


Рисунок 4.2 – Блок-схема добавления книг в корзину

При оформлении заказа пользователь нажимает на кнопку «оформить заказ».Если корзина не пуста, то будет выведено окно о успешном оформлении заказа. В противном случае выведется сообщение об ошибке. Блок-схема данного процесса представлена на рисунке 4.3.

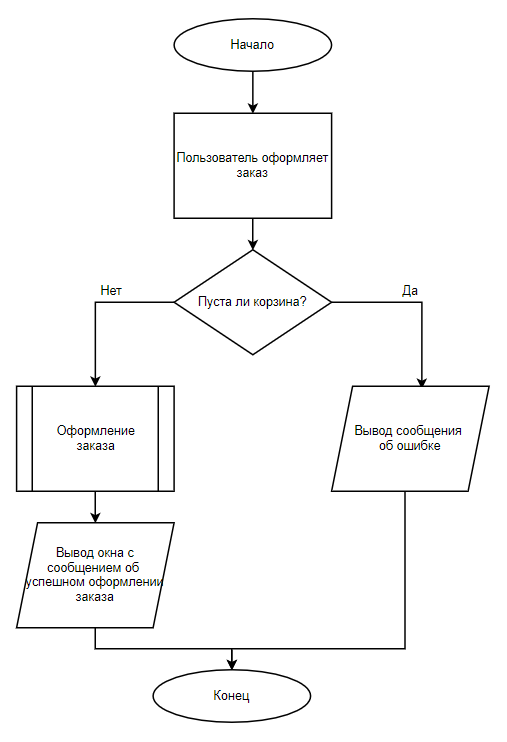


Рисунок 4.3 – Блок-схема оформления заказа

# 5 МОДЕЛИ ПРЕДСТАВЛЕНИЯ СИСТЕМЫ И ИХ ОПИСАНИЕ

Для описания представления системы используется язык UML.

UML – унифицированный язык моделирования – это система обозначений, которую можно применять для объектно-ориентированного анализа и проектирования. Его можно использовать для визуализации, спецификации, конструирования и документирования программных систем.

**5.1 Диаграмма вариантов использования**

Диаграмма вариантов использования обеспечивает высокоуровневое

описание того, что система в состоянии сделать и с кем (или чем) она будет

взаимодействовать.

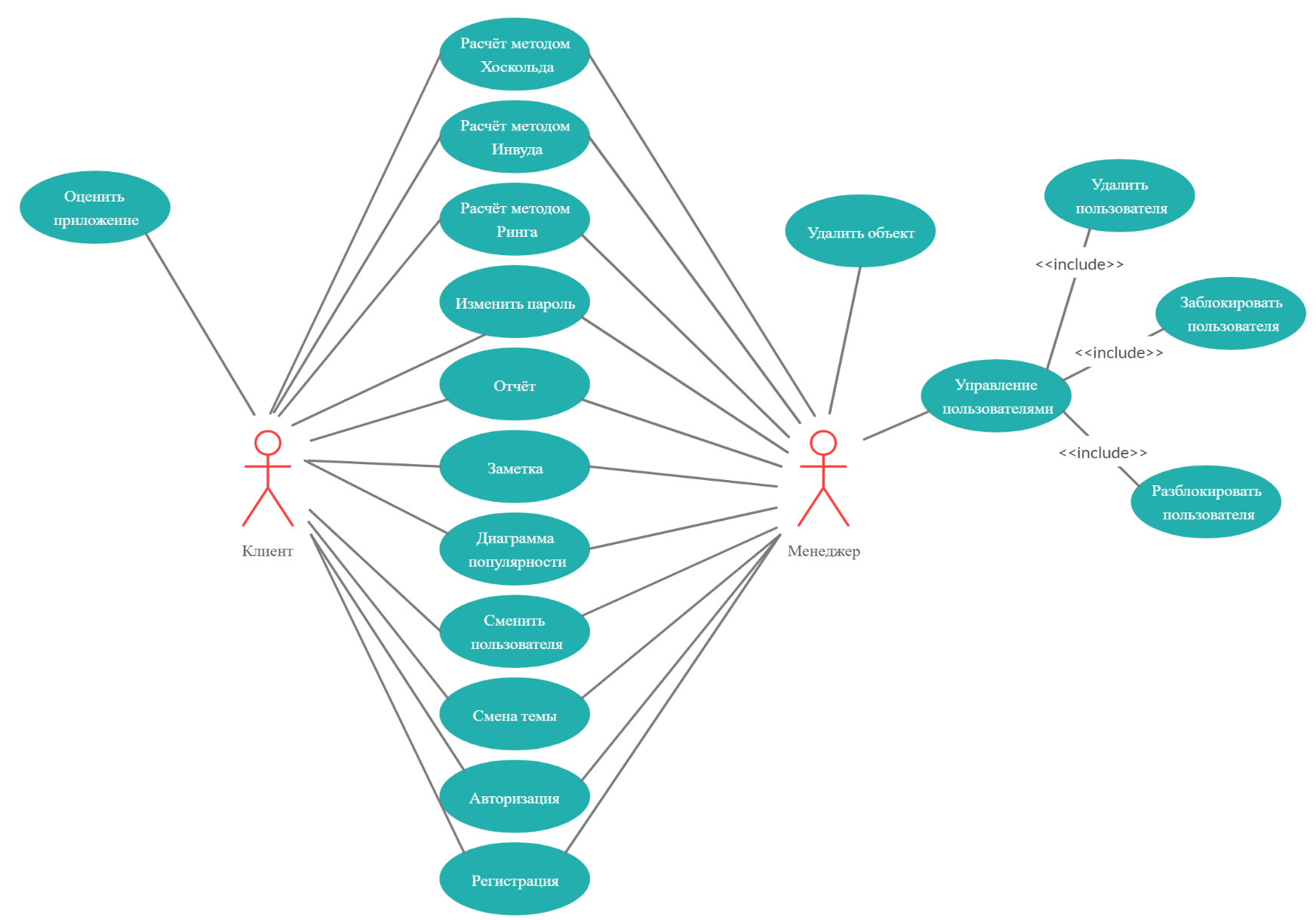
На рисунке 5.1.1 представлена диаграмма вариантов использования для клиента и администратора.

Рисунок 5.1.1 – Диаграмма вариантов использования для клиента и администратора

Субъектами системы являются клиент и администратор.

