Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение   
высшего образования

**Иркутский национальный исследовательский технический университет**

|  |
| --- |
| Институт информационных технологий и анализа данных |
| наименование института |

Допускаю к защите

|  |  |
| --- | --- |
| Руководитель |  |
|  | подпись |
|  | З.А. Бахвалова |
|  | И.О. Фамилия |

|  |
| --- |
|  |
| Разработка прикладного программного обеспечения |
| наименование темы |

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовой работе по дисциплине

|  |
| --- |
| Технологии разработки программных комплексов |

|  |  |  |
| --- | --- | --- |
| 1.001.00.00 – ПЗ | | |
| обозначение документа | | |
| Выполнил студент |  | ИСТб-20-3 | |  |  | |  | Д.Э. Афанасьев |
|  |  | шифр | |  | подпись | |  | И.О. Фамилия |
| Нормоконтроль |  |  | |  |  | |  | З.А. Бахвалова |
|  |  |  | |  | подпись | |  | И.О. Фамилия |
| Курсовая работа защищена с оценкой | | | | |  | | | |

Иркутск 2023 г.

Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

**ИРКУТСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

**ЗАДАНИЕ**

**на курсовую работу**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **По курсу** | Технологии разработки программных комплексов | | | | | | | | | | | | | |
| **Студенту** | Афанасьеву Д.Э. | | | | | | | | | | | | | |
|  | (фамилия, инициалы] | | | | | | | | | | | | | |
| **Тема работы** | Разработка прикладного программного обеспечения | | | | | | | | | | | | | |
| **Исходные данные** | | Система информирования посетителей в книжном | | | | | | | | | | | |
| магазине. Проанализировать предметную область и реализовать систему в | | | | | | | | | | | | | |
| соответствии с жизненный циклом ПО. | | | | | | |  | | | | | | |
| **Рекомендуемая литература** | | | | | | |  | | | | | | | |
| |  | | --- | | 1. СТО 0052020 СИСТЕМА МЕНЕДЖМЕНТА КАЧЕСТВА. Учебно | | методическая деятельность. Оформление курсовых проектов (работ) и | | выпускных квалификационных работ технических специальностей | | (http://www.istu.edu/structure/57/2506/); | | 1. Буч Г. Объектно ориентированное проектирование с примерами применения: Пер. С англ. — М.: Конкорд, 1992519 с., ил | | 1. Курсовая работа «Разработка прикладного программного обеспечения» | | Руководство и методические указания для студентов направления | | 09.03.02 «Информационные системы и технологии». Бахвалова З.А. – | | Иркутск, Изд во ИРНИТУ, 2018, 61 с. | | | | | | | | | | | | | | | |
| Дата выдачи задания | | |  | | | 16 | |  | февраля | | |  | 2023 г. | | |
|  | | | | | | | | | | | | | | |
| Задание получил | | | |  | | | | |  | | Д.Э. Афанасьев | | | | | |
|  | | | | подпись | | | | |  | | И.О. Фамилия | | | | | |
|  | | | | |  | | |  |  |  | | |  | | |
| Дата представления работы руководителю | | | | |  | | |  | мая | | |  | 2023г. | |
|  | | | | | | | | | | | | | | | |
| Руководитель курсовой работы | | | |  | | | | |  | | З.А. Бахвалова | | | | | |
|  | | | | подпись | | | | |  | | И.О. Фамилия | | | | | |

**Содержание**

[1 Формирование требований 5](#_Toc135817543)

[1.1 Описание предметной области 5](#_Toc135817544)

[1.2 Описание проблемы 5](#_Toc135817545)

[1.3 Цель 5](#_Toc135817546)

[1.4 Постановка задачи 5](#_Toc135817547)

[1.5 Словари 6](#_Toc135817548)

[1.6 Функциональность ПО 8](#_Toc135817549)

[2 Анализ функциональных требований 8](#_Toc135817550)

[2.1 Модель предметной области (IDEF0) 8](#_Toc135817551)

[2.2 Диаграмма функций 11](#_Toc135817552)

[2.3 Описание потоков данных 13](#_Toc135817553)

[2.4 Концептуальная модель хранилища данных 15](#_Toc135817554)

[2.5 Описание вариантов использования 16](#_Toc135817555)

[Вариант использования «Получение сведений о наличии» 16](#_Toc135817556)

[Вариант использования «Добавить в список интересующих изданий» 18](#_Toc135817557)

[Вариант использования «Формирование заявки на поставку» 19](#_Toc135817558)

[2.6 Диаграммы состояний 21](#_Toc135817559)

[2.7 Описание интерфейса 22](#_Toc135817560)

[3 Проектирование 25](#_Toc135817561)

[3.1 Инструменты 25](#_Toc135817562)

[3.2 Реализация 31](#_Toc135817563)

[3.3 Описание алгоритма 31](#_Toc135817564)

[3.4 Интерфейс 34](#_Toc135817565)

[4 Тестирование 35](#_Toc135817566)

[4.1 Исходный код авто тестов 35](#_Toc135817567)

[5 Документация пользователя 37](#_Toc135817568)

[Заключение 40](#_Toc135817569)

[Список использованных источников 41](#_Toc135817570)

**Введение**

Современные технологии продолжают вносить существенные изменения во все сферы нашей жизни, и розничная торговля не исключение. Вместе с ростом популярности электронных книг и онлайн-платформ для чтения, казалось бы, традиционные книжные магазины должны уступить место виртуальным аналогам. Однако, они все еще имеют свое место в нашем обществе, предлагая не только доступ к печатным изданиям, но и уникальный опыт ощущения и запаха книг.

Один из главных вызовов, с которым сталкиваются книжные магазины, заключается в обеспечении эффективного и удобного обслуживания своих клиентов. Посетители магазинов часто испытывают трудности в поиске нужной книги, что приводит к необходимости обращаться к кассирам с вопросами о наличии и расположении конкретного издания. Это отнимает время и отвлекает персонал магазина от основной задачи – обслуживания покупателей.

Для решения этой проблемы предлагается создание приложения для поиска информации о наличии и расположении книги в книжном магазине. Данное приложение будет работать локально внутри магазина и позволит клиентам самостоятельно находить интересующие их книги, не обращаясь к кассирам. Это сократит время, затрачиваемое на поиск и улучшит общее качество обслуживания в магазине.

Целью данной курсовой работы является разработка данного решения. Будут рассмотрены технические аспекты создания приложения, такие как выбор платформы и инструментов разработки, а также функциональные возможности, необходимые для удобного использования приложения клиентами магазина.

Отдельное внимание будет уделено изучению предметной области и ее проблем, описанию необходимых функциональных требований к предлагаемому решению, анализу этих требований с различных сторон, а также проектированию приложения и хранилища данных, с учетом проделанного анализа.

Именно эти минимальные работы необходимы в современном мире для создания качественного программного обеспечения. Будущим специалистам в области разработки программных продуктов необходимо научиться выполнять перечисленные работы верно, чтобы стать действительно востребованным и ценным кадром.

# 1 Формирование требований

## 1.1 Описание предметной области

Заказчик – директор небольшого книжного магазина, занимающегося продажей печатных изданий. Магазин занимается продажей печатных изданий, которые характеризуются названием, автором, описанием, изображением обложки, жанром, годом выпуска, стоимостью, количеством копий в наличии, расположением в магазине (либо один из отделов, либо склад). Издания с одинаковым названием различают по автору. Необходимо автоматизировать процесс получения посетителем информации о печатных изданиях и исключить или минимизировать участие продавца в этом процессе. В магазине используется платформа «1С: Предприятие» для учета товаров и взаимодействия с клиентами. В магазине работает всего несколько сотрудников: директор магазина, занимающийся заказами поставок, два продавца, работающих посменно, уборщица и внештатный специалист по «1С: Предприятие».

## 1.2 Описание проблемы

Посетители магазина часто ищут конкретные печатные издания. Для того, чтобы узнать, есть ли это издание в наличии в данном магазине, посетители обращаются к кассиру. В магазине существует внутренняя система учета товаров и кассир может произвести поиск информации о запрашиваемом издании. Однако, кассир не всегда имеет возможность выполнять такой поиск, так как может быть занят обслуживанием покупателей в очереди. При этом возникает большая вероятность, что человек, который ищет издание покинет магазин. Это приводит к потере потенциальной прибыли и лояльности посетителя. Кроме того, директор магазина хочет узнавать, какие отсутствующие издания посетители хотели бы купить, чтобы повысить продажи.

Предлагаемое решение – разработка приложения, позволяющего посетителям производить поиск имеющихся в магазине печатных изданий и оставлять заявки на закупку отсутствующих изданий.

## 1.3 Цель

Избежать потери прибыли и повысить лояльность клиентов за счет частичной автоматизации процесса получения информации об изданиях.

## 1.4 Постановка задачи

Программа должна представлять собой часть распределенной информационной системы магазина (включающей «1С Предприятие») и будет установлена на находящийся в торговом зале компьютер с ОС Windows.

Программа должна работать с данными, содержащимися в уже имеющейся системой учета издания, созданной на основе «1С: Предприятия». Посетители магазина будут производить поиск издания по названию, программа на экране будет отображать список изданий (даже если нашлось всего одно издание), либо показывать сообщение об отсутствии данных об издании в системе.

В списке пользователь должен будет выбрать один из предложенных вариантов, после чего откроется страница выбранного издания.

На странице издания должна отображаться информация об издании: название, автора, год издания, изображение обложки (если есть в системе), жанр, описание, стоимость, количество в наличии и расположение в магазине. У пользователя должна быть возможность формировать список из выбранных им изданий, чтобы сохранять результаты всех поисков в одном месте. Кроме того, должна быть возможность вернуться к окну поиска.

Если издания нет в наличии, пользователь должен иметь возможность подать заявку на поставку этого издания. Для подачи заявки пользователь должен предоставить свои контакты (телефон и/или почта), также должна быть возможность оставить свое имя. Приложение сохранит данные о заявке в системе, включая издание, контакты и имя пользователя, а также дату создания заявки.

Полученные заявки будут использоваться 1С специалистом магазина при заказе поставок изданий в магазин. Контакты будут собираться для дальнейшего оповещения людей, оставляющих заявки для рассылки оповещений о поступлении изданий. Такая рассылка происходит с использованием уже существующего модуля на базе платформы «1С: Предприятие», поэтому разрабатываемая программа должна использовать базу данных этой платформы. Автоматизацией процессов переноса данных для поставок и рассылки оповещений заниматься не нужно (в штат магазина входит 1С специалист, занимающийся этим).

