Министерство образования и науки Российской Федерации

ФГБОУ ВО «БАЙКАЛЬСКИЙ ГОСУДАРСТВЕННЫЙ

УНИВЕРСИТЕТ»

Кафедра информатики и кибернетики

Междисциплинарная курсовая работа «Разработка программ»

*Разработка автоматизированной системы управления фондом библиотеки*

*ООО «Букинист»*

Автор:

Семеусова Алена Юрьевна

Группа БИ-15-1

Руководитель:

к.т.н., доцент

Артамонов И. В.

Иркутск

2017

**Оглавление**

[Введение 3](#_Toc485848769)

[Системный анализ 3](#_Toc485848770)

[Разработка проекта программы 6](#_Toc485848771)

[Диаграмма прецедентов 7](#_Toc485848772)

[Диаграмма классов 8](#_Toc485848773)

[Диаграмма действий 17](#_Toc485848774)

[Диаграмма состояний 17](#_Toc485848775)

[Диаграмма последовательностей 17](#_Toc485848776)

[Диаграмма размещения 23](#_Toc485848777)

[Диаграмма компонентов 23](#_Toc485848778)

[Диаграммы интерфейсов 23](#_Toc485848779)

[Пользовательский интерфейс 28](#_Toc485848780)

[Тестирование программы 28](#_Toc485848781)

[Заключение 37](#_Toc485848782)

[Список использованных источников 38](#_Toc485848783)

[Приложения 39](#_Toc485848784)

[Приложение 1. Техническое задание 39](#_Toc485848785)

[Приложение 2. Таблицы описания прецедентов 59](#_Toc485848805)

[Приложение 3. Пользовательский интерфейс 65](#_Toc485848806)

# Введение

Целью данной курсовой работы является проектирование информационной системы библиотеки, которая позволит автоматизировать деятельность ООО «Букинист».

К задачам курсовой работы относятся:

* Анализ предметной области;
* Разработка UML-диаграмм;
* Проектирование базы данных;

Объектом исследования является компания ООО «Букинист», занимающаяся библиотечной деятельностью.

Предметом исследования является процесс автоматизации управления фондами библиотеки.

Тема является актуальной, так как на сегодняшний день любая компания и библиотека в том числе, для эффективной организации своей деятельности нуждается в информационных технологиях.

Программа позволит вести учет читателей, пользователей системы, библиотечного фонда. Автоматизирует деятельность сотрудников по регистрации читателей, книг, выдаче прокатов, оказании услуг. Также обеспечит быстрый доступ к актуальной информации о текущем состоянии прокатов, оплате услуг, задолженностях читателей. Формируемый отчет о прибыли за указанный период позволит наглядно показать результаты работы организации. Эти функции помогут улучшить качество деятельности библиотеки и повысить эффективность работы сотрудников библиотеки ООО «Букинист».

# Системный анализ

ООО «Букинист» реализует библиотечную деятельность. Кроме сдачи в прокат книг, организация предоставляет ряд платных услуг. Основной задачей является отслеживание финансовых показателей работы библиотеки.

Работа с читателями организована следующим образом: каждый читатель имеет стандартные реквизиты - ФИО, адрес, телефон, залог. После идентификации определяется тип заказа: прокат книг или оказание дополнительной услуги. Читатель выбирает книги, виды услуг, время. После согласования данных с читателем заказ регистрируется.

Система должна предоставлять навигацию по категориям записей: прокаты, услуги, читатели, книги, авторы, сотрудники, типы книг, типы услуг. Информация должна предоставляться в виде таблиц. Создание и редактирование осуществляются в отдельных формах.

Каждый прокат характеризуется следующими параметрами:

* Идентификатор;
* Читатель;
* Дата начала проката;
* Дата возврата книг;
* Дата возврата книг по факту;
* Стоимость проката;
* Штраф;
* Оплаченная сумма;
* Книга;

Каждая услуга характеризуется следующими параметрами:

* Идентификатор;
* Читатель;
* Дата начала;
* Дата конца;
* Дата конца по факту;
* Стоимость услуги;
* Оплаченная сумма;
* Количество;
* Вид услуги;

Каждый читатель характеризуется следующими параметрами:

* Идентификатор;
* ФИО;
* Телефон;
* Адрес;
* Залог;

Каждая книга характеризуется следующими параметрами:

* Идентификатор;
* Название;
* Жанр;
* Дата издания;
* Дата регистрации;
* Издательство;
* Количество страниц;
* Оригинальное название;
* Количество томов;
* Стоимость книги;
* Стоимость проката (выражается в процентах от стоимости книги);
* Автор;
* Тип книги;
* Переводчик;
* Место хранения;
* Аннотация;

Каждый тип книги характеризуется следующими параметрами:

* Идентификатор;
* Название;

Каждый тип услуги характеризуется следующими параметрами:

* Идентификатор;
* Название;
* Стоимость;
* Описание;

Каждый объект отличается уникальным идентификатором. Одинаковых идентификаторов у объектов, принадлежащих одному классу, быть не может.

С данной информационной системой должны работать следующие группы пользователей:

* Администратор;
* Библиограф;
* Библиотекарь;

При работе с системой администратор должен иметь возможность решать следующие задачи:

* Просматривать, редактировать, добавлять и удалять пользователей системы;
* Составлять отчеты;
* Просматривать, редактировать, добавлять и удалять книги;
* Просматривать, редактировать, добавлять и удалять читателей;
* Просматривать, редактировать, добавлять и удалять авторов;
* Просматривать, редактировать, добавлять и удалять типы книг;
* Просматривать, редактировать, добавлять и удалять прокаты;
* Просматривать, редактировать, добавлять и удалять виды услуг;

При работе с системой библиограф должен иметь возможность решать следующие задачи:

* Просматривать, редактировать, добавлять и удалять книги;
* Просматривать, редактировать, добавлять и удалять авторов;
* Просматривать, редактировать, добавлять и удалять типы книг;
* Просматривать, редактировать, добавлять и удалять виды услуг;

При работе с системой библиотекарь должен иметь возможность решать следующие задачи:

* Просматривать, редактировать, добавлять и удалять прокаты;
* Просматривать, редактировать, добавлять и удалять читателей;
* Просматривать книги;
* Просматривать авторов;
* Просматривать виды услуг;

# Разработка проекта программы

При разработке проекта программы были использованы средства Visual Studio.