Каждый пользователь должен пройти процесс авторизации либо регистрации.

Пользователь и администратор могут получить оценку объекта недвижимости одним из трёх методов, изменить пароль учётной записи, составить отчёт о расчёте и записать его в файл, просмотреть диаграмму популярности методов, выйти из системы, сменить тему. Расчёт методом Ринга: заполнение необходимой информации об объекте, расчёт стоимости методом Ринга и занесение полученных данных в таблицу. Расчёт методом Инвуда: заполнение необходимой информации об объекте, расчёт стоимости методом Инвуда и занесение полученных данных в таблицу. Расчёт методом Хоскольда: заполнение необходимой информации об объекте, расчёт стоимости методом Хоскольда и занесение полученных данных в таблицу. Изменить пароль: ввести новый пароль, записать в таблицу. Отчёт: составление отчёта по данным последнего расчёта стоимости и сохранение данных объекта и его стоимости в текстовый файл. Диаграмма популярности методов: анализ таблицы объектов, определение соотношения количества расчётов, выполненных определённым методом, представление графической информации на основе полученных данных в виде круговой диаграммы. Сменить пользователя: выход из системы, запуск процесса авторизации. Сменить тему: сменить графическое оформление приложения.

Также клиенту доступна оценка приложения. Оценка приложения: выставление клиентом приложению балла от 1 до 5 и запись его в таблицу.

Менеджер осуществляет работу с таблицей объектов и клиентов посредством возможностей удаления объекта, управления пользователями. Удаление объекта: выбор объекта из таблицы, удаление его из таблицы. Управление пользователями: удалить пользователя, заблокировать пользователя, разблокировать пользователя. Удалить пользователя: выбор учётной записи из таблицы, её удаление. Заблокировать пользователя: перевод статуса поля «доступ» учётной записи в состояние «заблокирован», что означает запрет на доступ к системе, обновление статуса в таблице. Разблокировать пользователя: перевод статуса поля «доступ» учётной записи в состояние «разрешён», что означает разрешение доступа пользователя к системе, обновление статуса в таблице.

Смоделировав диаграмму вариантов использования, мы получаем полное графическое представление вариантов использования нашей системы.

Определены субъекты системы, клиент и менеджер.

**5.2 Диаграмма состояний**

Диаграмма состояний описывает все возможные состояния одного экземпляра определенного класса и возможные последовательности его переходов из одного состояния в другое, то есть моделирует все изменения состояний объекта как его реакцию на внешние воздействия.

Диаграмма состояний представлена в Приложении Г.

После запуска программы ожидается ввод данных для авторизации. Если была нажата кнопка «выход», завершается работа программы. Если были введены данные, они отправляются на сервер. После отправки данных на сервер ожидается получение данных от сервера. Сервер посылает SQL-запрос, после чего ожидает данных от БД. После получения данных от БД сервер обрабатывает их и отправляет результат клиенту. Клиент проверяет полученный результат, если он положительный, начинается работа пользователя с системой, если же результат отрицательный, клиент переходит в состояние ожидания ввода.

**5.3 Диаграмма последовательностей**

Диаграмма последовательности описывает поведение только одного варианта использования. На такой диаграмме отображаются только экземпляры объектов и сообщения, которыми они обмениваются между собой.

Диаграмма состояний представлена в Приложении Е.

На данной диаграмме отображён процесс авторизации пользователя.

После запуска приложения пользователь вводит данные, с помощью которых формируется запрос на авторизацию, который отправляется серверу. Сервер отправляет запрос на проверку данных в базу данных, а после получения данных пользователя от базы данных отправляет их клиенту и начинает работу в режиме, определённым ролью пользователя. После окна авторизации открывается либо окно администратора, либо окно клиента, в зависимости от роли.

**5.4 Диаграмма развёртывания и компонентов**

Диаграмма развертывания предназначена для иллюстрации элементов и компонентов программы, которые существуют во время выполнения. В этом случае отображаются только экземпляры компонентов программы, которые являются исполняемыми файлами или динамическими библиотеками.

Диаграмма развертывания содержит графическое представление процессоров, устройств, процессов и взаимосвязей между ними. Также она описывает особенности физического представления системы.

Диаграмма развёртывания представлена в Приложении Д.

На данной диаграмме узлами являются ПК клиента, ПК администратора, Сервер и База Данных.

Для запуска приложения необходимо наличие установленной среды JDK версии JDK8 на компьютерах менеджера, клиентов и сервера.

Обмен данными между ПК менеджера, ПК клиента и сервером осуществляется по протоколу TCP/IP, а связь сервера с базой данных осуществляется по протоколу Hibernate .

**5.5 Диаграмма классов**

Диаграмма классов описывает типы объектов системы и различные статические связи между ними.

Диаграмма классов представлена в Приложении Ж.

**5.6 Паттерны проектирования**

При разработке программы были использованы паттерны MVC и DAO.

MVC разделяет работу приложения на три отдельные функциональные роли: модель данных (model), пользовательский интерфейс (view) и управляющую логику (controller).

Основная идея этого паттерна в том, что и контроллер, и представление зависят от модели, но модель никак не зависит от этих двух компонент.

Диаграмма взаимодействия компонентов паттерна «MVC» изображена на рисунке 5.6.1

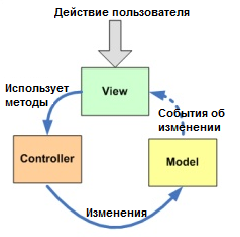
****

Рисунок 5.6.1 – Диаграмма для паттерна «MVC»

Функциональность паттерна DAO заключается в том, чтобы скрыть от приложения все сложности, связанные с выполнением операций CRUD в базовом механизме хранения. Это позволяет обоим слоям развиваться отдельно, ничего не зная друг о друге.

Используя DAO, можно удобно разделять уровни доступа при работе с базой данных, чётко видеть эти уровни доступа и легко оперировать ими, не привязываясь к конкретной реализации хранения данных.

# 6 РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ

Работа с программным приложением проходит в трёх режимах: работа в режиме администратора, работа клиента и работа серверной части.

## 6.1 Работа сервера

Данное программное приложение было разработано с использованием протокола TCP/IP. Для работы пользователя с программой должен быть запущен предварительно сервер.

На серверном окне выводится вся информация о работе программы: количество клиентов, взаимодействующих с данным сервером, отслеживание состояния потоков ввода/вывода, отчет о действиях клиента.