## 1.5 Словари

**Словарь предметной области**

|  |  |  |
| --- | --- | --- |
| **№** | **Сущность** | **Описание** |
| 1 | Издание/печатное издание | Печатная продукция, продающаяся в магазине. |
| 2 | Заявка | Зафиксированное в системе желание посетителя купить конкретное отсутствующее издание. |
| 3 | Жанр | Категория, к которой относится издание |
| 4 | Стоимость | Цена печатного издания, выраженная в рублях |
| 5 | Наличие | Количество экземпляров данного издания в наличии |
| 6 | Расположение в магазине | Конкретные стеллаж и полка, на которых расположено издание в магазине. Записываются в специальном 4-х символьном формате, где 2 первых символа – буквы, идентификатор стеллажа, последние 2 символа – цифры, номер полки на стеллаже (напр. АА01) |
| 7 | Список интересующих изданий | Формируемый посетителем список изданий с информацией об их расположении в магазине. Список нужен для сохранения результатов сбора информации об изданиях в едином месте. Подразумевается, что пользователь сможет сфотографировать его или записать при необходимости. |
| 8 | Стеллаж | Стеллаж в магазине, имеющий идентификатор из 2 букв |
| 9 | Полка | Одна из полок стеллажа. Обладает числовым номером. |

**Словарь по Абботу**

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **Сущность** | **Глагол** | **Пояснение глагола** |
| 1 | Издание/печатное издание | Искать, отображать |  |
| 2 | Заявка | Формировать, сохранять |  |
| 3 | Жанр | Узнавать |  |
| 4 | Стоимость | Узнавать |  |
| 5 | Наличие | Узнавать |  |
| 6 | Расположение в магазине | Узнавать |  |
| 7 | Список изданий | Формировать, распечатывать |  |
| 8 | Стеллаж | - |  |
| 9 | Полка | - |  |

**Объектно-ориентированный словарь**

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **Сущность** | **Данные** | **Методы** |
| 1 | Издание/печатное издание | Название, автор, жанр, цена, наличие в магазине, расположение в магазине | Искать, отображать |
| 2 | Заявка | Издание, имя, номер телефона, электронная почта, дата | Сохранять |
| 3 | Жанр | Может быть:   1. Детектив 2. Роман 3. Драма 4. Комедия 5. Фантастика 6. Классика 7. Бизнес 8. Наука и техника 9. Философия 10. и т. д. | Узнавать |
| 4 | Стоимость | Цена в рублях | Узнавать |
| 5 | Наличие | Количество в шт. | Узнавать |
| 6 | Расположение в магазине | Идентификатор стеллажа и номер полки | Узнавать |
| 7 | Список изданий | Название издания, расположение в магазине | Формировать, распечатывать |
| 8 | Стеллаж | Идентификатор |  |
| 9 | Полка | Числовой номер |  |

## 1.6 Функциональность ПО

**Проверка наличия издания в магазине**

Пользователь должен иметь возможность проверить, есть ли в наличии то или иное издание. Для этого он должен произвести поиск сведений об издании по названию. Информация о наличии должна представлять из себя количество изданий в штуках, которое можно купить в данный момент в магазине. Если количество изданий равно нулю, то считается, что издание отсутствует. Если искомого издания вовсе нет в системе, это означает, что издание не учтено.

**Определение расположения изданий в магазине**

Пользователь должен иметь возможность узнать на какой полке это издание находится. Информация о расположении – это идентификатор стеллажа (напр. АА) и номер полки на этом стеллаже (напр. 01), записанные вместе. Если издание отсутствует, информацию о расположении получить должно быть нельзя (хотя в системе учета сохраняется информация о последнем расположении).

**Формирование списка интересующих посетителя изданий, имеющихся в магазине**

Пользователь должен иметь возможность формировать список нужных ему изданий по мере поиска нужной ему информации о наличии и расположении изданий, чтобы в удобной форме получить доступ к собранной информации.

**Оставление посетителем заявки на поставку изданий, которые он хочет купить**

Пользователь должен иметь возможность сформировать заявку на поставку изданий в случае, если издания отсутствуют в наличии, то есть их количество равно нулю. При этом, он оставляет свои контактные данные (почта и email) и имя. Заявка должна сохраняться в системе. Заявки не могут быть оставлены на неучтенные издания.

# 2 Анализ функциональных требований

# 2.1 Модель предметной области (IDEF0)

Модель, представленная на рисунке 1 описывает процесс покупки книг в книжном магазине.

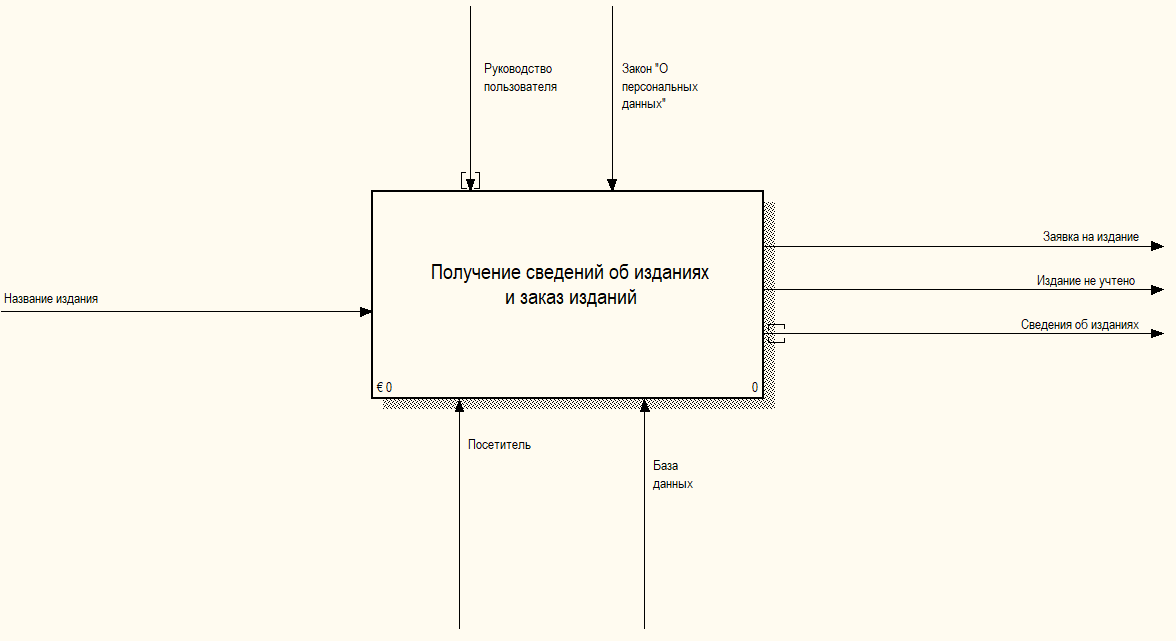


Рисунок 1 – Основной процесс приложения. Уровень A0

Целевыми сведениями, получаемыми в ходе выполнения процесса, является информация о наличии и расположении изданий в магазине. Заявка на издание может быть сделана только для тех изданий, которые отсутствуют в магазине во время выполнения процесса, то есть этот результат процесса может быть получен не всегда.

На рисунке 2 представлено описание декомпозиции модели первого уровня.

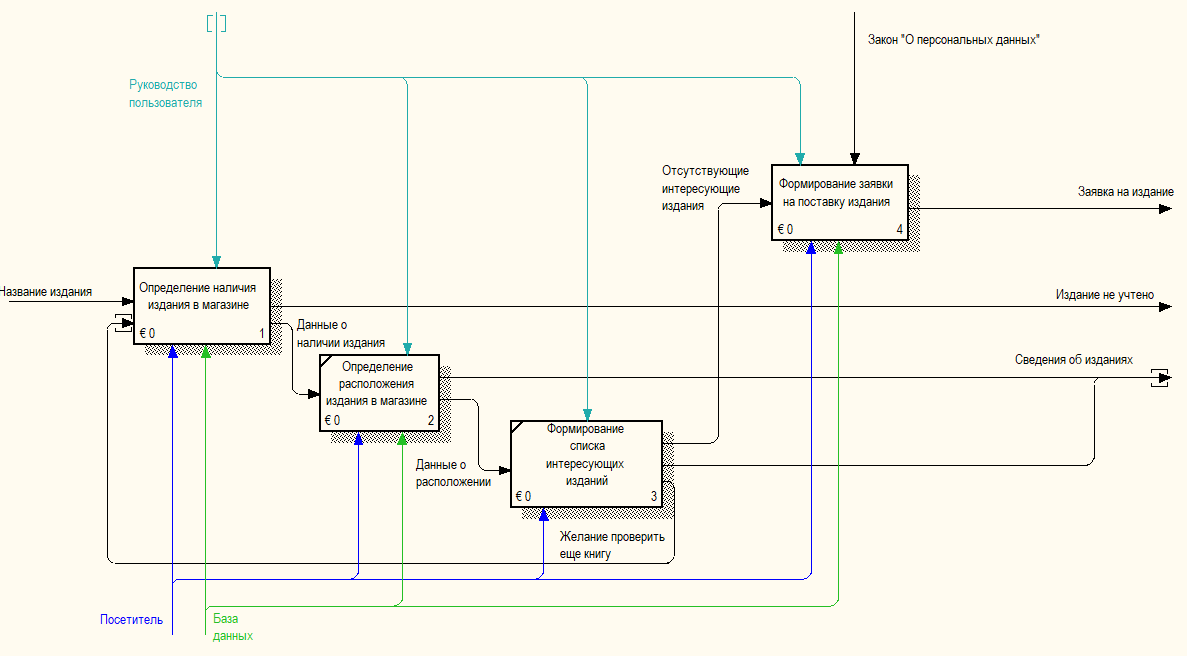


Рисунок 2 – Декомпозиция процесса получения сведений об издании и оставления заявки. Уровень А1

Процесс покупки книг в магазине включает 4 подпроцесса. При желании купить конкретное издание, посетитель, используя программу, проверяет наличие издания в магазине. Из всех необходимых посетителю изданий он может сформировать список в той же программе, содержащий информацию о наличии и расположении изданий и используемый далее при поиске изданий на полках. В списке могут присутствовать как издания в наличии, так и отсутствующие, для отсутствующих изданий может быть сформирована заявка от посетителя.

Подразумевается, что собранные с помощью программы сведения, помогут посетителю быстрее найти и собрать издания с полок и при этом не отвлекать кассира от обслуживания других покупателей.

На рисунке 3 описывается декомпозиция процесса определения наличия издания в магазине.

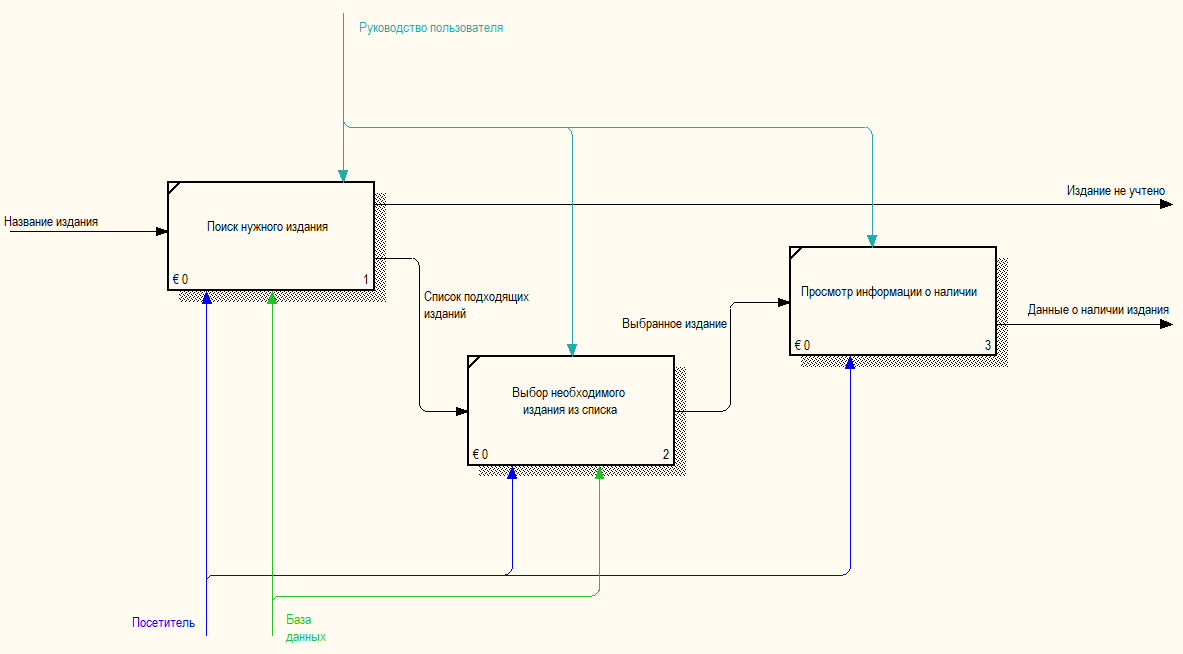


Рисунок 3 - Декомпозиция процесса определения наличия издания. Уровень А2

Чтобы у компьютера не скапливались очереди, а посетители могли быстро производить все необходимые действия с программой, все этапы должны быть не загружены действиями. Таким образом, поиск будет производится по самому минимальному требуемому набору характеристик, а уже из списка будет выбираться необходимое издание. Например, поиск по названию и быстрый просмотр нескольких вариантов будет гораздо быстрее, чем заполнение целой формы, включающей название, авторов, жанры, тип обложки и других возможных характеристик.