## **Диаграмма прецедентов**

Диаграмма прецедентов отображает отношения между актерами и прецедентами. Для рассматриваемой информационной системы, три актёра – администратор, библиотекарь, библиограф. Для каждого актёра определены прецеденты, которые он может реализовывать в системе (Рисунок 1).

Прецеденты администратора:

* просмотр списка пользователей с последующим редактированием, удалением или добавлением нового пользователя;
* просмотр списка прокатов с последующим редактированием, удалением;
* просмотр списка книг, с последующим редактированием, удалением;
* просмотр списка авторов с последующим редактированием, удалением;
* просмотр списка читателей с последующим редактированием, удалением;
* просмотр списка типов книг с последующим редактированием, удалением;
* просмотр списка видов услуг с последующим редактированием, удалением;

Прецеденты библиографа:

* просмотр списка книг, с последующим редактированием, удалением или добавлением новой книги;
* просмотр списка авторов с последующим редактированием, удалением или добавлением нового автора;
* просмотр списка типов книг с последующим редактированием, удалением или добавлением нового типа;
* просмотр списка видов услуг с последующим редактированием, удалением или добавлением новой услуги;
* Прецеденты библиотекаря:
* просмотр списка проката, с последующим редактированием, удалением или добавлением нового проката/услуги;
* просмотр списка читателей с последующим редактированием, удалением или добавлением нового читателя;
* просмотр списка книг;
* просмотр списка видов услуг;
* просмотр списка авторов;

## **Диаграмма классов**

Диаграмма классов отображает отношения между классами в системе и их содержимое. Для каждого класса определены процедуры, функции и конструкторы. Существует один абстрактный класс с атрибутами и виртуальными методами, реализуемыми каждым из их двух классов-наследников. Все классы являются открытыми и связаны между собой отношениями типа «один-к-одному», «один-ко-многим» (Рисунок 2).

***Person.*** Класс Person является предком классов Reader и User. Этот класс содержит атрибуты: FIO, ID, Phone, Status (сотрудник уволен или работает).

+add() — функция добавления человека в БД;

+save() — функция сохранения изменений.

***Reader.*** Данный класс наследует все операции класса Person. Этот класс содержит атрибуты Person и атрибуты: Address (домашний адрес), Deposit (залог клиента).

+Reader() — конструктор класса Reader без параметров;

+Reader (string ID, string FIO, string Phone, string Address, string Deposit, string Status) — конструктор класса Reader.

***allReaders.*** Класс, позволяющий работать со списком всех читателей.

+ FindAll(string sorting, Reader filtering, string sortingA, int count, int page) — выводит список всех читателей, sorting — сортировка по первому параметру, filtering — фильтрация, sorting — сортировка по второму параметру, count — количество строк на странице, page — текущая страница;

+ FindByID(int ID) — поиск читателя по ID;

+ GetMaxID() — получение максимального ID;

+ Getkolvo(string sorting, Reader filtering, string sortingA, int count, int page) — получение количества читателей, удовлетворяющих условиям запроса.

***User.*** Данный класс наследует все операции класса Person. Этот класс содержит атрибуты, описывающие отдельного работника и атрибуты: Position (должность), Login (личный код доступа).

+User() — конструктор класса User без параметров;

+User (string ID, string FIO, string Position, string Phone, string Login, string status) — конструктор класса User.

***allUsers.*** Класс, позволяющий работать со списком всех пользователей

+ FindAll(string sorting, User filtering, string sortingA, int count, int page) — выводит список всех пользователей, sorting — сортировка по первому параметру, filtering — фильтрация, sorting — сортировка по второму параметру, count — количество строк на странице, page — текущая страница;

+ FindByID(int ID) — поиск пользователя по ID;

+ GetMaxID() — получение максимального ID;

+ Getkolvo(string sorting, User filtering, string sortingA, int count, int page) — получение количества пользователей, удовлетворяющих условиям запроса.

***TypeOfService.*** Класс, отображающий типы дополнительных услуг. Атрибуты: ID, Describtion, Name, Status, Price.

+add() — функция добавления типа в БД;

+save() — функция сохранения изменений;

+TypeOfService() — конструктор класса TypeOfService без параметров

+ TypeOfService (string ID, string Name, string Descrip, string Price, string Status) — конструктор класса TypeOfService.

***allTypeOfService.*** Класс, позволяющий работать со списком всех типов дополнительных услуг

+ FindAll(string sorting, TypeOfService filtering, string sortingA, int count, int page) — выводит список всех типов доп. услуг, sorting — сортировка по первому параметру, filtering — фильтрация, sorting — сортировка по второму параметру, count — количество строк на странице, page — текущая страница;

+ FindByID( int ID) — поиск типа по ID;

+ Getkolvo(string sorting, TypeOfService filtering, string sortingA, int count, int page) — получение количества типов услуг, удовлетворяющих условиям запроса;

+ GetMaxID() — получение максимального ID.

***Service.*** Класс, отображающий оказанные услуги. Атрибуты: ID, Status(активен или нет), Name(прокат/услуга), TypeId(тип услуги), kolvo(количество услуг), Price, Reader(читатель), BookId(книга), Fine(Штраф), DataSt (дата начала), DataEnd(дата возврата), DataEndFact(дата возврата по факту). Связан с классом TypeOfService, при оформлении заявки нужно указать тип услуги.

+add() — функция добавления услуги в БД;

+save() — функция сохранения изменений;

+Service () — конструктор класса Service без параметров;

+ Service (string ID, string Name, string TypeId, string DateSt, string DateEnd, string DateEndFact, string Price, string Paid, string kolvo, string ReaderID, string Status, string Fine, string BookId) — конструктор класса Service.