Серверное окно представлено на рисунке 6.1.1.

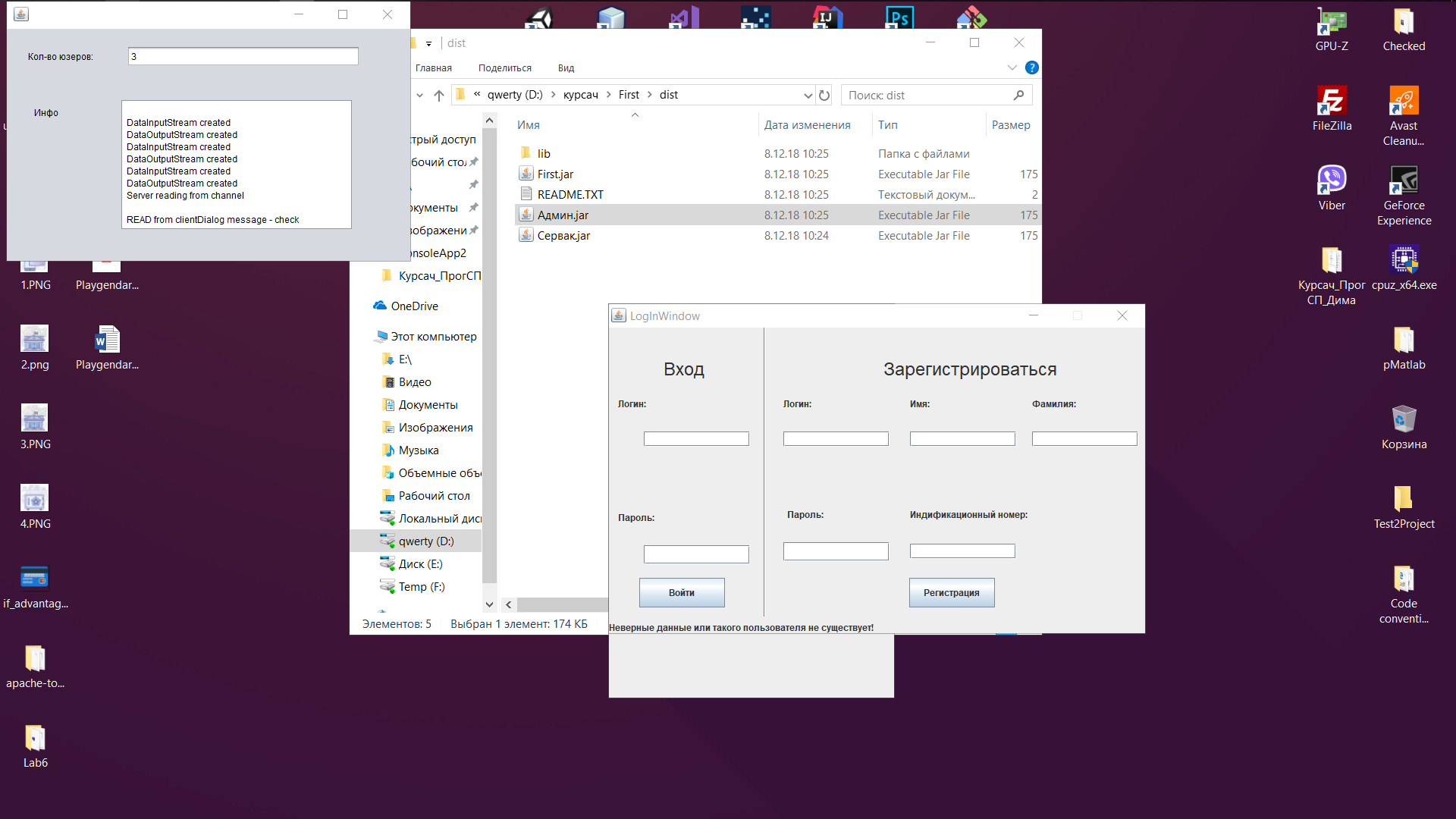


Рисунок 6.1.1 – Работа серверной части приложения

## 6.2 Работа приложения в режиме администратора

При запуске приложения от лица администатора пользователю предлагается авторизироваться. При попытке регистрации в качестве администратора система выведет сообщение об ошибке (рисунок 6.2.2).

При регистрации и авторизации пользователю предлагается заполнить три поля: логин, пароль и отметить галочкой поле «администратор».