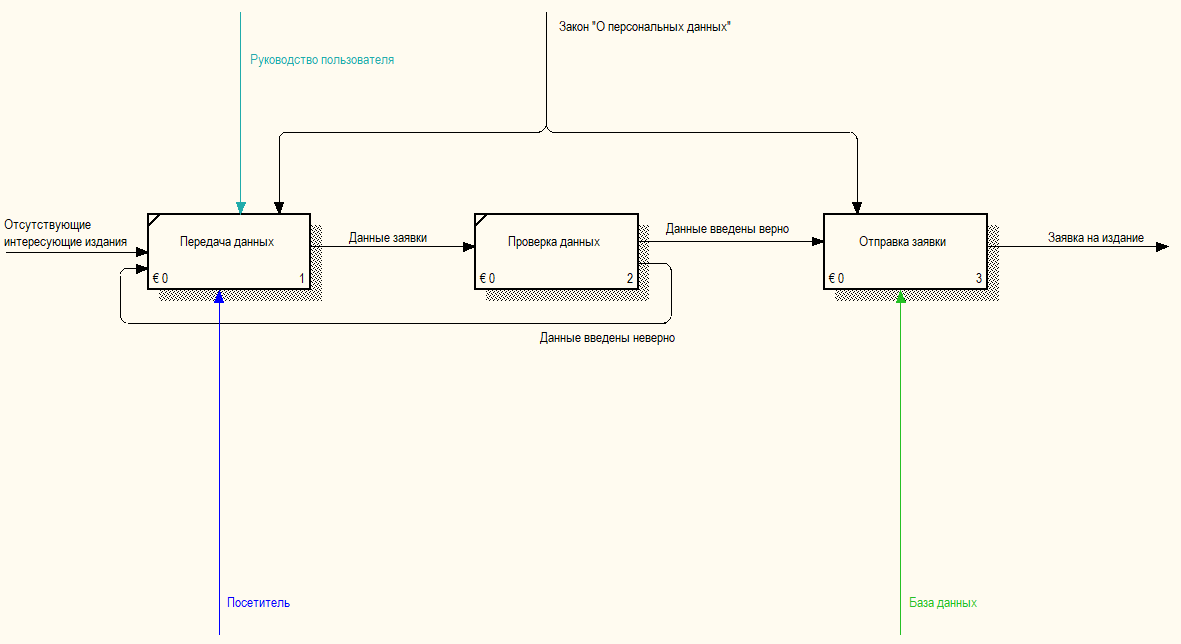


Рисунок 4 – Декомпозиция процесса формирования заявки на отсутствующие издания. Уровень А2

Передача данных совершается в привычном для приложений формате (форма). Данные заявки включают в себя данные посетителя (имя, контактные данные) и перечень изданий, которые выбрал пользователь. Проверка происходит только для данных пользователя и включает в себя проверку того, что данные не пусты и формат контактных данных (телефона и почты).

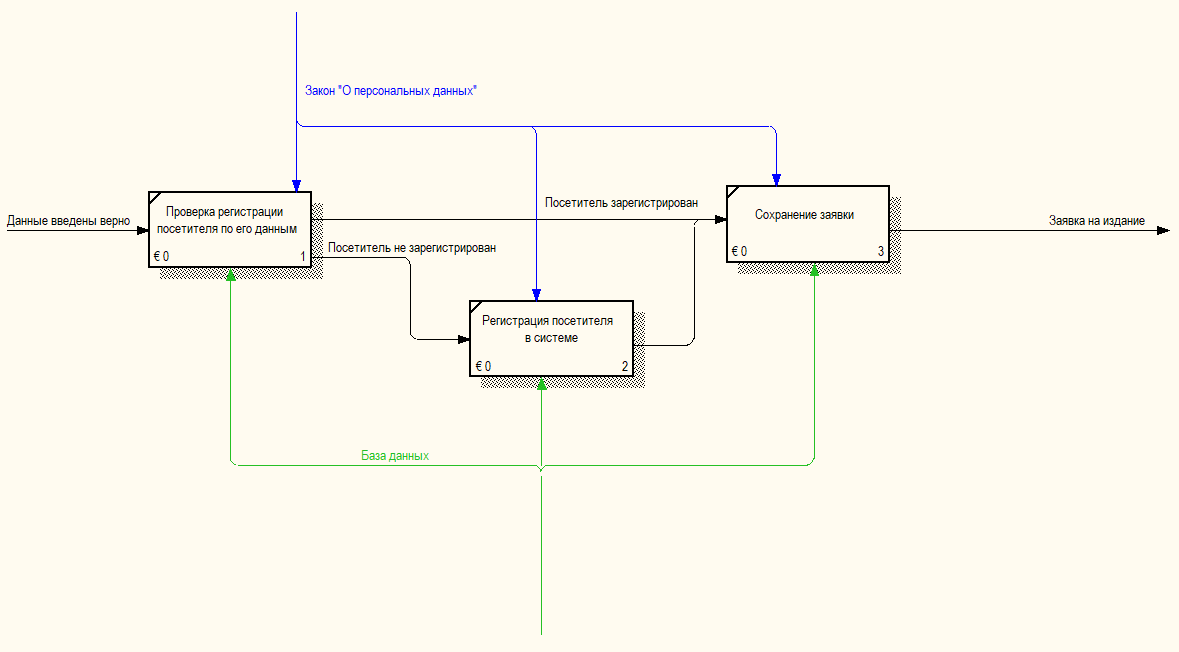


Рисунок 5 – Декомпозиция процесса отправки заявки. Уровень A3

# 2.2 Диаграмма функций

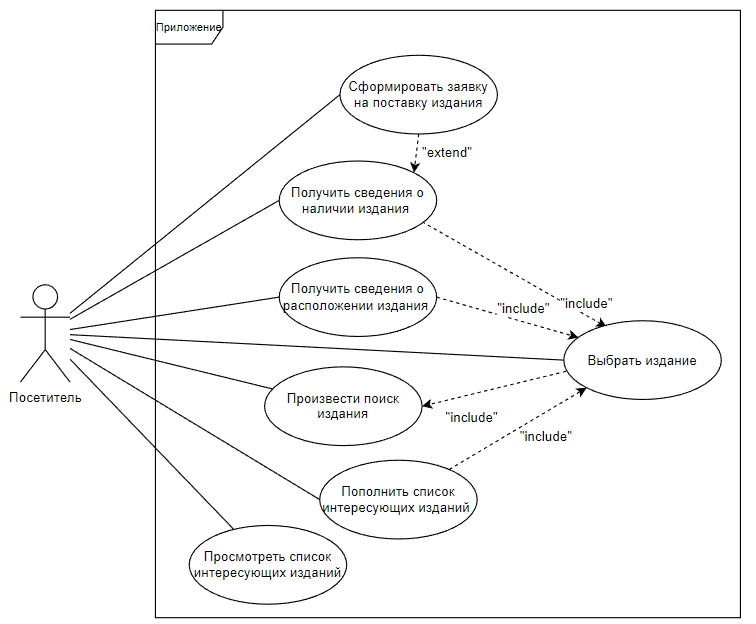


Рисунок 5 – Диаграмма, описывающая функции системы

Функции посетителя магазина соответствуют процессам, описанным в модели предметной области.

Поскольку все функции приложения тесно связаны с изданиями, выбор издания – первоочередная функция посетителя, ведь именно с выбранным изданием он и будет работать в приложении.

Функции продавца и администратора системы не описаны, так как они выполняются в рамках других процессов, отличных от тех, что рассматриваются здесь. Кроме того, они слабо связаны с разрабатываемым приложением.

# 2.3 Описание потоков данных

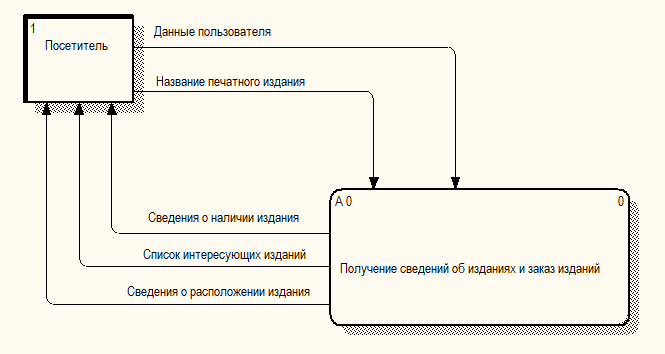


Рисунок 6 – Диаграмма, описывающая потоки данных процесса получения сведений об изданиях и заказа изданий. Уровень A0

Помимо посетителя, внешней сущностью является система учета на основе «1С: Предприятие», но, поскольку части этой системы учета используются внутри процесса, то обмен данными с ней будет рассмотрен при декомпозиции.

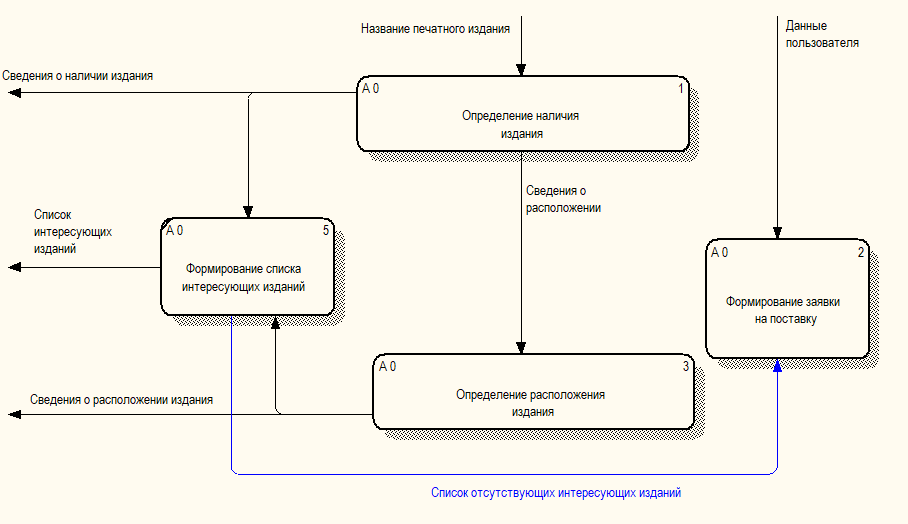


Рисунок 7 - Диаграмма, описывающая потоки данных процесса получения сведений об изданиях и заказа изданий. Уровень A1

Как и было упомянуто ранее, добавилась дополнительная внешняя сущность, которая получает заявки для дальнейшей рассылки уведомлений.