***allService.*** Класс, позволяющий работать со списком всех услуг.

+ FindAll(string sorting, Service filtering, string sortingA, int count, int page, DateTime DateStA, DateTime DateStB, DateTime DateEndA, DateTime DateEndB, DateTime DateEndFacA, DateTime DateEndFacB, int priceA, int priceB) — выводит список всех услуг, sorting — сортировка по первому параметру, filtering — фильтрация, sorting — сортировка по второму параметру, count — количество строк на странице, page — текущая страница;

+ Getkolvo(string sorting, Service filtering, string sortingA, int count, int page, DateTime DateStA, DateTime DateStB, DateTime DateEndA, DateTime DateEndB, DateTime DateEndFacA, DateTime DateEndFacB, int priceA, int priceB) — получение количества прокатов, удовлетворяющих условиям запроса;

+ FindByID(int ID) — поиск услуги по ID;

+ GetMaxID() — получение максимального ID.

***Book.*** Класс, отображающий все книги.

Атрибуты: Annotation(аннотация), Autor(автор), DataOfAdd(дата поступления), DataOfPublication(дата публикации), Genre(жанр), ID, Location(размещение), Name, NumberOfPages(кол-во стр.), NumberOfVolumes(кол-во томов), OriginalName(оригинальное имя), Price(цена), Publisher(издательство), Status(в библиотеке или нет), Translator(переводчик), Type(тип книги: журнал, аудио-кига и т.д.). Связан с классами Type, Autor, при создании карточки книги нужно указать автора и тип книги.

+add() — функция добавления книги в БД;

+save() — функция сохранения изменений;

+Book() — конструктор класса Book без параметров;

+ Book (string ID, string Name, string IDAutor, string Annotation, string Add, string Publication, string Jenre, string Location, string KolInstant, string KolPage, string KolVol, string OriginalName, string Price, string PriceOfRental, string Publisher, string Translator, string Type, string Status) — конструктор класса Book.

***allBook.*** Класс, позволяющий работать со списком всех книг.

+ FindAll(string sorting, Book filtering, string sortingA, int count, int page, DateTime YearA, DateTime YearB, int PriceA, int PriceB) — выводит список всех книг, sorting — сортировка по первому параметру, filtering — фильтрация, sortingА — сортировка по второму параметру, count — количество строк на странице, page — текущая страница, YearA — фильтрация по году издания (от), YearB — фильтрация по году издания (до), PriceA — фильтрация по цене издания (от), PriceB — фильтрация по цене издания (до);

+ FindByID(int ID) — поиск книги по ID;

+ Getkolvo(string sorting, Book filtering, string sortingA, int count, int page, DateTime YearA, DateTime YearB, int PriceA, int PriceB) — получение количества книг, удовлетворяющих условиям запроса;

+ GetMaxID() — получение максимального ID.

***Autor.*** Класс, отображающий автора.

Атрибуты: FIO, ID, Nick (псевдоним), Descrip(описание автора), Status(удален или активен)

+add() — функция добавления автора в БД;

+save() — функция сохранения изменений;

+Autor() — конструктор класса Autor без параметров;

+ Autor (string ID, string FIO, string Nick, string Descrip, string Status) — конструктор класса Autor.

***allAutor.*** Класс, позволяющий работать со списком всех авторов.

+ FindAll(string sorting, Autor filtering, string sortingA, int count, int page) — выводит список всех авторов, sorting — сортировка по первому параметру, filtering — фильтрация, sorting — сортировка по второму параметру, count — количество строк на странице, page — текущая страница;

+Getkolvo(string sorting, Autor filtering, string sortingA, int count, int page) — получение количества авторов, удовлетворяющих условиям запроса;

+ FindByID(int ID) — поиск автора по ID;

+ GetMaxID() — получение максимального ID.

***Type.*** Класс, отображающий тип книги.

Атрибуты: Name, ID, Status(удален или активен)

+add() — функция добавления типа в БД;

+save() — функция сохранения изменений;

+ Type() — конструктор класса Type без параметров;

+ Type (string ID, string Name, string Status) — конструктор класса Type.

***allType.*** Класс, позволяющий работать со списком всех типов книг.

+ FindAll(string sorting, TypeB filtering, string sortingA, int count, int page) — выводит список всех типов, sorting — сортировка по первому параметру, filtering — фильтрация, sorting — сортировка по второму параметру, count — количество строк на странице, page — текущая страница;

+Getkolvo(string sorting, TypeB filtering, string sortingA, int count, int page) — получение количества типов книг, удовлетворяющих условиям запроса;

+ FindByID(ID: Int) — поиск типа по ID;

+ GetMaxID() — получение максимального ID.