Окно запуска представлено на рисунке 6.2.1

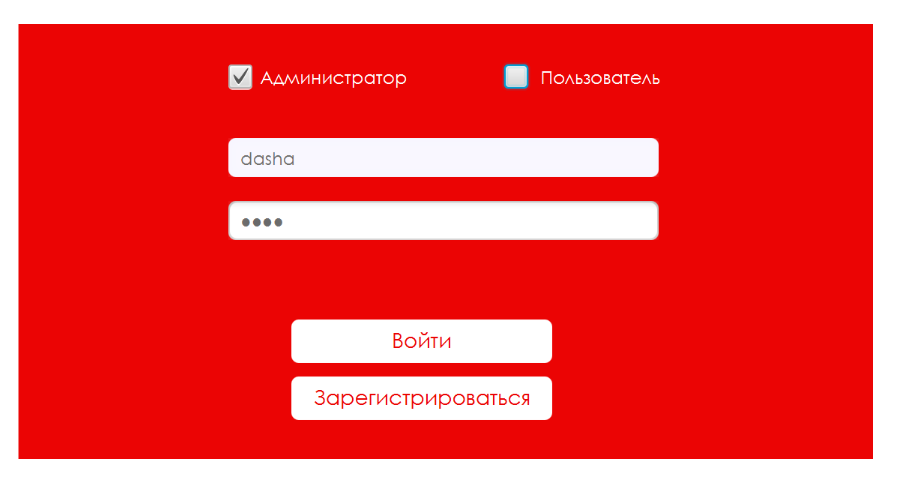


Рисунок 6.2.1 – Окно запуска в режиме администратора

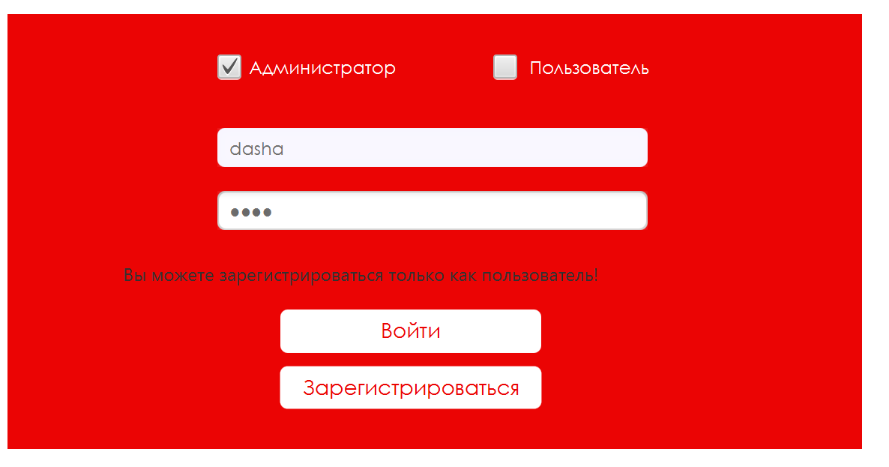


Рисунок 6.2.2 – Ошибка при попытке регистрации в качестве администратора

При авторизации при некорректном заполнении полей пользователю приходит сообщение о присутствии ошибок ввода.

При вводе логина и пароля в режиме администратора пользователю открывается главное меню программы. Администратору доступны запросы на выполнение действий с пользователями, заказами, книгами и просмотр статистики. Также присутствует кнопка возвращения на предыдущее окно «назад». (рис.6.2.3)

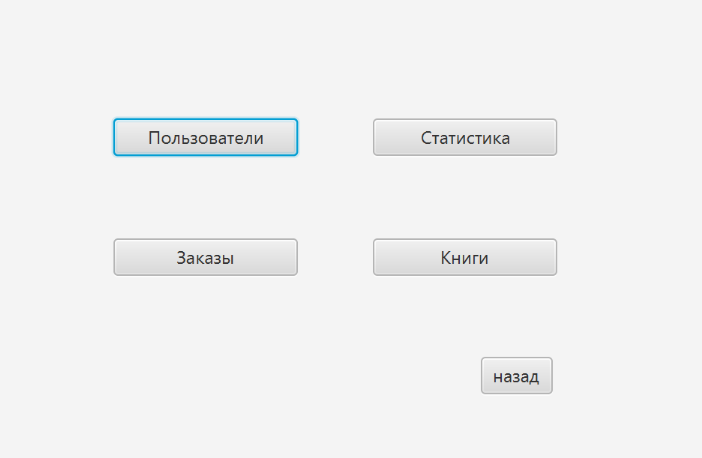


Рисунок 6.2.3 – Главное меню в режиме администратора

При выборе раздела меню «Книги» администратору открывается меню, представленное на рисунке 6.2.4. В таблице указаны порядковый номер книги, ее автор, тип, количество и цена. Есть опции редактирования книги, удаления по ее id . Также можно добавить новую книгу. При успешном добавлении появляется соответствующее сообщение (рисунок 6.2.5).

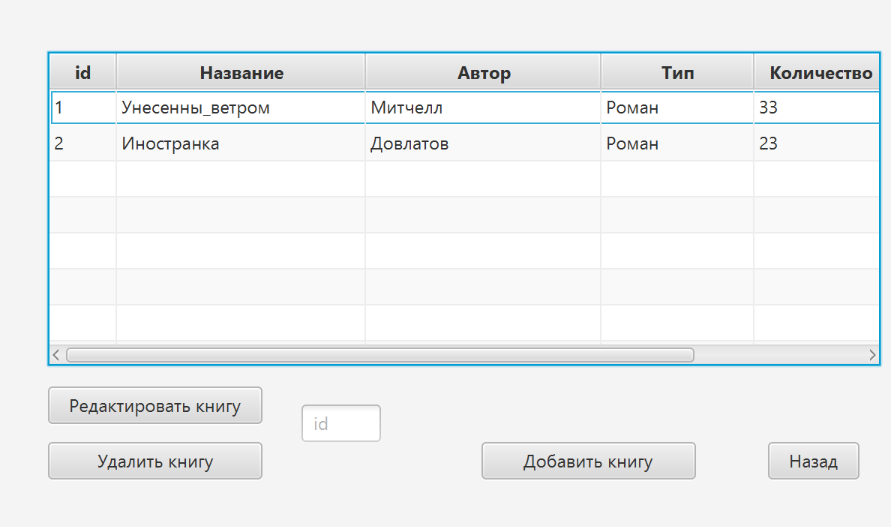


Рисунок 6.2.4 – Раздел меню «Книги»

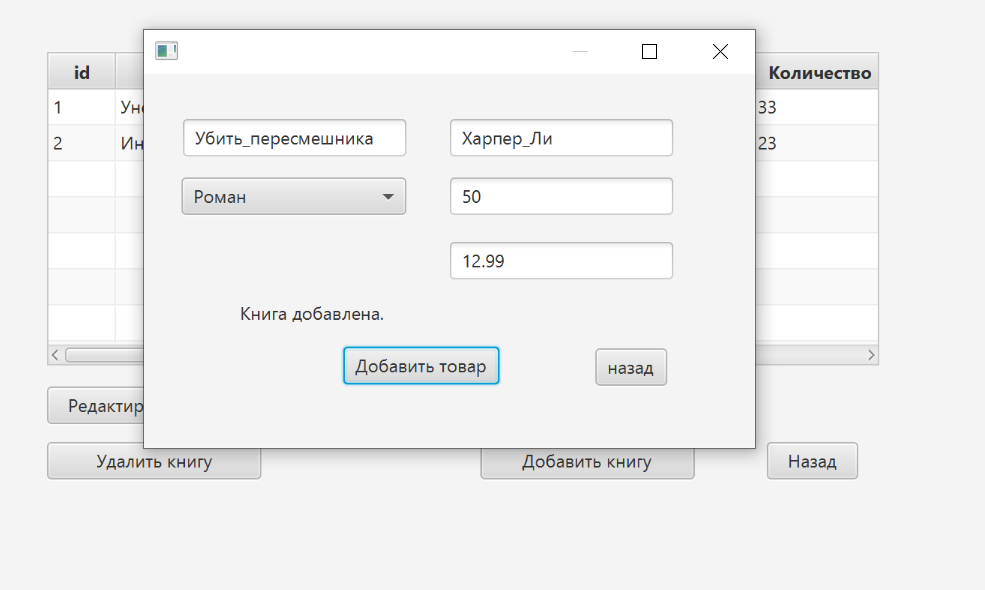


Рисунок 6.2.4 – Добавление книги

При выборе раздела меню «Заказы» администратору открывается окно с таблицей заказов, отправленных на подтверждение пользователями. В таблице указаны идентификационный номер заказа,итоговая стоимость и количество книг, а также статус заказа. Статус заказа по умолчанию «в обработке». Администратор может подвердить или отклонить заказ (рисунок 6.2.5).