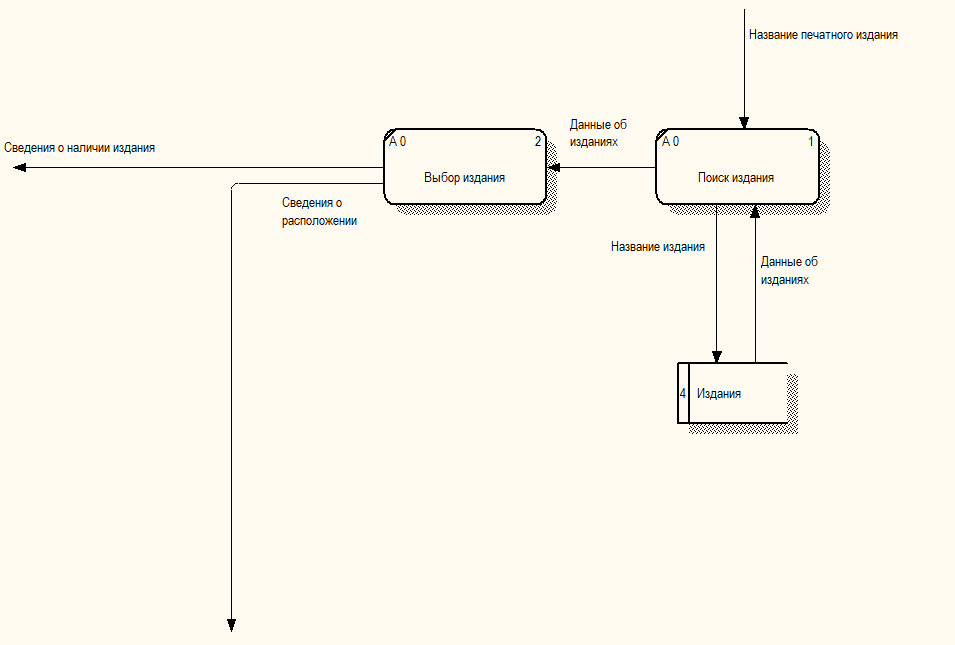


Рисунок 8 - Диаграмма, описывающая потоки данных процесса определения наличия издания. Уровень A2

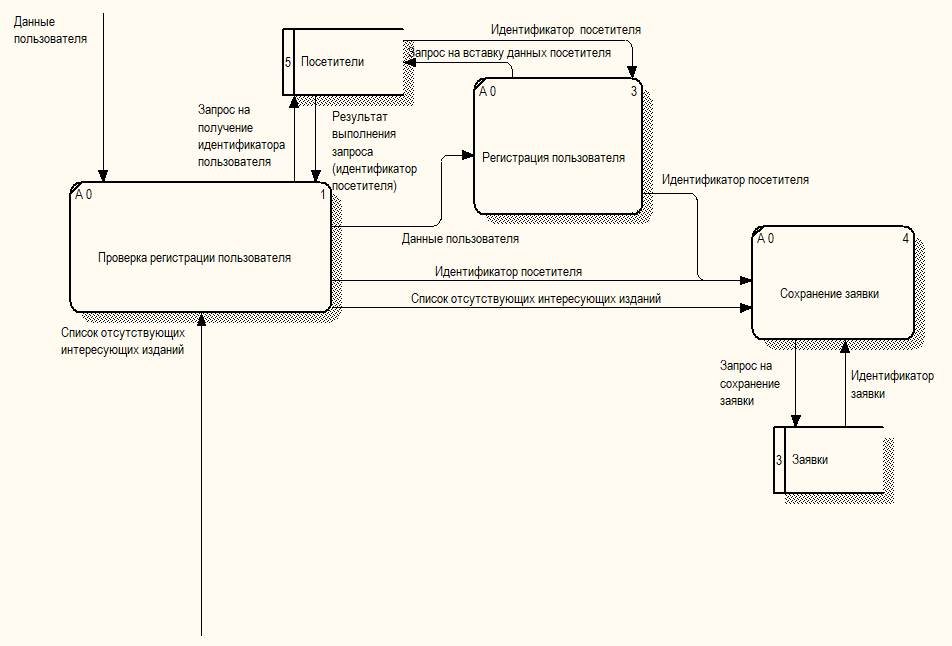


Рисунок 9 - Диаграмма, описывающая потоки данных процесса формирования заявки на поставку. Уровень A2

После проверки данных, либо посетитель регистрируется в системе (если посетитель не был зарегистрирован до этого), либо используются его уже зарегистрированные данные (если посетитель зарегистрирован в системе).

# 2.4 Концептуальная модель хранилища данных

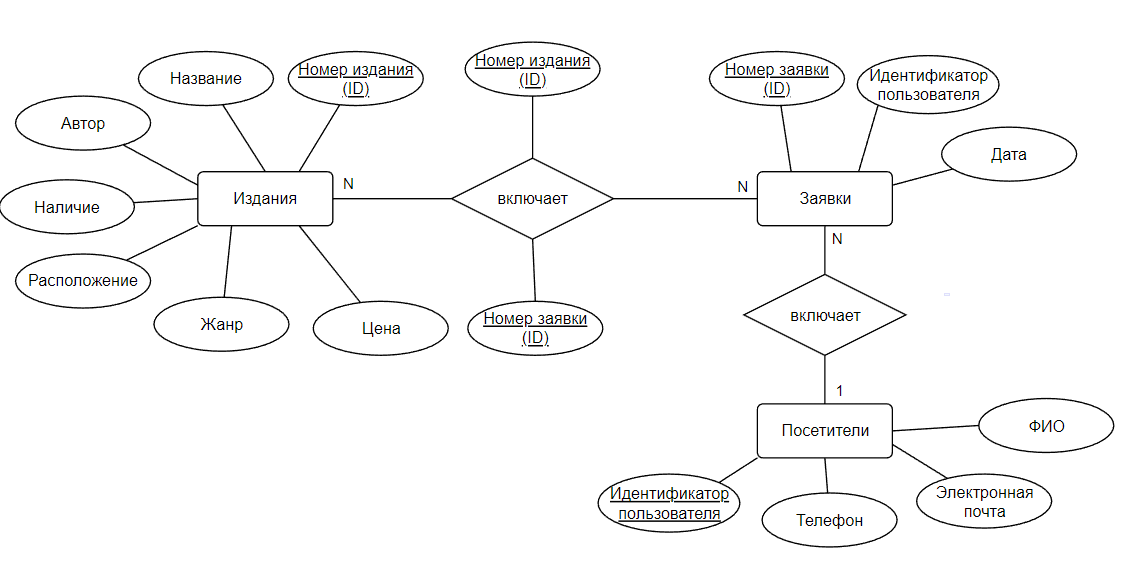


Рисунок 10 – ER-модель хранилища данных

Поскольку хранилище данных уже существует в виде реляционной базы данных, на диаграмме показана только та ее часть, которая будет использоваться приложением.

# 2.5 Описание вариантов использования

## Вариант использования «Получение сведений о наличии»

Спецификация варианта использования «Получить сведения о наличии издания»

**Цель:** Узнать, в наличии ли издание

**Активные субъекты:** Посетитель

**Краткое описание:** Активный субъект может получить информацию о наличии интересующего его издания в данном магазине

**Предусловия:** нет.

**Постусловия:** нет.

**Основной поток событий:**

1. Посетитель производит поиск издания по названию, используя форму поиска
2. Приложение запрашивает у БД список подходящих под название изданий
3. Приложение отображает в интерфейсе полученный список
4. Посетитель выбирает нужное издание в списке
5. Приложение считывает данные о выбранном издании
6. Приложение отображает в интерфейсе подробную информацию о выбранном издании, включая наличие издания
7. Посетитель просматривает информацию о наличии
8. Конец варианта использования

**Альтернативные потоки событий:** нет.

**Сценарий обработки ошибок:** нет.

**Специальные требования:** нет.

**Дополнительные замечания:** нет.

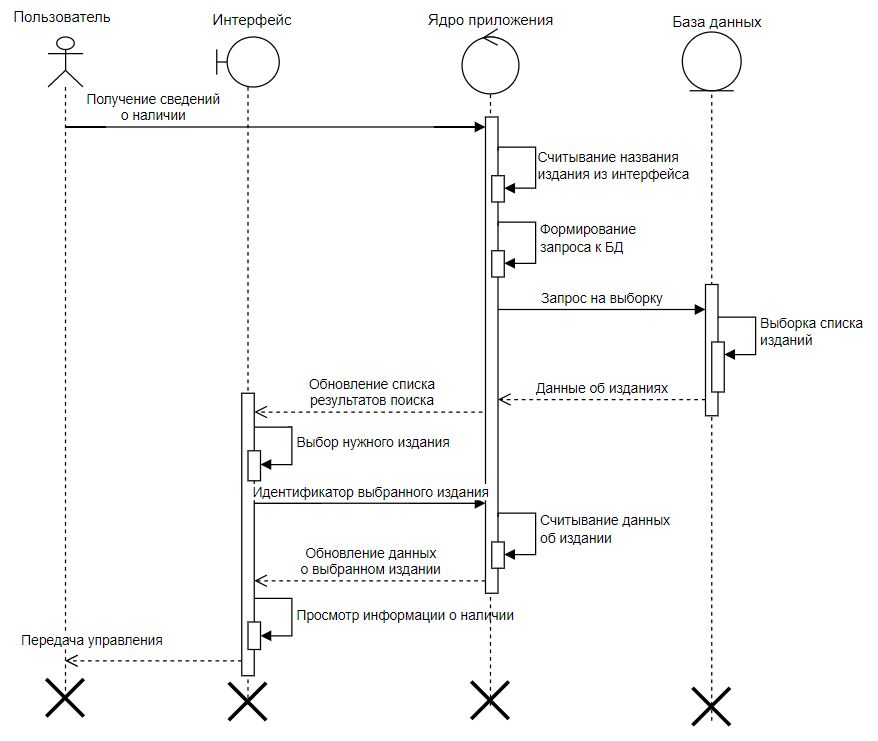


Рисунок 11 - Диаграмма последовательностей варианта использований *«Получить сведения о наличии издания»*

## Вариант использования «Добавить в список интересующих изданий»

Спецификация варианта использования «Добавить в список интересующих изданий»

**Цель:** Сохранить результат поиска сведений об издании

**Активные субъекты:** Посетитель

**Краткое описание:** Активный субъект может добавлять издания в список, чтобы сохранять результаты получения сведений о наличии и расположении.

**Предусловия:** должен быть выполнен вариант использования «Получить сведения о наличии издания» или «Получить сведения о расположении издания».

**Постусловия:** список должен быть очищен.

**Основной поток событий:**

1. Посетитель инициирует вариант использования
2. Приложение считывает данные о выбранном издании (название, наличие, расположение)
3. Приложение добавляет новый элемент в список интересующих изданий, содержащий считанные данные
4. Приложение обновляет список интересующих изданий, отображаемый на странице
5. Конец варианта использования

**Альтернативные потоки событий:** нет.

**Сценарий обработки ошибок:** нет.

**Специальные требования:** нет.

**Дополнительные замечания:** нет.

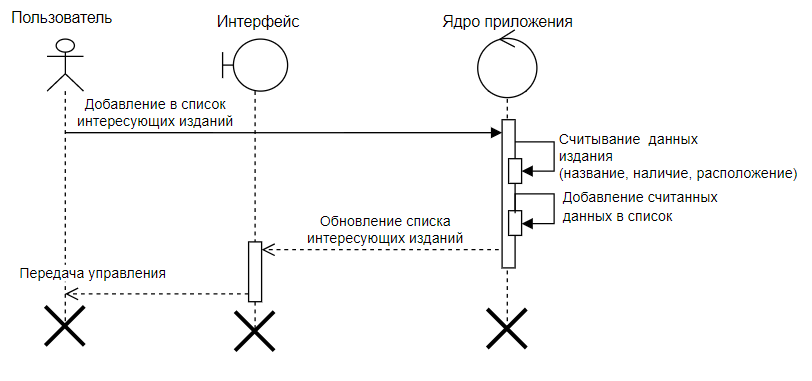


Рисунок 12 - Диаграмма последовательностей варианта использований *«Добавить в список интересующих изданий»*

## Вариант использования «Формирование заявки на поставку»

Спецификация варианта использования «Сформировать заявку на поставку издания»