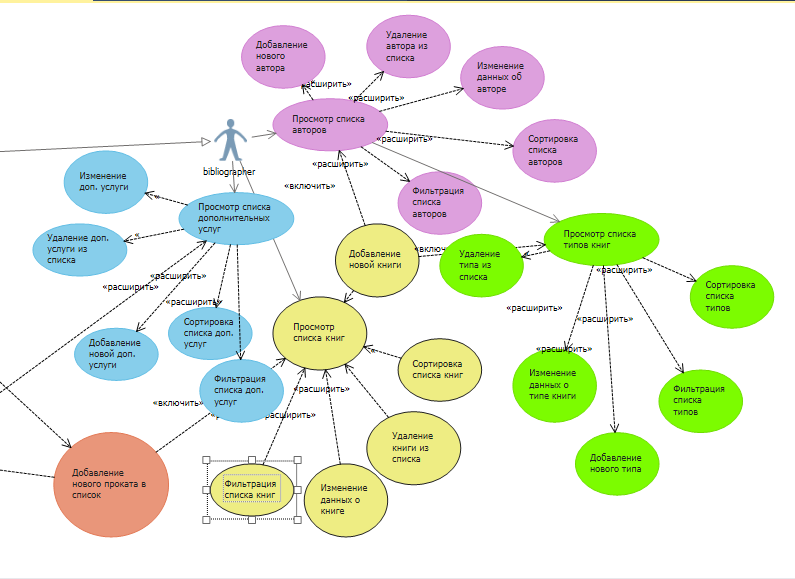


Рисунок 1.Диаграмма прецедентов

## 

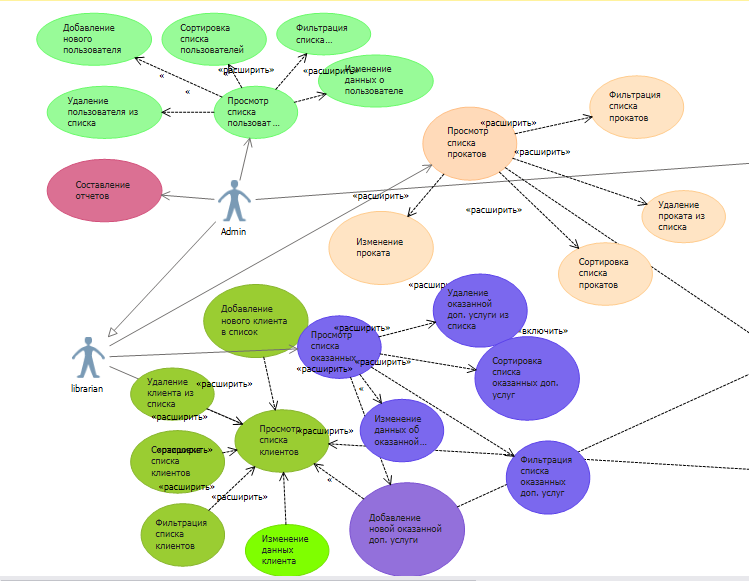


Рисунок 2. Диаграмма прецедентов(2)



Рисунок 3. Диаграмма классов (начало)



Рисунок 4. Диаграмма классов (конец)

## **Диаграмма действий**

Диаграмма действий описывает деятельность библиографа — создание, удаление, изменения проката. После авторизации для библиографа выводится пользовательский интерфейс, где он может выбрать нужный раздел. После выбора раздела выводится список объектов (в зависимости от раздела). Менеджер может создать новый прокат, редактировать или удалить существующий. Подобные действия совершаются над клиентами, услугами (Рисунок 4).

## **Диаграмма состояний**

Диаграмма описывает все возможные состояния проката на продолжении ее жизненного цикла. Сначала прокат на заполнении, потом активен. Возможно продление проката. После сдачи книг прокат закрывается. Последнее состояние проката «удален» (Рисунок 7).

## **Диаграмма последовательностей**

Диаграмма описывает процесс добавления проката. Пользователь нажимает на главной форме кнопку “Новый прокат”, открывается форма добавления нового проката. Пользователь выбирает читателя: открывается форма поиска читателя, из класса «allReaders» возвращается читатель, соответствующий критериям поиска. Выбор книги: открывается форма поиска книги, из класса «allBooks» возвращается книга, соответствующая критериям поиска. После заполнения формы и нажатия на кнопку «Сохранить», классу Rental присваиваются атрибуты, функция GetMaxID() возвращает максимальный идентификатор, который присваивается новому прокату (Рисунок 8).



Рисунок 5. Диаграмма действий (Создание проката)



Рисунок 6. Диаграмма действий (удаление проката)



Рисунок 7. Диаграмма действий (удаление проката)



Рисунок 8. Диаграмма состояний

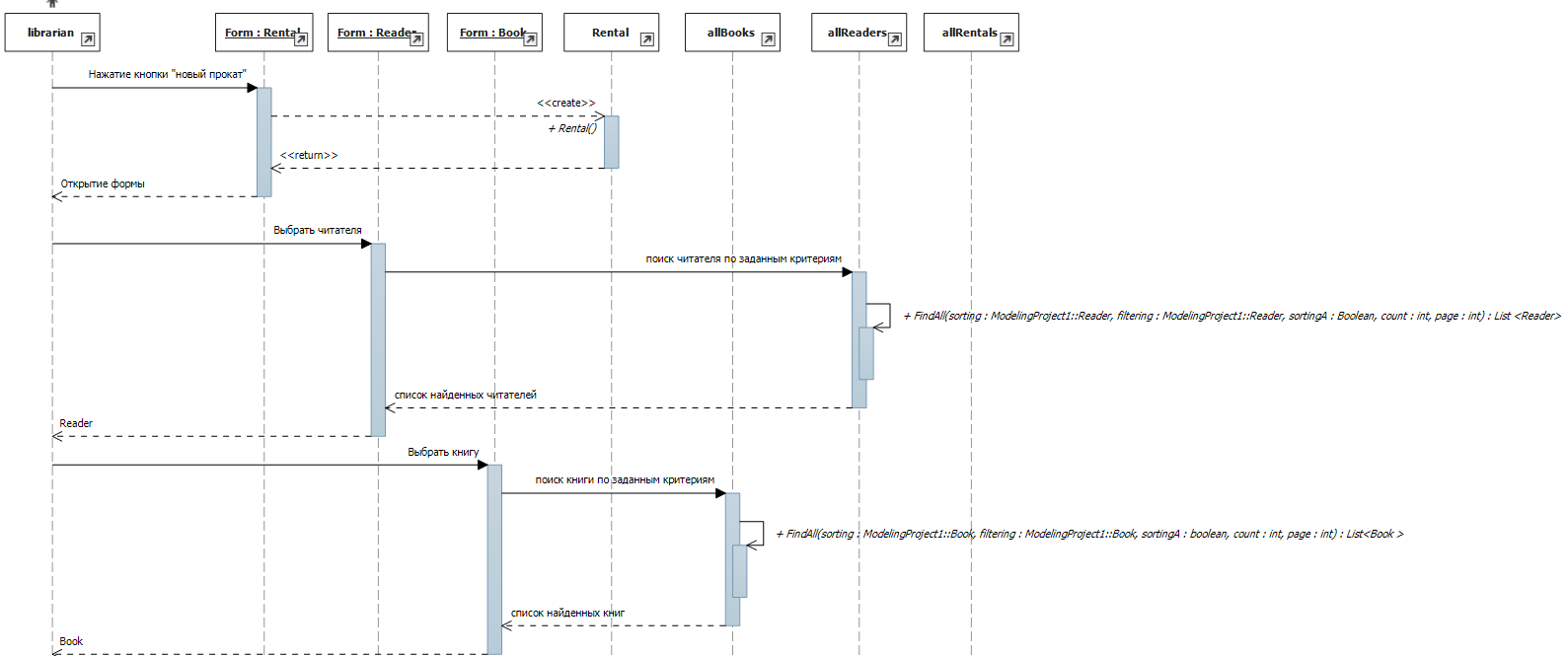


Рисунок 9. Диаграмма последовательностей (начало)