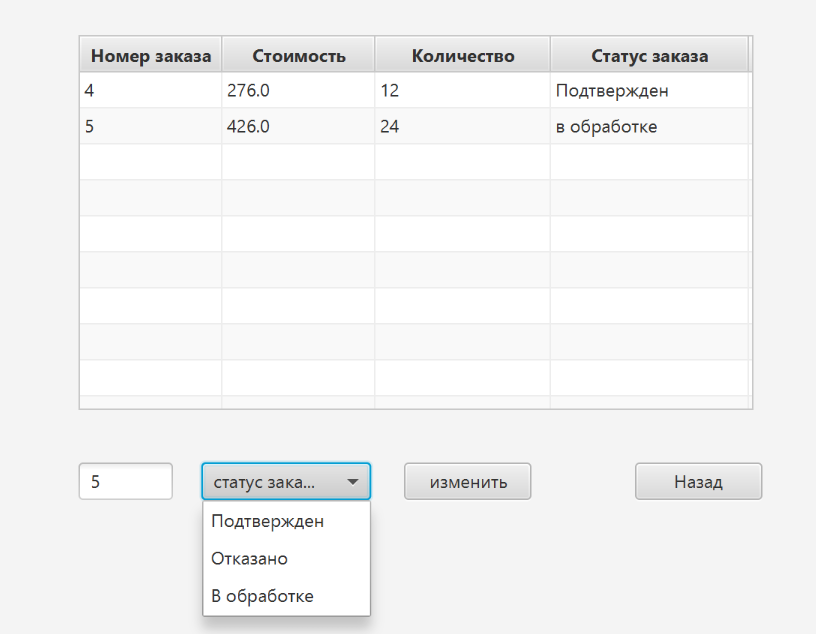


Рисунок 6.2.5 – Раздел меню администратора «Заказы»

При выборе раздела меню «Пользователи» администратору открывается окно с таблицей пользователей. В таблице указаны идентификационный номер пользователя, его логин и пароль. Администратор может удалить пользователя и добавить администратора по id (рисунок 6.2.5).

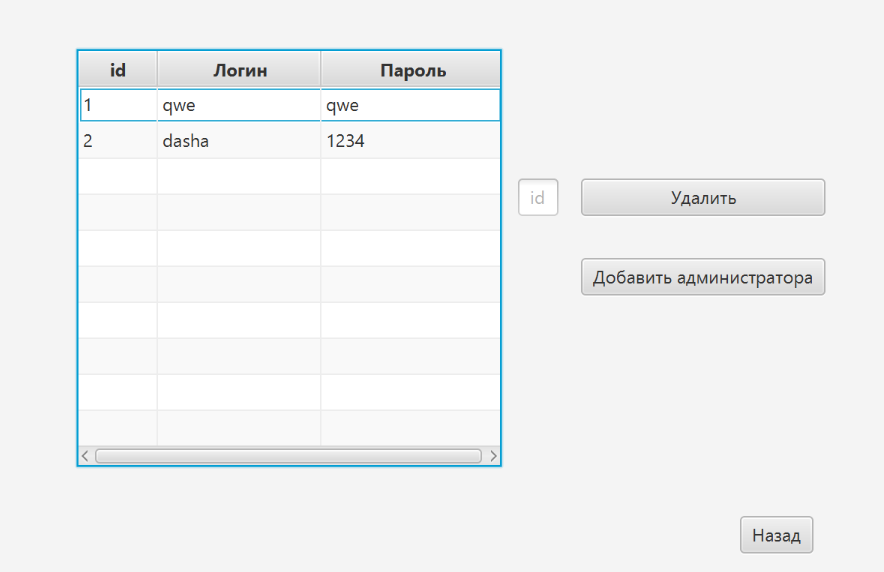


Рисунок 6.2.5 – Раздел меню администратора «Пользователи»

## 6.3 Работа приложения в режиме клиента

При запуске приложения в режиме клиента пользователю сразу же предлагается зарегистрироваться или авторизироваться в системе (рисунок 6.3.1.) .

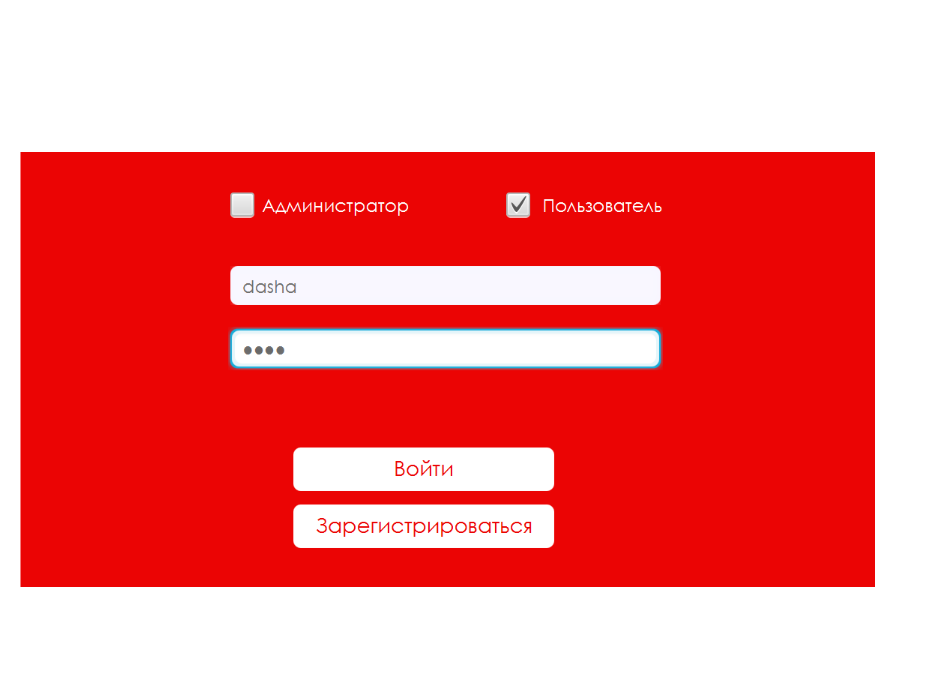


Рисунок 6.3.1 – Окно запуска приложения в режиме клиента

При регистрации пользователю необходимо заполнить следующие поля: логин, имя, фамилия, пароль и указать пол.