**Цель:** запросить у магазина заказать отсутствующие интересующие издания.

**Активные субъекты:** Посетитель

**Краткое описание:** Активный субъект запросить поставку издания и уведомить себя, оставив свои контактные данные.

**Предусловия:** в списке интересующих изданий должны быть добавлены отсутствующие издания.

**Постусловия:** нет.

**Основной поток событий:**

1. Посетитель открывает форму создания заявки на отсутствующие интересующие издания
2. Посетитель заполняет контактную информацию
3. Посетитель отправляет заявку
4. Приложение считывает идентификаторы отсутствующих интересующих изданий
5. Приложение считывает введенные пользователем контактные данные
6. Приложение запрашивает у БД данные о регистрации посетителя (идентификатор посетителя) по введенным контактам
7. Приложение проверяет регистрацию посетителя (если идентификатор есть – значит зарегистрирован). Если посетитель не зарегистрирован, активируется альтернативный поток событий 1.
8. Приложение отправляет запрос на сохранение заявки в БД
9. Конец варианта использования

**Альтернативные потоки событий:**

A1:

7. Приложение отправляет запрос на регистрацию посетителя в системе

8. Приложение отправляет запрос на сохранение заявки в БД

9. Конец варианта использования

**Сценарий обработки ошибок:** нет.

**Специальные требования:** нет.

**Дополнительные замечания:** нет.

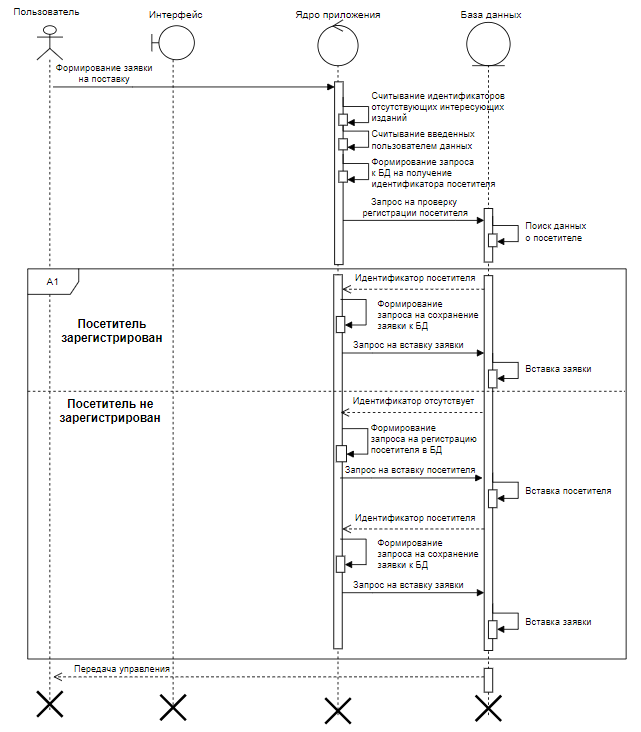


Рисунок 13 - Диаграмма последовательностей варианта использований *«Сформировать заявку на поставку издания»*

# 2.6 Диаграммы состояний

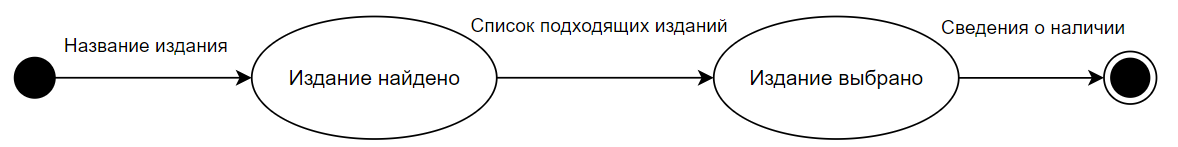


Рисунок 14 – Переходы между состояниями в варианте использования «Получение сведений о наличии»

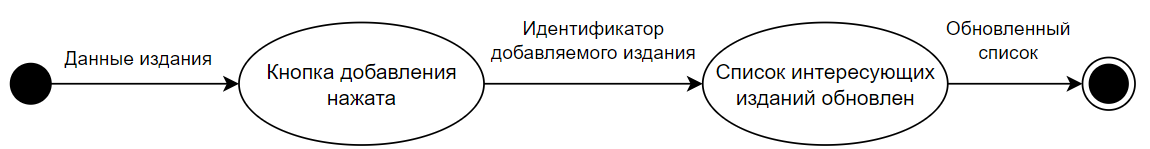


Рисунок 15 - Переходы между состояниями в варианте использования «Формирование списка интересующих изданий»

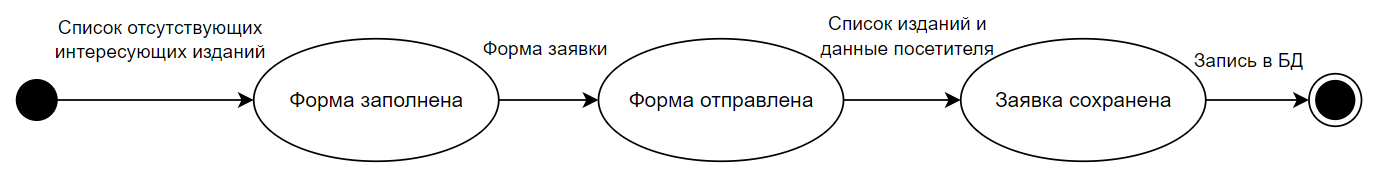


Рисунок 16 - Переходы между состояниями в варианте использования «Формирование заявки на поставку издания»

# 2.7 Описание интерфейса

Таблица 4 – Описание элементов интерфейсов

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Окно** | **Функция** | **Элемент интерфейса** | **Описание элемента** | **Переход** |
| Окно поиска | Получение сведений об издании | Поле ввода названия | Поле, в которое пассажир вводит название интересующей книги | - |
| Получение сведений об издании | Кнопка поиска | После ввода названия запускается поиск по этому названию с помощью кнопки | - |
| Получение сведений об издании | Таблица результатов поиска | Таблица, в которой отображается список найденных изданий (название, автор, жанр). Каждый элемент списка позволяет перейти в окно издания | - |
| Сохранение интересующих изданий | Таблица сохраненных изданий | Таблица, в которой отображается список выбранных пользователем изданий (название, наличие, расположение). Каждый элемент списка позволяет перейти в окно издания | - |
| Сохранение интересующих изданий | Кнопка обнуления списка | Кнопка, очищающая таблицу выбранных пользователем изданий | - |
| Получение сведений об издании | Текстовое поле основных данных | Поле, содержащее название и автора издания | - |
| Получение сведений о наличии издания | Текстовое поле наличия | Поле, содержащее информацию о наличии издания в магазине | - |
| Получение сведений о расположении издания | Текстовое поле расположения | Поле, содержащее информацию о расположении издания в магазине | - |
| Создание заявки на поставку издания | Кнопка создания заявки | Кнопка, отображающаяся только при отсутствии издания и открывающая модальное окно с формой заявки | Модальное окно заявки |
| Модальное окно заявки | Создание заявки на поставку издания | Форма заявки | Форма с полями для контактных данных и кнопкой отправки заявки | - |
| Создание заявки на поставку издания | Кнопка закрытия формы | Закрывающая окно заявки кнопка | Окно поиска |

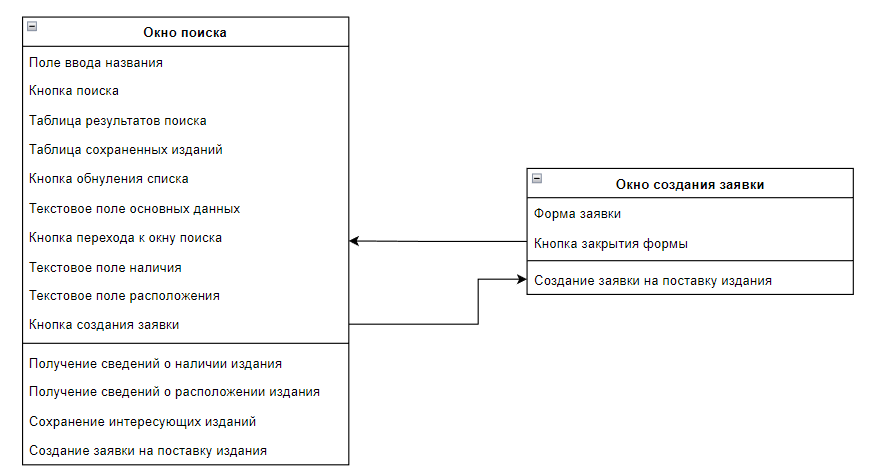


Рисунок 17 – Диаграмма интерфейсных классов

На рисунке 17 изображена диаграмма интерфейсных классов, где отображены главные окна и их наполнение и функции, выполняемые на этих окнах. Есть 2 обычных окна (окно поиска, окно издание) и 1 модальное окно, открывающееся в окне издания (модальное окно создания заявки).

Почти все функции приложения могут быть выполнены на основном окне, кроме функции отправки заявки. Это необходимо для того, чтобы работа с приложением проходила более быстро. Таким образом, очереди к приложению не будут скапливаться, или будут малы.

## 3 Проектирование

## 3.1 Инструменты

ПО будет является десктоп-приложением, но работать на основе веб технологий. Будет использован инструментарий программных средств:

1. **Electron** - это фреймворк для разработки кросс-платформенных приложений с использованием веб-технологий, таких как HTML, CSS и JavaScript. Поскольку для меня привычнее работать именно с веб стеком, это позволит максимально быстро создать десктоп-приложение.
2. **Bootstrap** - фреймворк для создания эстетически приятного и отзывчивого пользовательского интерфейса без написания собственных стилей CSS. Снимет необходимость в задании собственных стилей элементов интерфейса, ускорив разработку.
3. **jQuery** - библиотека JavaScript, заметно упрощающая манипуляции с элементами HTML и создание интерактивности. Помимо этого, позволит более просто совершать HTTP запросы.
4. **Node.js** - платформа для выполнения JavaScript-кода вне браузера, которая обеспечивает возможность написания ядра приложения. Позволит использовать фреймворк для написания ядра приложения.
5. **Express.js** - гибкий фреймворк на платформе Node.js. Значительно ускорит разработку ядра приложения за счет уже созданной логики.
6. **Playwright** – библиотека, позволяющая провести тестирование интерфейса. Самая удобная для тестирования приложений electron.

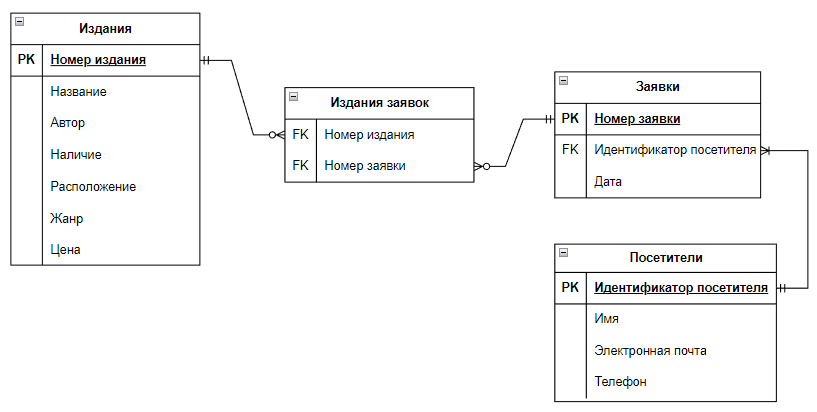


Рисунок 18 – Логическая модель базы данных

Как видно из логической модели базы данных, между таблицами изданий и заявок была разрешена связь «многие-ко-многим», тем самым достигнута 3 форма нормализации.