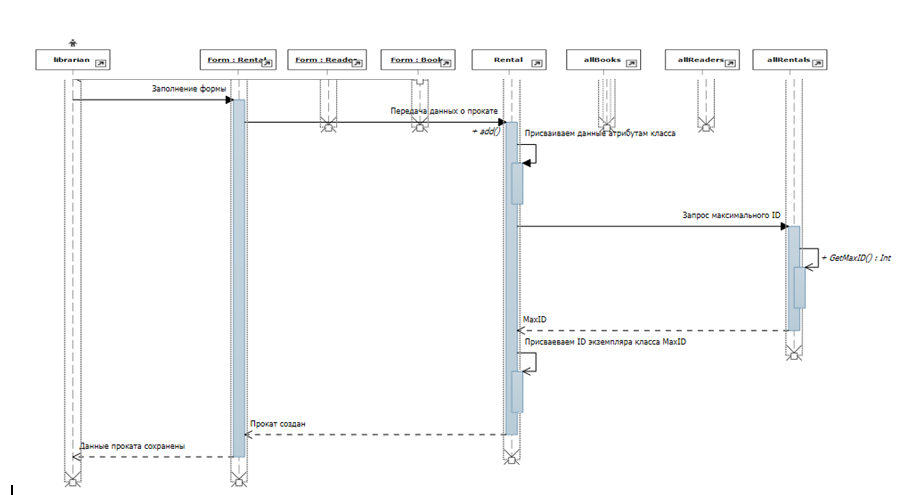


Рисунок 10. Диаграмма последовательностей (конец)

## **Диаграмма размещения**

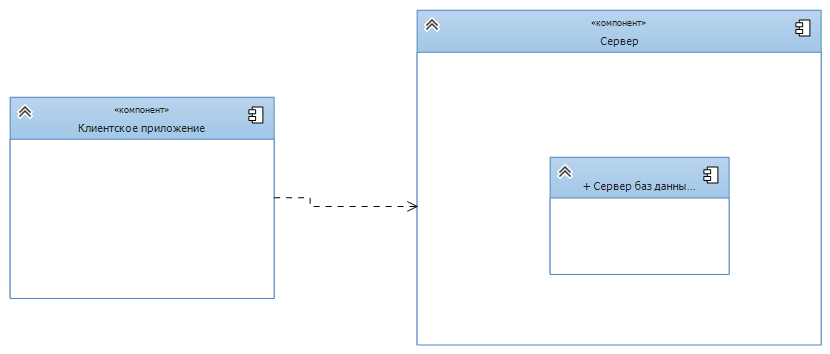
Диаграмма размещения отражает физические взаимосвязи между программными и аппаратными компонентами системы. На диаграмме размещения изображены узлы выполнения программных компонентов, а также объектов. Показано, что клиентское приложение, установленное на компьютере пользователя, взаимодействует с сервером, который содержит в себе базу данных.

Рисунок 11. Диаграмма размещения

## **Диаграмма компонентов**

На данной диаграмме изображены все компоненты: клиентское приложение, книги, читатели, прокаты, пользователи, авторы, типы книг, виды услуг. Эти компоненты взаимодействуют друг с другом с помощью интерфейсов (Рисунок 11).

## **Диаграммы интерфейсов**

***IReader***

+ AddReader (Reader: Reader) — функция, добавляющая читателя в базу данных. Параметр «Reader» — читатель, которого необходимо добавить в БД;

+ SaveReader (Reader: Reader) — функция, редактирующая данные о читателе. Параметр «Reader» — читатель, которого необходимо редактировать в БД;

+ FindByIDReader(ID: Int): Reader — функция, осуществляющая поиск читателя в базе данных по ID и возвращающая найденный, если такой есть.

+ GetMaxIDReader(): Int — функция, возвращающая ID последнего читателя в базе данных.

+ GetListReaders(sorting: string, sortingA: string, filtering: Reader, count: Int, page: Int) List <Reader> — функция, возвращающая список читателей с заданными параметрами. Параметры:

* sortintg: string — отвечает, по какому полю будет сортироваться список;
* sortingA: string — отвечает, по возрастанию или убыванию будут сортироваться элементы;
* filtering: Reader — отвечает за фильтрацию;
* count: int — отвечает, сколько элементов необходимо показать;
* page: int — отвечает, с какой страницы начинать поиск элементов.

+ GetkolvoReaders(sorting: string, sortingA: string, filtering: Reader, count: Int, page: Int): Int — функция, возвращающая общее количество записей, удовлетворяющих заданным параметрам.

***IService***

+ AddService(Service: Service) — функция, добавляющая прокат в базу данных. Параметр «Service» — прокат, который необходимо добавить в БД;

+ SaveService(Service: Service) — функция, редактирующая данные проката. Параметр «Service» — прокат, который необходимо редактировать в БД;

+ FindByIDService (ID: Int): Service — функция, осуществляющая поиск проката в базе данных по ID и возвращающая найденный, если такой есть.

+ GetMaxIDService (): Int — функция, возвращающая ID последнего проката в базе данных.

+ GetListService (sorting: string, sortingA: string, filtering: Service, count: Int, page: Int) List <Service> — функция, возвращающая список прокатов с заданными параметрами. Параметры:

* sortintg: string — отвечает, по какому полю будет сортироваться список;
* sortingA: string — отвечает, по возрастанию или убыванию будут сортироваться элементы;
* filtering: Service — отвечает за фильтрацию;
* count: int — отвечает, сколько элементов необходимо показать;
* page: int — отвечает, с какой страницы начинать поиск элементов.

+ GetkolvoService (sorting: string, sortingA: string, filtering: Service, count: Int, page: Int): Int — функция, возвращающая общее количество записей, удовлетворяющих заданным параметрам.