Регистрация пользователя представлена на рисунке 6.3.2.

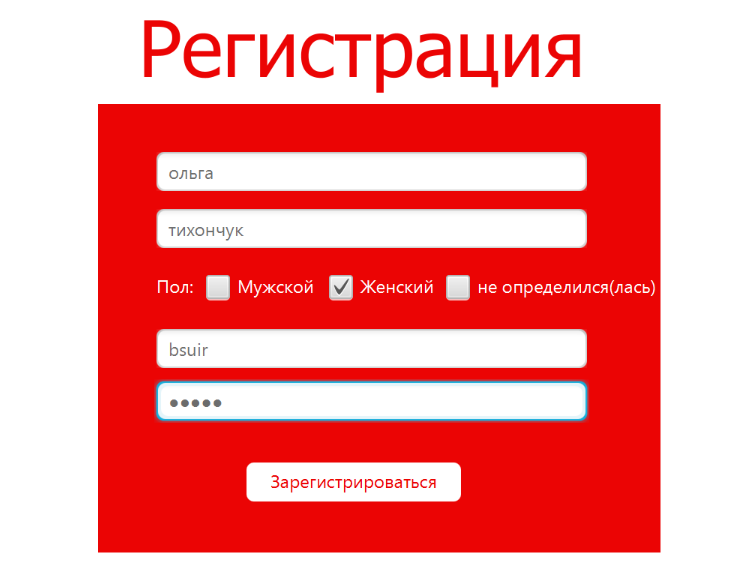


Рисунок 6.3.2 –Регистрация клиента

При входе в программу в режиме клиента открывается главное меню. Клиенту доступны следующие опции: каталог книг, корзина, таблица заказов.

Окно главного меню представлено на рисунке 6.3.3.

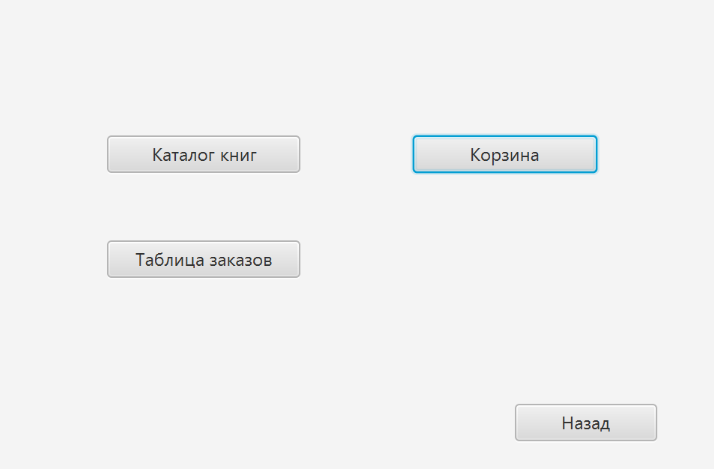


Рисунок 6.3.3 – Главное меню клиента

При выборе раздела меню «Каталог книг» пользователю открывается таблица, в которой указаны уникальный идентификационный номер книги, название, автор, тип, цена и количество. Пользователь может добавить книгу в корзину, указав ее id и введя желаемое количество. При успешном добавлении будет выведено соответсвующее сообщение (рисунок 6.3.4.).

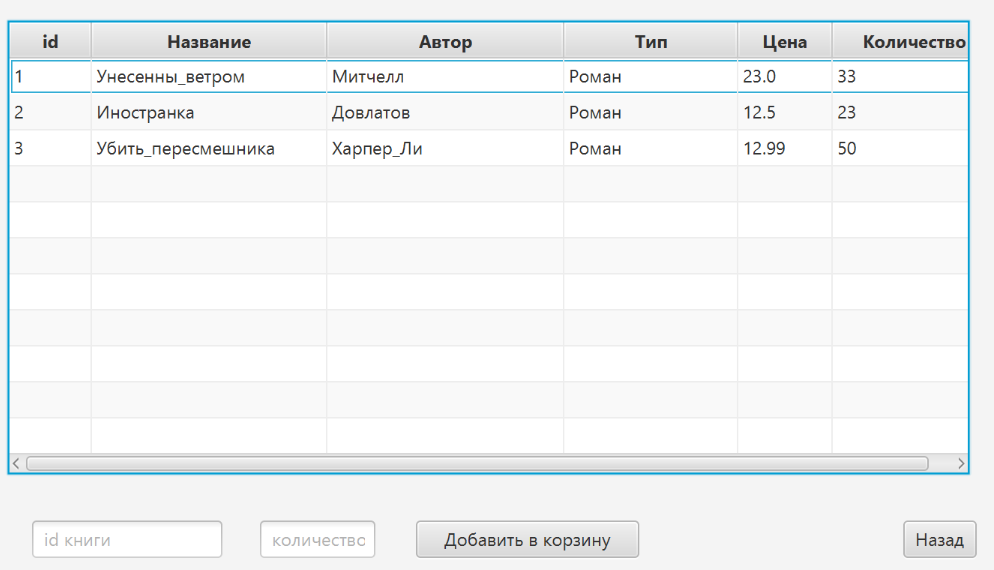


Рисунок 6.3.4 – Раздел меню «Каталог книг»

При выборе раздела меню «Корзина» пользователю открывается окно с информацией о содержимом его корзины для покупок книг. В таблице указаны идентификационный номер записи корзины, название книги, желаемое количество книг и их цена. Пользователь может удалить книгу по id из корзины и оформить заказ (рисунок 6.3.7). При успешном оформлении заказа будет выведено соотетсвующее сообщение (рисунок 6.3.8)..