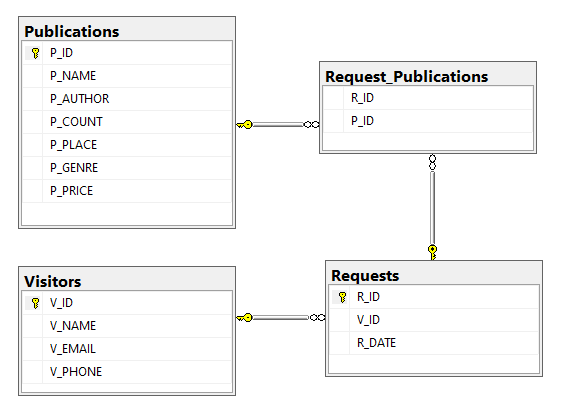


Рисунок 19 – Физическая модель базы данных

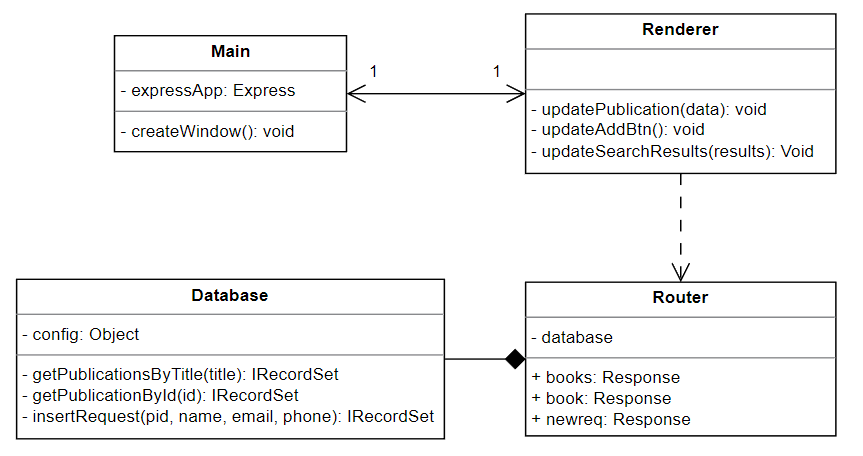


Рисунок 20 – Взаимодействие классов приложения

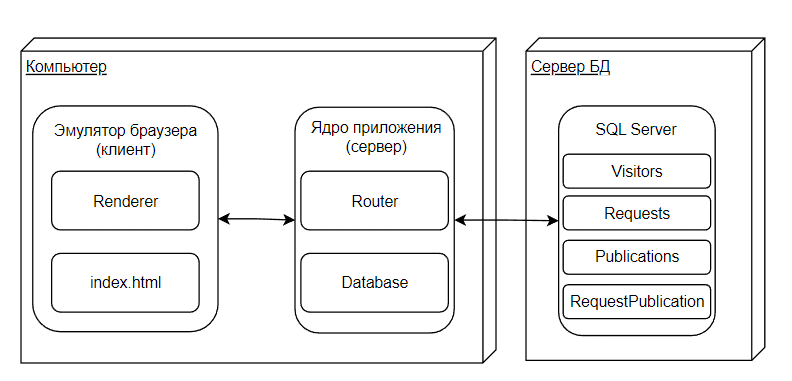


Рисунок 21 – Диаграмма развертывания приложения

Среда работы приложения – компьютер, расположенный в книжном магазине. Electron будет запускать эмулятор браузера в окне Windows, через этот интерфейс пользователь будет взаимодействовать с ядром приложения, являющимся аналогом серверной части в классических веб-архитектурах. Само ядро будет общаться с СУБД для обмена данными, база данных расположена на отдельном компьютере в локальной распределенной сети магазина.

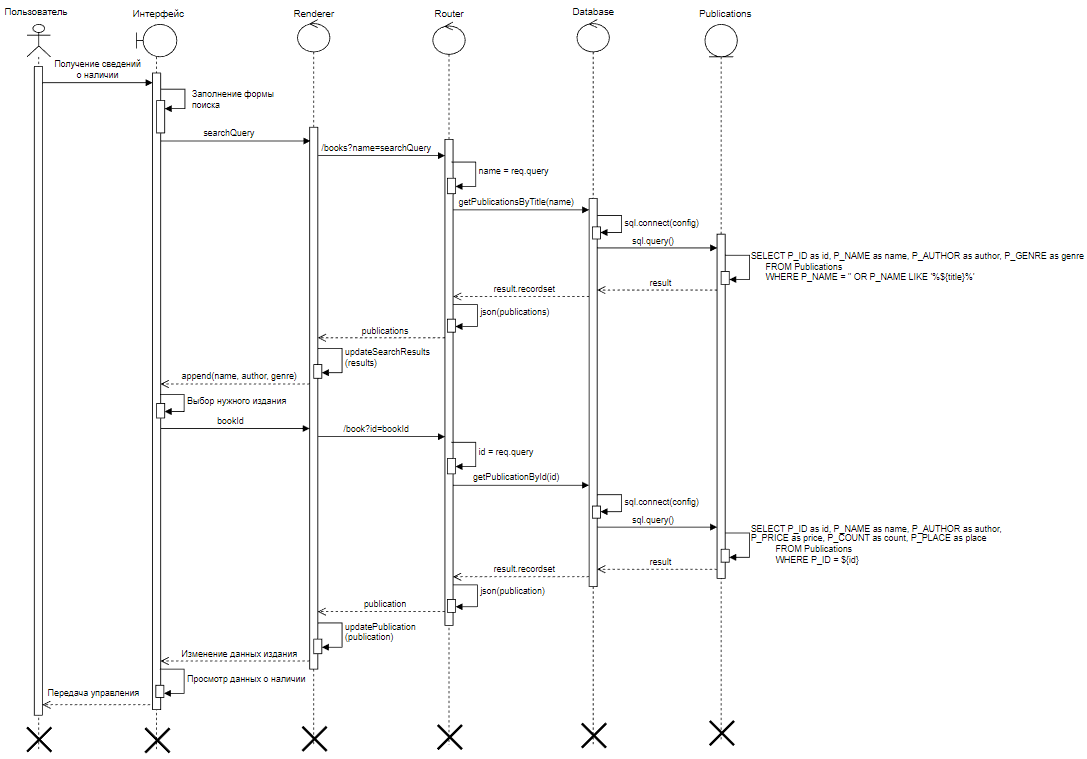


Рисунок 22 – Диаграмма, описывающая последовательность действий при выполнении функции «Получение сведений о наличии»

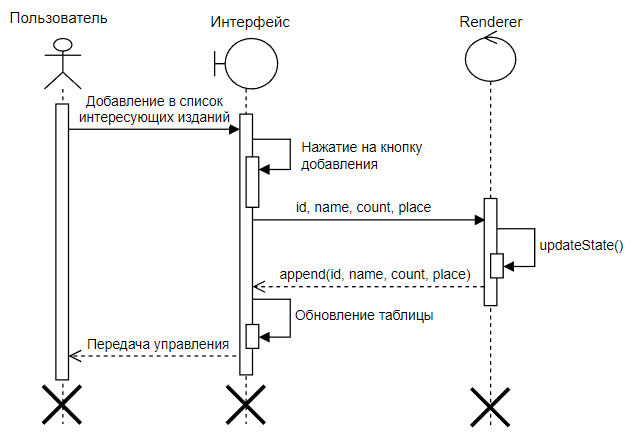


Рисунок 23 – Диаграмма, описывающая последовательность действий при выполнении функции «Формирование списка интересующих изданий»

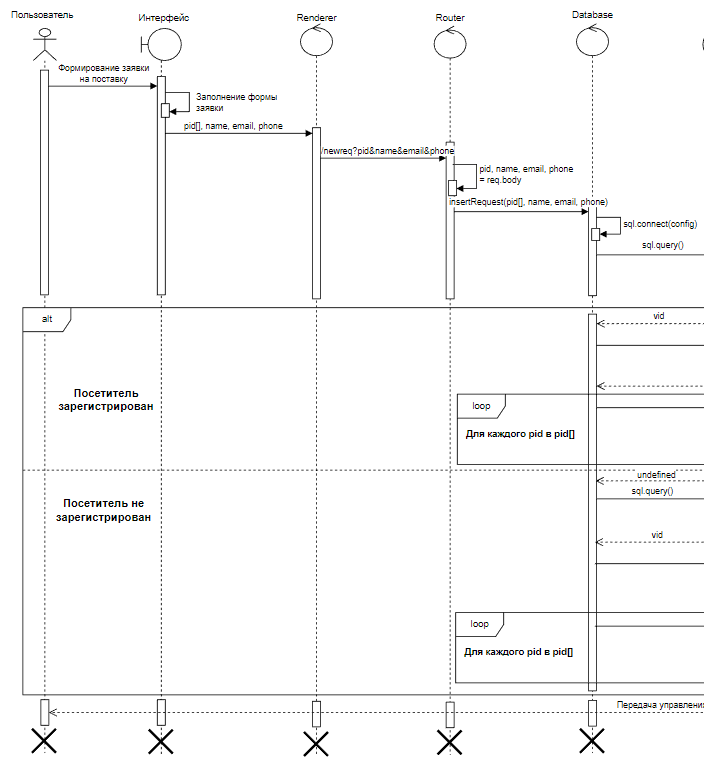


Рисунок 24 – Диаграмма, описывающая последовательность действий при выполнении функции «Формирование заявки на поставку». Часть 1

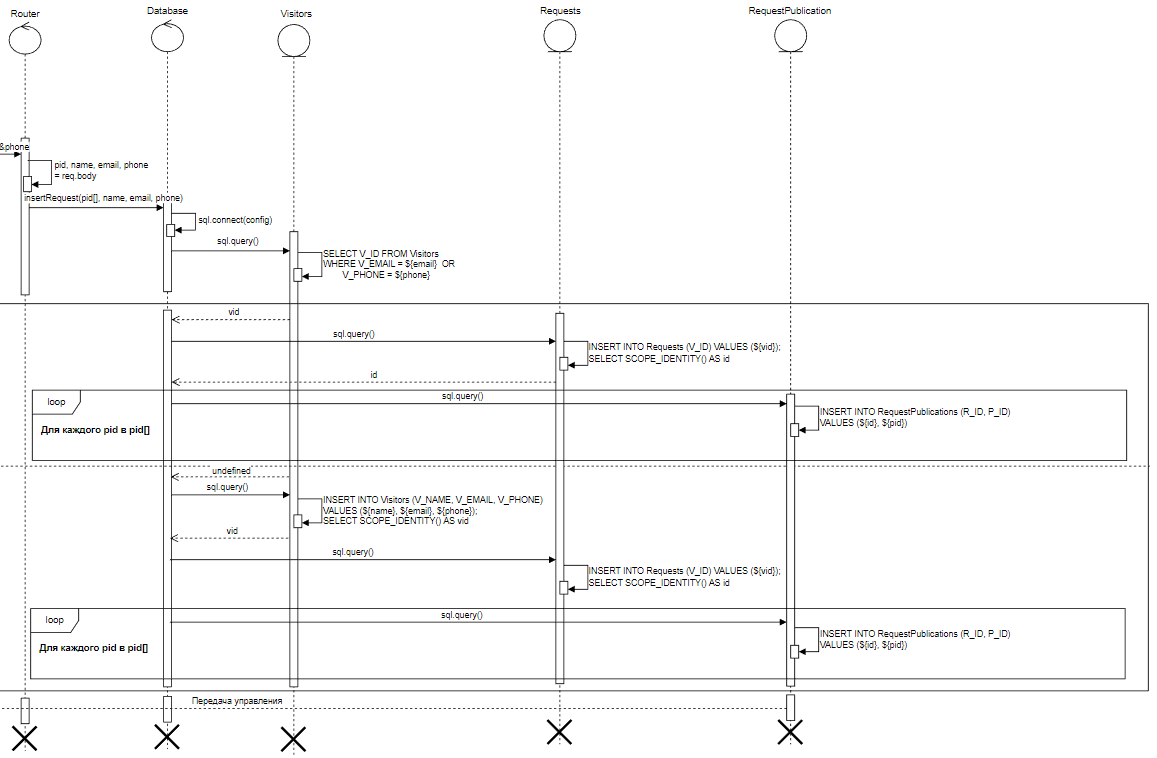


Рисунок 24 – Диаграмма, описывающая последовательность действий при выполнении функции «Формирование заявки на поставку». Часть 2

# 3.2 Реализация

**Main** – точка входа в приложение, отвечает за создание окна приложения и запуск ядра приложения.

**Renderer** – контроллер, отслеживает и изменяет интерфейс, обменивается данными с ядром приложения.

**Database** содержит методы для взаимодействия с базой данных SQL Server. Является частью ядра приложения, то есть моделью.