Остальные интерфейсы аналогичны интерфейсу «IReader» (Рисунок 12).



Рисунок 12. Диаграмма компонентов

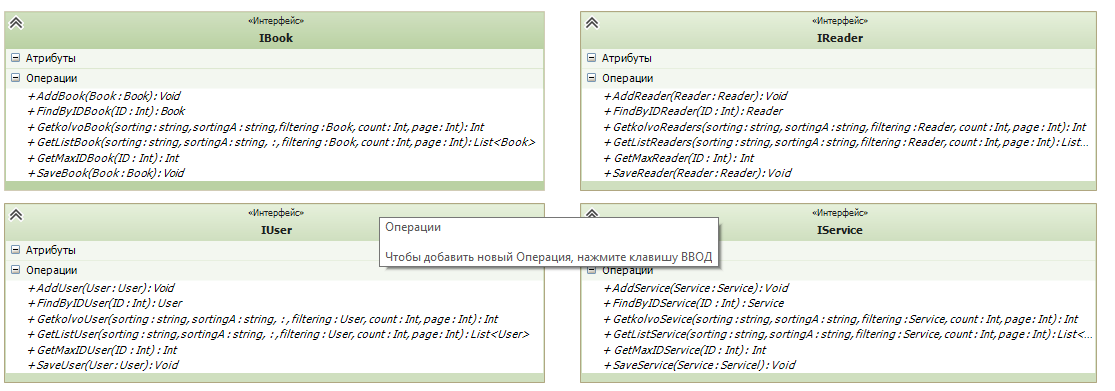


Рисунок 13. Интерфейсы

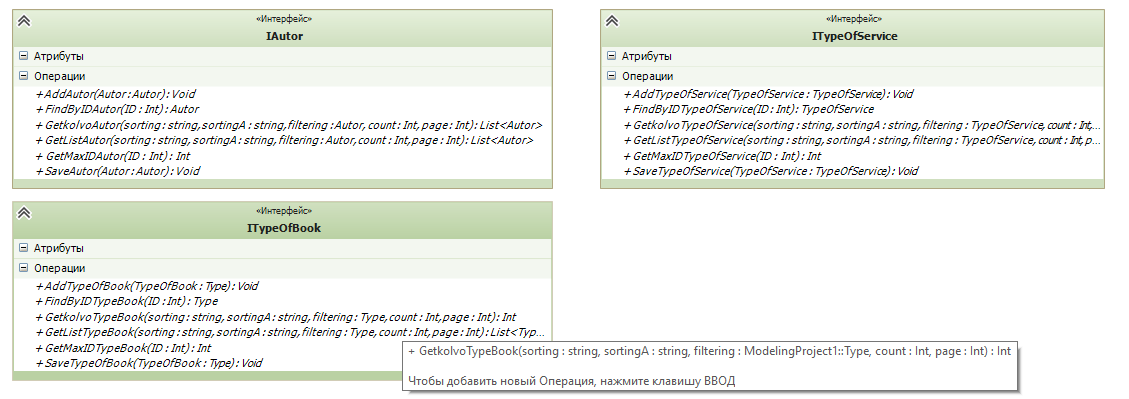


Рисунок 14. Интерфейсы (2)

# Пользовательский интерфейс

Соответствие основным принципам разработки пользовательского интерфейса:

1) Принцип структуризации. Пользовательский интерфейс структурирован так, чтобы близкие по смыслу, родственные его части были связаны видимым образом, а независимые – разделены;

2) Принцип простоты. Наиболее распространённые действия выполняются максимально просто, чтобы существенно снизить затрачиваемые усилия;

3) Принцип наблюдаемости. Все операции и данные, необходимые для решения определённой задачи, хорошо видны, когда пользователь пытается её решить;

4) Принцип обратной связи. Пользователь получает сообщения о действиях программы и о важных событиях внутри неё.

5) Принцип толерантности. Интерфейс предусматривает возможные ошибки пользователя.

*Интерфейс для веб-приложения*

Интерфейс для веб-приложения отличается удобством использования в веб-браузерах. На каждой странице расположены текстовые поля, набор кнопок, различные таблицы (см. приложение 3).

*Интерфейс для настольного приложения*

Интерфейс для настольного приложения отличается удобством использования на различных персональных компьютерах. На каждой форме расположены текстовые поля, набор кнопок и различные таблицы. Ведется активный диалог с пользователем (приложение 3).

Тщательно продуманный, лёгкий в изучении и понимании, учитывающий человеческие возможности и ограничения интерфейс гарантирует удобство работы с программой и, в конечном итоге, её коммерческий успех.

# Тестирование программы

Войдем в систему от имени администратора.

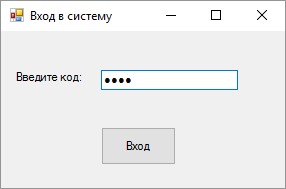


Рисунок 15. Вход в систему

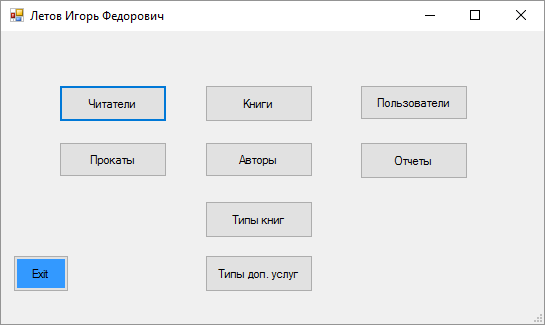


Рисунок 16. Главная форма

Создадим нового читателя, заполняя все существующие поля некоторыми значениями:

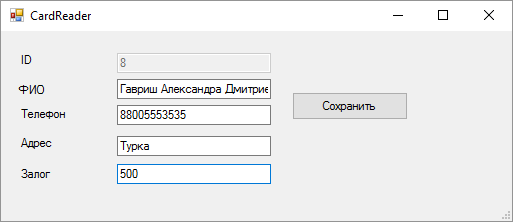


Рисунок 17. Новый читатель

При нажатии кнопки «Сохранить» запись о новом читателе добавляется в базу данных, таблица обновляется, и добавленная заявка отображается на экране.