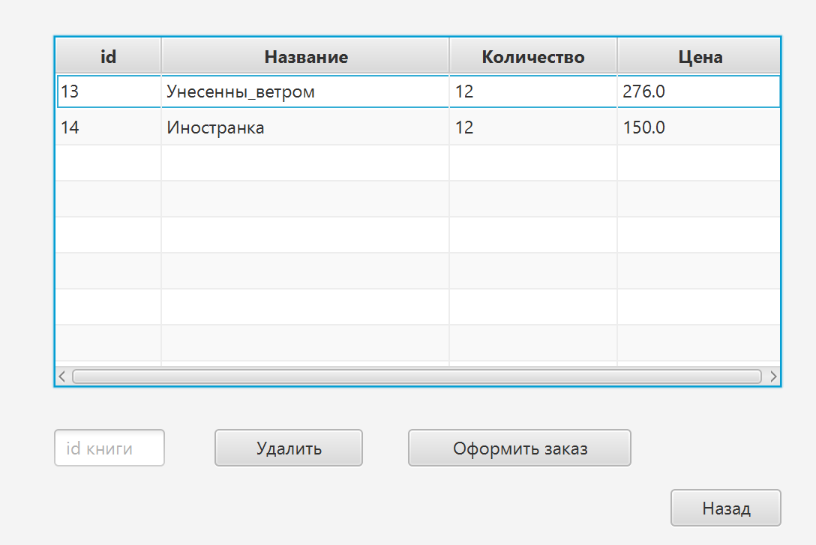


Рисунок 6.3.7 – Раздел меню «Информация об аккаунте»

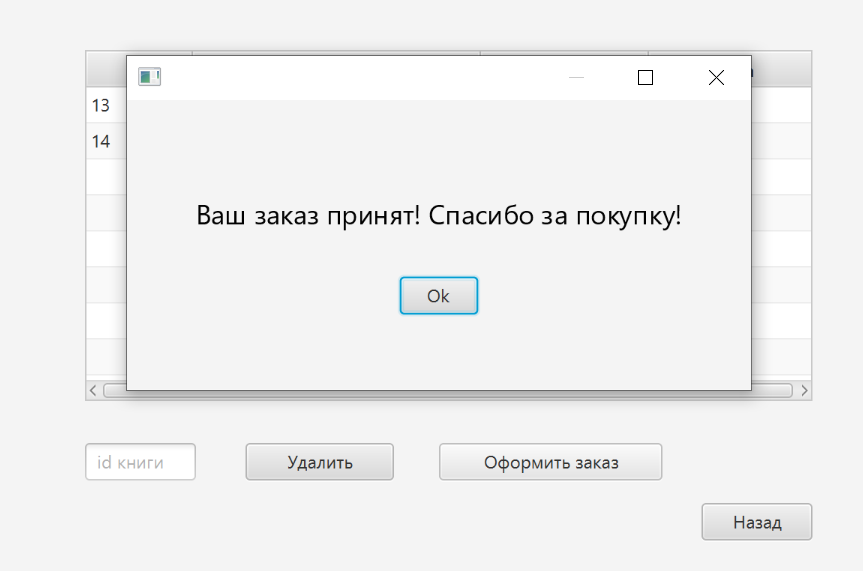


Рисунок 6.3.7 – Успешное оформление заказа

При выборе раздела меню «Заказы» пользователю открывается таблица с номером заказа, итговой стоимость заказа, количестве книг и статусом. Пользователь может просматривать свои заказы на предмет изменения статуса его оформленного заказа.

Данный раздел меню представлен на рисунке 6.3.8.

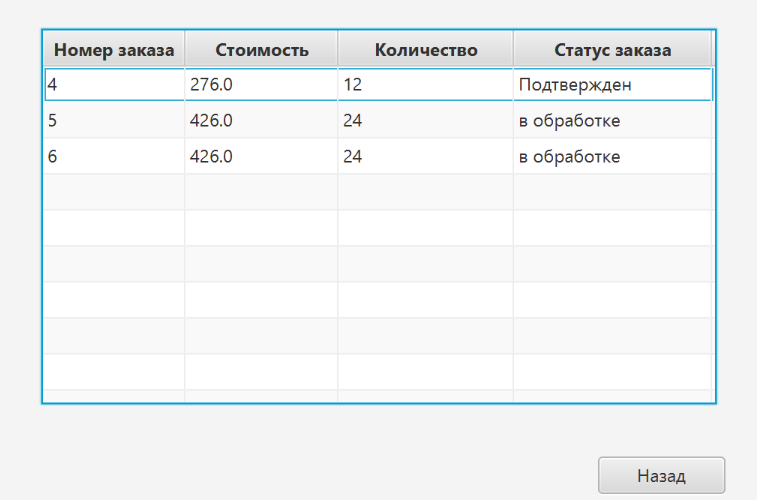


Рисунок 6.3.8 – Раздел меню «Таблица заказов»

# ЗАКЛЮЧЕНИЕ

В результате разработки данного проекта был создан программный модуль, предоставляющий заказ новых книг.

Интерфейс данного программного продукта интуитивно понятен каждому пользователю, а также защищен от ввода некорректных значений.

Применение подобных средств реляционной базы данных, как таблицы, позволяет упростить процессы редактирования и записи данных. Используемое в данном проекте шифрование обеспечивает хранение данных в зашифрованном виде, позволяет сделать систему безопасности более надежной и защищенной от всевозможных внешних атак.

Конечный продукт обладает высокой актуальностью на рынке, так как отличается от своих аналогов понятностью интерфейса, удобным и понятным функционалом, совмещение двух систем автоматизации оценки объектов недвижимости, покупке и продажи ее. Обладает надежным шифрованием, обеспечивающим конфиденциальность личной информации.