**Router** – маршрутизатор запросов, отправляемых из контроллера. Маршруты представлены в таблице 5. Так же является частью модели MVC.

В архитектуре MVC в качестве представления выступает файл разметки index.html.

Таблица 5 – Спецификация функций-обработчиков ядра приложения

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Маршрут | Метод | Параметры | Описание |
| 1 | /books | GET | name – название издания | Получение списка изданий, полностью или частично совпадающих с name |
| 2 | /book | GET | id – идентификатор издания | Получение полной информации о конкретном издании, включая наличие и расположение |
| 3 | /newreq | POST | pid – идентификаторы изданий  name – имя пользователя  email – эл. почта пользователя  phone – телефон пользователя | Сохранение заявки на поставку изданий, созданной пользователем |

## 3.3 Описание алгоритма

Приложение условно разделено на 2 части. Это разделение похоже на стандартное деление в веб проектах: frontend и backend (ядро приложения). Пользователь взаимодействует с интерфейсом, в ответ на это взаимодействие, контроллер совершает тот или иной запрос в ядро приложения для получения или сохранения данных. После получения ответа от ядра, контроллер изменяет содержимое интерфейса. Ответы приходят в формате JSON.

Таким образом производятся 3 основные функции: получение сведений о наличии, получение сведений о расположении, формирование заявки на поставку издания.

Протокол общения между frontend и ядром – HTTP.

Для выполнения функции формирования списка интересующих изданий, ядро приложения не задействуется, список формируется только на стороне frontend путем простого добавления строк в таблицу.

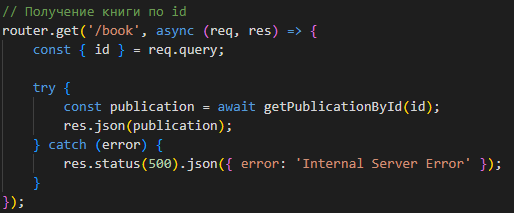


Рисунок 25 – Функция-обработчик, получающая данные из БД

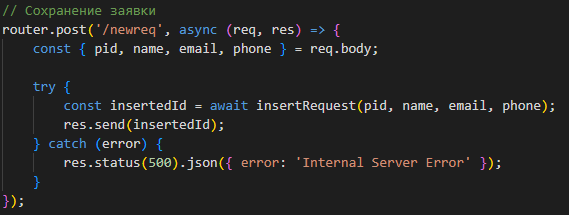


Рисунок 26 – Функция-обработчик, создающая запись в БД



Рисунок 27 – Обработчик, совершающий вставку в список выбранных изданий

## 3.4 Интерфейс

Макет интерфейса были созданы в программе Figma

Ссылка на макет интерфейса: (<https://www.figma.com/file/ywfkrHjxoXuHWRZwvo0rXI/trpk-coursework?type=design&t=qFDDErqvfVOF3Sq1-6>)

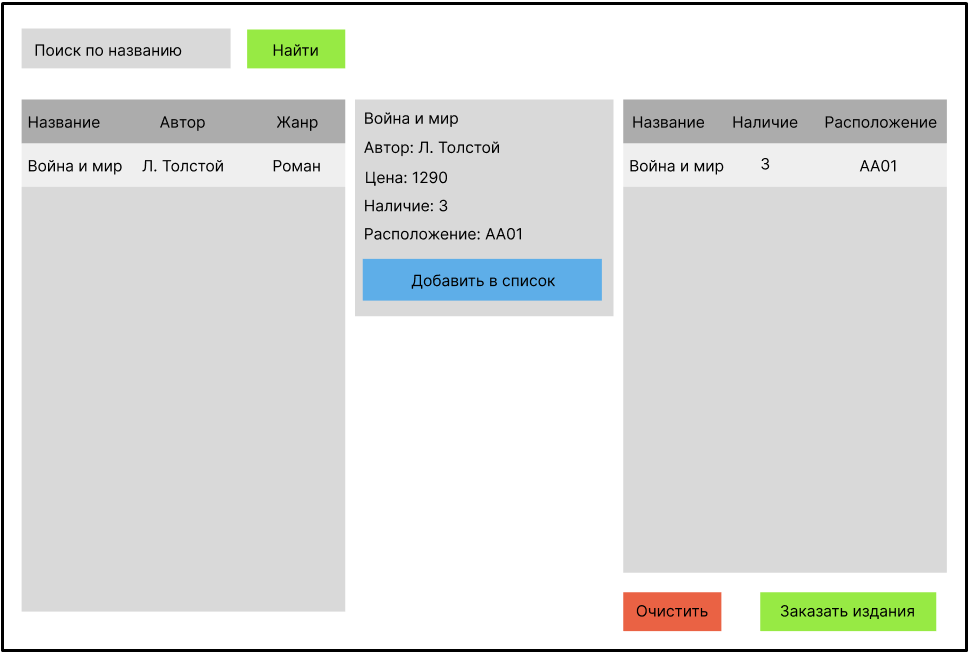


Рисунок 28 – Основное окно

Слева расположена таблица результатов поиска изданий, строки можно нажимать, таким образом в центральной части экрана появляется подробная информация о выбранном издании. Нажав на кнопку «Добавить в список» Можно добавить выбранное издание в список интересующих изданий, расположенный справа. Этот список можно очистить, нажав на кнопку под списком. Рядом с кнопкой очистки расположена кнопка заказа изданий, открывающая модальное окно с формой отправки заявки.

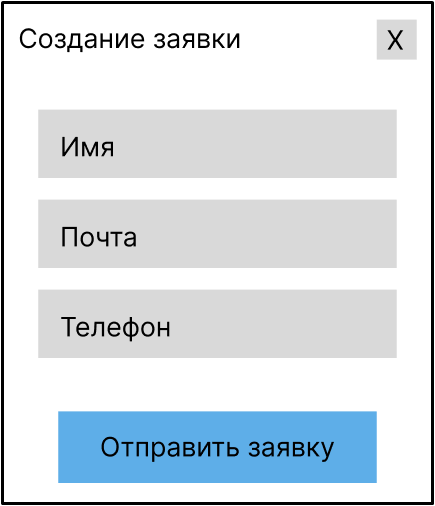


Рисунок 29 – Модально окно с формой заявки

## 4 Тестирование

## 4.1 Исходный код авто тестов

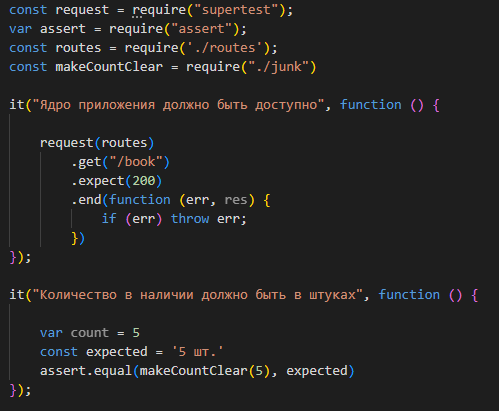


Рисунок 30 – Автоматические юнит и интеграционный тесты

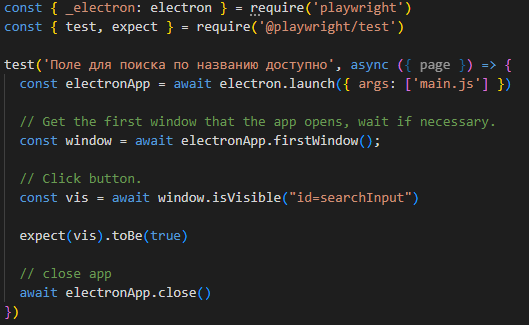


Рисунок 31 – Автоматический тест интерфейса

Таблица 6 – Описание авто тестов

|  |  |  |
| --- | --- | --- |
| № | Тип авто теста | Описание |
| 1 | Unit тест | Проверяет корректность преобразования количества в строку наличия |
| 2 | Интеграционный тест | Проверяет доступ к ядру приложения |
| 3 | End2End тест | Проверяет доступ к приложению, используя встроенный браузер |

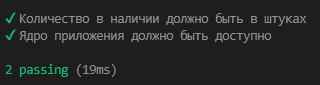


Рисунок 32 – Результат выполнения unit теста и интеграционного теста



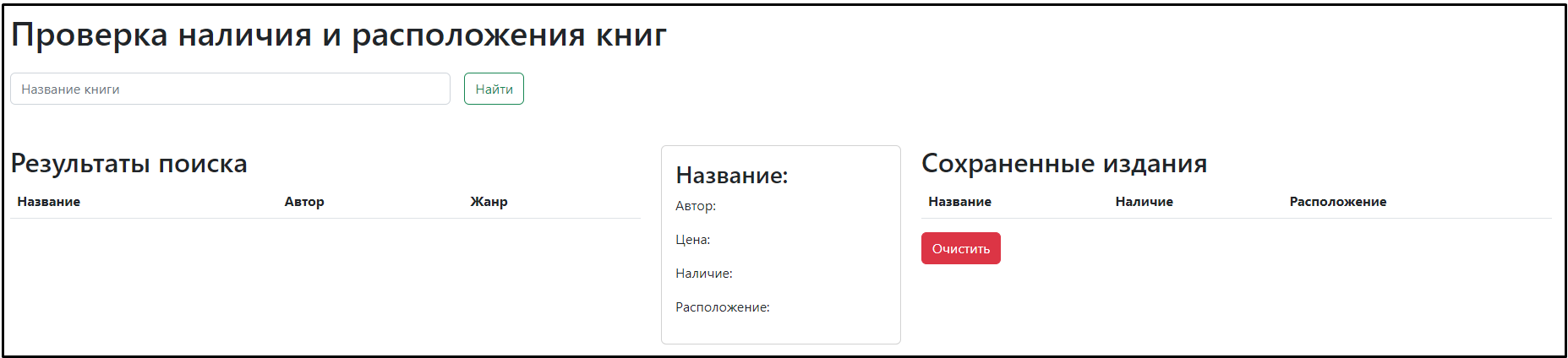
Рисунок 33 – Результат End2End теста используя Selenium

(Проверка доступности содержимого главной страницы)

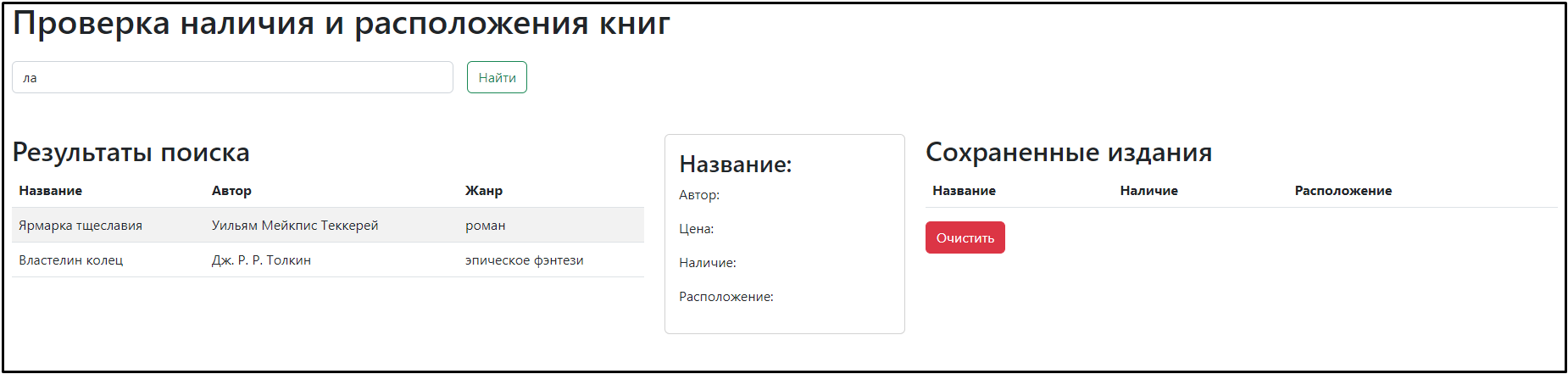
## 5 Документация пользователя

**Получение сведений о наличии и расположении издания в магазине**

По умолчанию перед вами открыто основное окно (окно поиска):