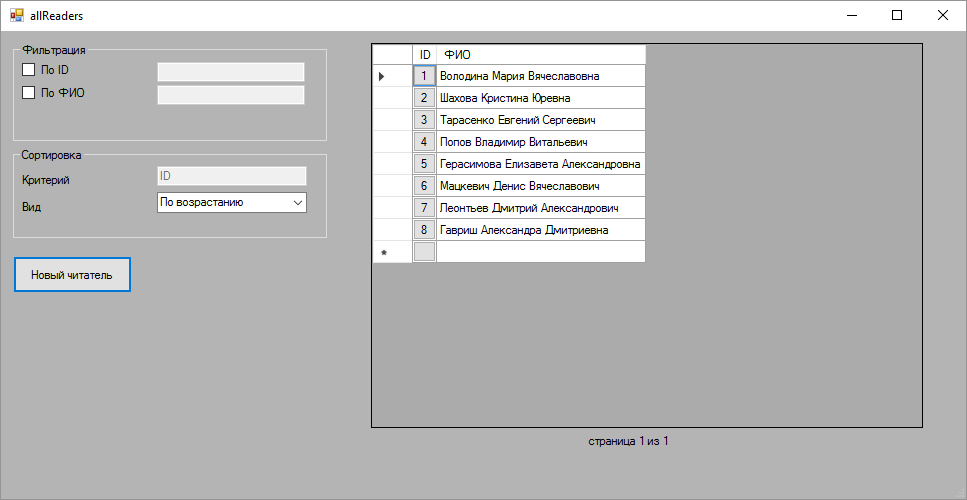


Рисунок 18. Все читатели

При нажатии на поле «ID» открывается форма, заполненная характеристиками выбранного читателя, пользователем вносятся необходимые коррективы. После нажатия на кнопку «Сохранить» изменения сохраняются, и таблица обновляется.

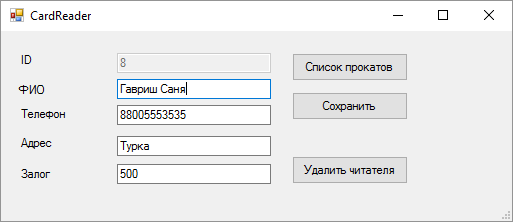


Рисунок 19. Редактирование карточки читателя

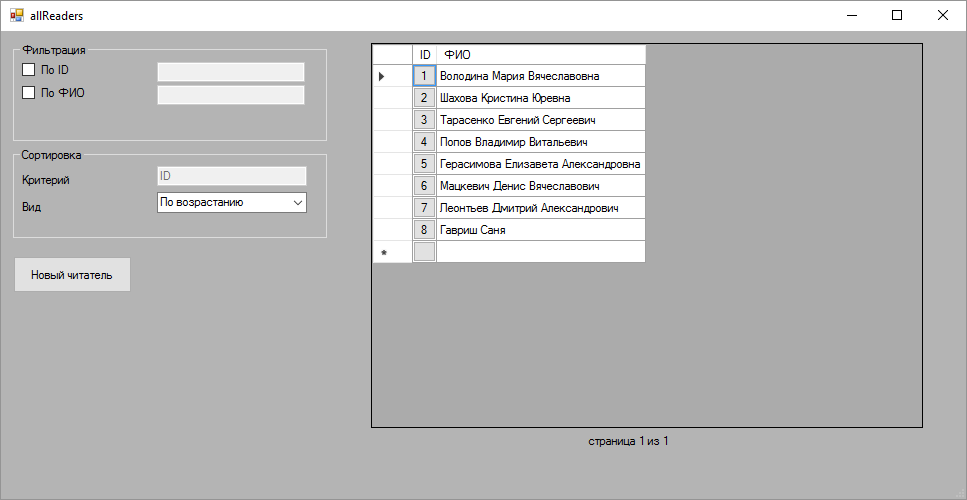


Рисунок 20. Редактирование читателя

Создадим новый прокат, заполняя все существующие поля некоторыми значениями:

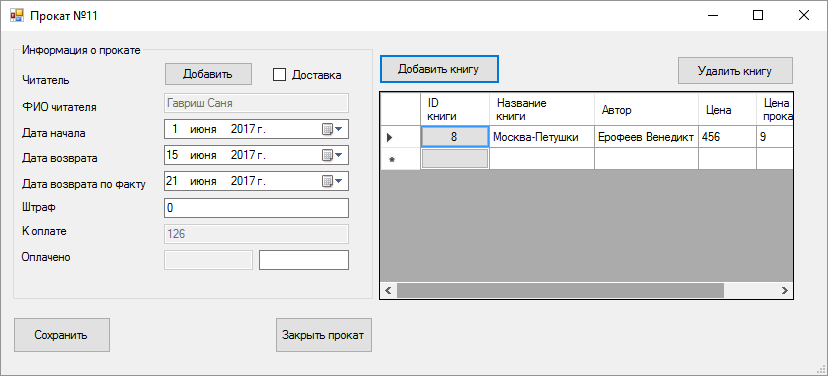


Рисунок 21. Создание проката

При нажатии на кнопку «Добавить книгу» открывается форма поиска книг. Нажав на кнопку «Выбрать», книга добавляется в прокат.