В ходе написания программы использовались основные средства языка программирования Java, позволяющие наиболее эффективно реализовать курсовой проект на базе соединения Клиент-Сервер и подойти к его разработке с точки зрения объектно-ориентированного программирования.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

[1] Википедия – свободная энциклопедия [Электронный ресурс]. – Электронные данные. – Режим доступа: https://ru.wikipedia.org/wiki/%D0%9E%D1%86%D0%B5%D0%BD%D0%BA%D0%B0\_%D0%BD%D0%B5%D0%B4%D0%B2%D0%B8%D0%B6%D0%B8%D0%BC%D0%BE%D1%81%D1%82%D0%B8

[2] (R) Оценка [Электронный ресурс]. – Электронные данные. – Режим доступа: https://rocenka.com/

[3] Про Оценка [Электронный ресурс]. – Электронные данные. – Режим доступа: http://proocenka.by/servises/kalkulyator-oczenki.html

[4] ArticleKZ [Электронный ресурс]. – Электронные данные. – Режим доступа: https://articlekz.com/article/9048

[5] ТЕТРАДЬ [Электронный ресурс]. – Электронные данные. – Режим доступа: http://7267507.ru/?page\_id=4042

[6] Хабр [Электронный ресурс]. – Электронные данные. – Режим доступа:

https://habr.com/ru/post/262243/

[7] Help iks [Электронный ресурс]. – Электронные данные. – Режим доступа: https://helpiks.org/9-19880.html

[8] Фин вопрос [Электронный ресурс]. – Электронные данные. – Режим доступа: http://finvopros.com/investitsionnaya-stoimost-eto.html

[9] CFin [Электронный ресурс]. – Электронные данные. – Режим доступа:

https://www.cfin.ru/appraisal/realty/profitable\_realestate\_valuation.shtml

# ПРИЛОЖЕНИЕ А

# (Обязательное)

**Листинг основного кода сервера**

# ПРИЛОЖЕНИЕ Б

# (Обязательное)

**Листинг основного кода клиента**

# ПРИЛОЖЕНИЕ В

# (Обязательное)

**Скрипт базы данныx**

CREATE SEQUENCE users\_seq;

CREATE TABLE IF NOT EXISTS users (

id\_user int NOT NULL DEFAULT NEXTVAL ('users\_seq'),

login varchar(30) NOT NULL,

password varchar(10) NOT NULL,

PRIMARY KEY (id\_user),

CONSTRAINT login UNIQUE (login)

) ;

CREATE SEQUENCE admin\_seq;

CREATE TABLE IF NOT EXISTS admin (

id\_admin int NOT NULL DEFAULT NEXTVAL ('admin\_seq'),

user\_id int NOT NULL,

PRIMARY KEY (id\_admin),

FOREIGN KEY (user\_id) references users (id\_user)

) ;

CREATE SEQUENCE books\_seq;

CREATE TABLE IF NOT EXISTS books (

id\_book int NOT NULL DEFAULT NEXTVAL ('books\_seq'),

name varchar(50) NOT NULL,

author varchar(50) NOT NULL,

type varchar(50) NOT NULL,

price double precision NOT NULL,

amount int NOT NULL,

PRIMARY KEY (id\_book),

CONSTRAINT name UNIQUE (name)

) ;

CREATE SEQUENCE baskets\_seq;

CREATE TABLE IF NOT EXISTS baskets (

id\_basket int NOT NULL DEFAULT NEXTVAL ('baskets\_seq'),

user\_id int NOT NULL,

book\_name varchar(50) unique NOT NULL,

price double precision NOT NULL,

amount int NOT NULL,

PRIMARY KEY (id\_basket),

FOREIGN KEY (user\_id) references users (id\_user),

FOREIGN KEY (book\_name) references books (name)

);

CREATE SEQUENCE orders\_seq;

CREATE TABLE IF NOT EXISTS orders (

order\_number int AUTO- GENERATED NOT NULL,

user\_name varchar(50) NOT NULL,

total\_price double precision NOT NULL,

total\_amount int NOT NULL,

status varchar(30) default ('в обработке') NOT NULL,

PRIMARY KEY (id\_order),

FOREIGN KEY (user\_name) references users (login);

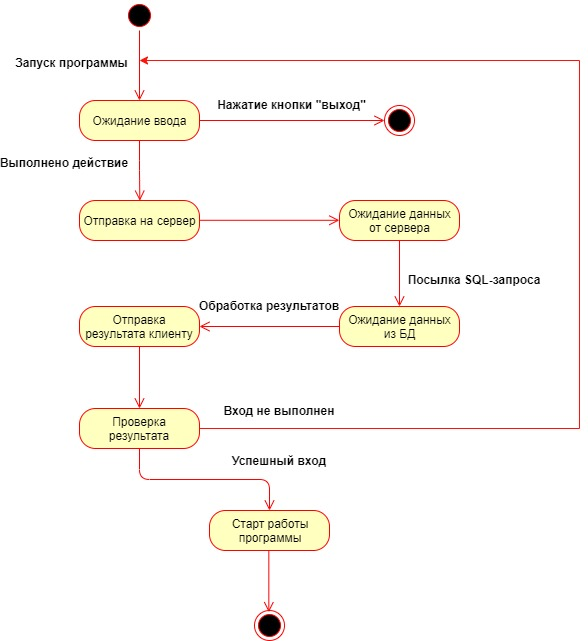
);

# 

# ПРИЛОЖЕНИЕ Г

# (Обязательное)

**Диаграмма состояний**



# ПРИЛОЖЕНИЕ Д

# (Обязательное)

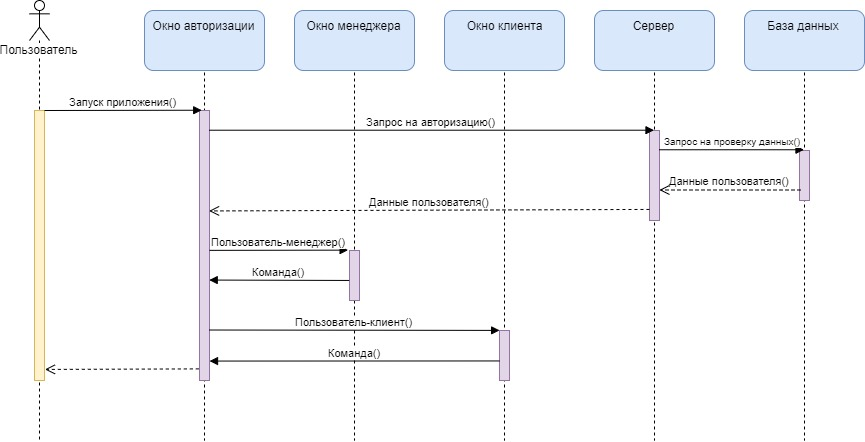
**Диаграмма развёртывания**

# 

# ПРИЛОЖЕНИЕ Е

# (Обязательное)

**Диаграмма последовательностей**



# ПРИЛОЖЕНИЕ Ж

# (Обязательное)

**Диаграмма классов**