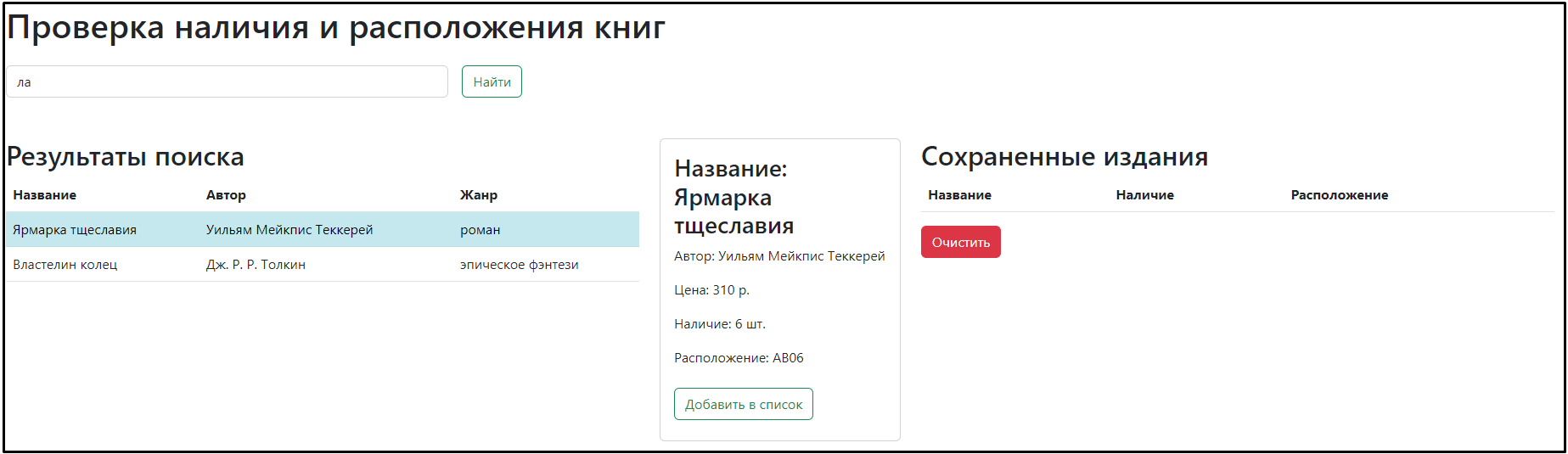
В верхней части окна находится форма для поиска интересующего вас издания. Для того, чтобы произвести поиск, необходимо ввести название книги в специальное поле, подписанное «Название книги», а затем нажать на кнопку «Найти» справа от этого поля.



*Пометка: вы можете ввести лишь часть названия, поиск будет произведен по этой части.*

В таблице, находящейся слева и подписанной как «Результаты поиска» вы увидите список найденных изданий. Здесь отображены все подошедшие под название или его часть издания.

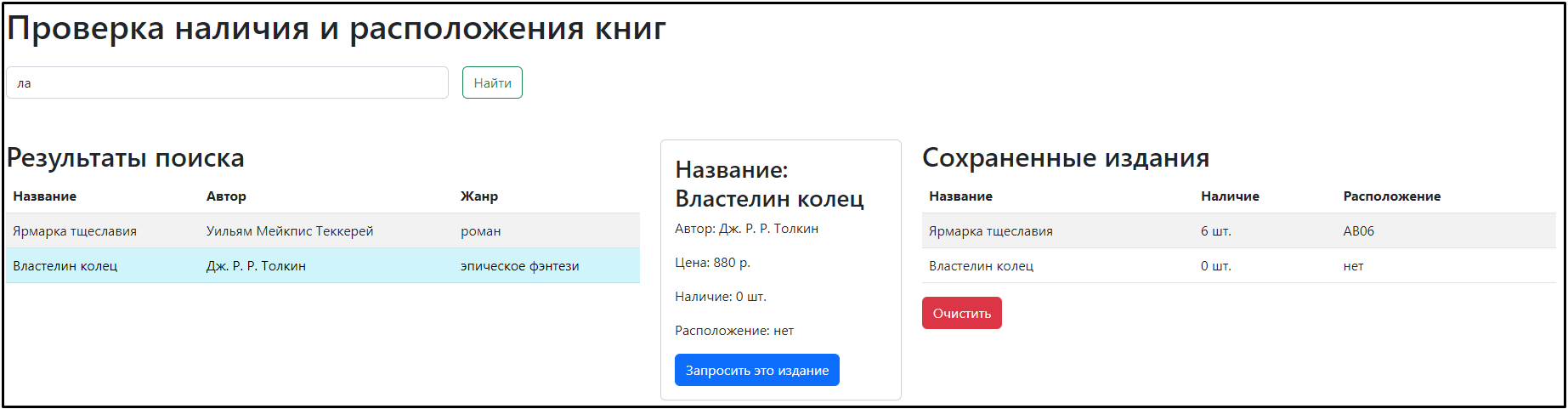
Дальше необходимо выбрать то издание, которое вы искали. Чтобы определить какое из изданий вам нужно, сравните авторов и жанры, а затем нажмите на соответствующую строку в таблице. Строка подсветится голубым цветом.



После выбора нужного вам издания подробная информация о нем отобразится в центральной части окна. Здесь будет находиться следующая информация: название, автор, цена, наличие (количество в штуках), расположение в магазине. Буквы в строке расположения обозначают стеллаж (все стеллажи в магазине подписаны). Цифры обозначают номер полки на стеллаже.

**Формирование списка интересующих изданий**

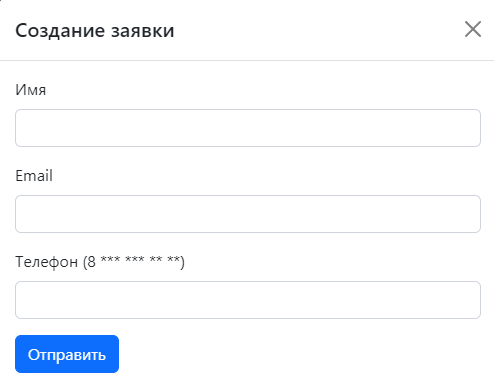
Если вы хотите собрать информацию о нескольких изданиях, добавляйте каждое из изданий в специальный список, позволяющий вам сохранить найденную вами информацию. Для того, чтобы сделать это, нажмите кнопку «Добавить в список», расположенной под информацией об издании в центральной части окна. После этого издание добавиться в список справа. Этот список содержит название, наличие и расположение каждого из добавленных в него изданий. Так вы сможете, например, сфотографировать этот список на телефон, или записать его вручную.



Если вы передумали использовать эту информацию или уже сохранили список для себя, очистите его, пожалуйста, нажав красную кнопку «Очистить» под списком.

**Формирование заявки** **на отсутствующее издание**

Если вы обнаружили, что изданий нет в наличии, вы можете оставить заявку на их закупку. Чтобы сделать это, нажмите кнопку «Оставить заявку», после чего перед вами откроется форма отправки заявки.



Здесь вам необходимо заполнить все перечисленные поля: ваши имя, электронная почта (email), номер телефона. После этого, если все введенные данные верны, достаточно нажать кнопку «Отправить», чтобы отправить заявку. Как только издание прибудет в магазин, вы будете оповещены по оставленным контактным данным.

**FAQ**

1. В: Я не могу найти свое издание, что мне делать?

О: Скорее всего, мы никогда не продавали это издание, поэтому наша система не знает о нем. Вскоре мы добавим функцию заказа таких изданий, а пока обратитесь для этого к менеджеру или кассиру.

1. В: Почему я не могу отправить заявку?

О: Тщательно перепроверьте введенные вами данные. Все поля должны быть обязательно заполнены. Вот примеры вводимых электронной почты и номера телефона: example@ex.ru,

89991112233

1. В: Почему я не могу сохранить издание в список?

О: Скорее всего, это издание уже добавлено в список сохраненных изданий. Если вы уверены, что это не так, обратитесь к менеджеру или кассиру.

По любым вопросам можете обращаться к менеджеру или кассиру, они обязательно вам помогут!

## Заключение

В ходе данной работы было проведено проектирование системы приложения для поиска информации о наличии и расположении книг в книжном магазине. Этот процесс включал определение и описание основных функциональных требований, анализ архитектуры приложения, разработку базы данных и пользовательского интерфейса.

Цель работы была достигнута, задачи выполнены. Было разработано приложение, позволяющее:

1. получать посетителям магазина информацию об интересующих их изданиях, включая информацию о наличии и информацию о расположении этих изданий в магазине;
2. составлять список из интересующих изданиях, чтобы сохранять всю найденную посетителями информацию об изданиях;
3. оставлять заявки на поставку изданий, которых нет в наличии на данный момент в магазине.

Все реализованные функции гораздо повысят лояльность клиентов, так как позволяют намного быстрее совершать необходимые в магазине действия, по сравнению с тем, когда приложения, аналогичного разработанному, нет. Кроме того, это позволяет кассирам выполнять свою основную работу эффективнее, поскольку они не отвлекаются на консультирование посетителей.

Анализ требований и проектирование приложения помогли лучше понять потребности пользователей и определить необходимые компоненты системы. В ходе работы были построены различные диаграммы, относящиеся к самым важным аспектам приложения. Среди таких диаграмм были: диаграмма бизнес-процессов, диаграмма потоков данных, диаграмма вариантов использований, диаграмма последовательностей действий, концептуальная диаграмма хранилища данных, логическая и физическая модели базы данных, диаграммы развертывания, классов, интерфейса и состояний.

Благодаря использованию Playwright был получен опыт в автоматизации веб-браузеров и извлечении данных с веб-страниц, а также автоматического тестирования интерфейса.

Использование Bootstrap для разработки пользовательского интерфейса позволило создать эстетически приятный и отзывчивый дизайн.

Кроме того, было составлено руководство пользователя для разработанного приложения. При этом стало заметно, что проделанные ранее анализ и проектирование помогли структурировать это руководство и описать его более последовательно и детально.

В целом, проведенное проектирование системы, работа с выбранными технологиями и разработка приложения позволили расширить знания и навыки в различных областях, связанных с разработкой приложений, баз данных, десктопных приложения с использованием веб технологий.

## Список использованных источников

1. Electron Docs // Electron URL: https://www.electronjs.org/docs/latest/ (дата обращения: 10.05.2023).

2. Playwright Docs // Playwright URL: https://playwright.dev/docs/ (дата обращения: 18.05.2023).

3. Руководство // Expressjs URL: https://expressjs.com/ru/guide (дата обращения: 12.05.2023).

4. Учебное пособие по диаграммам последовательностей: полное руководство с примерами // creately URL: https://creately.com/blog/ru/диаграмма/учебное-пособие-по-последовательной/ (дата обращения: 12.05.2023).

5. Урок по диаграммам развертывания // Lucidcharts URL: https://www.lucidchart.com/pages/ru/uml-deployment-diagram (дата обращения: 15.05.2023).

6. СТО 0052020 СИСТЕМА МЕНЕДЖМЕНТА КАЧЕСТВА. Учебно- методическая деятельность. Оформление курсовых проектов (работ) и выпускных квалификационных работ технических специальностей (http://www.istu.edu/structure/57/2506/).

7. Буч Г. Объектно ориентированное проектирование с примерами применения: Пер. С англ. — М.: Конкорд.