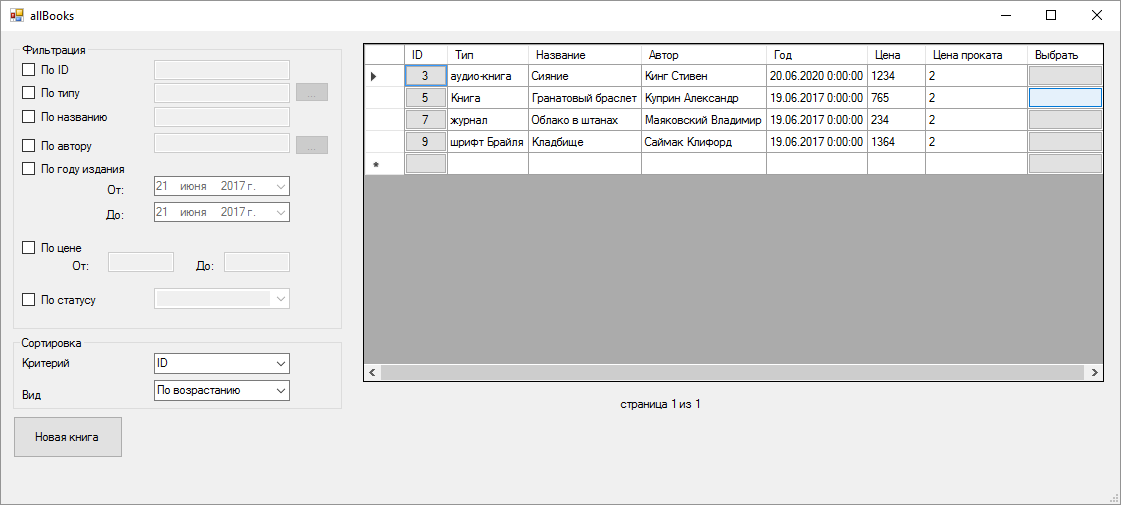


Рисунок 22. Все книги

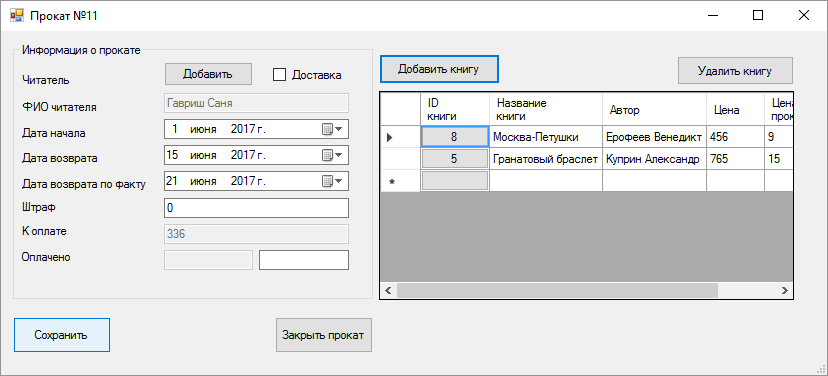


Рисунок 23. Создание проката(2)

При нажатии на копку «Сохранить» прокат добавляется в базу.

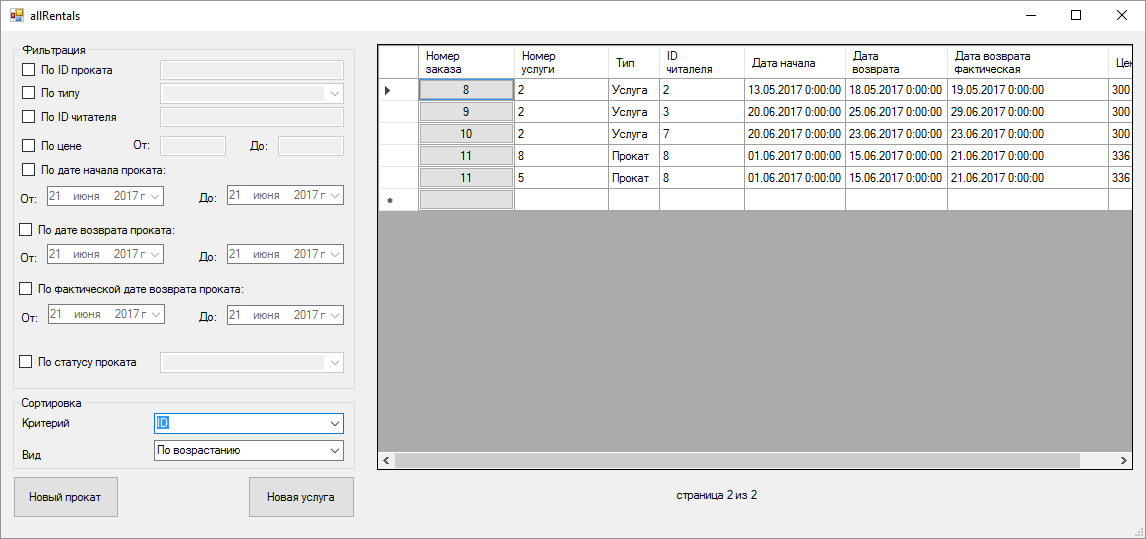
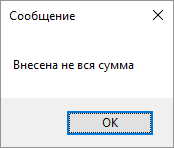


Рисунок 24. Список прокатов

При нажатии на номер заказа открывается форма проката. При нажатии на кнопку «Закрыть прокат», выполняется проверка оплаты проката.



Зайдя в программу от имени библиографа, создадим вид услуги, заполнив все необходимые поля.

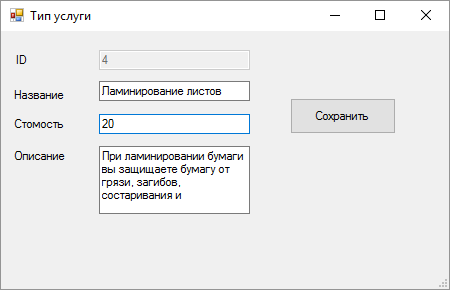


Рисунок 25. Новая услуга

После нажатия кнопки «Сохранить» вид услуги появляется в списке. Список всех услуг можно сортировать и фильтровать по выбранным параметрам.

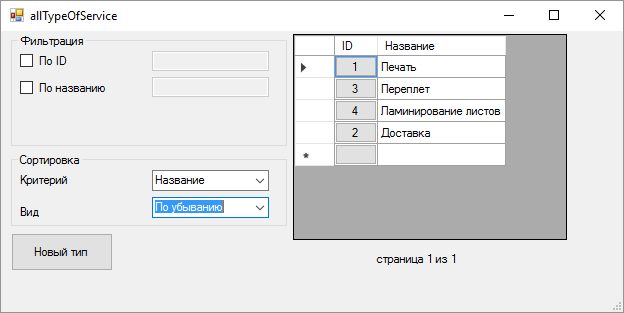


Рисунок 26. Сортировка

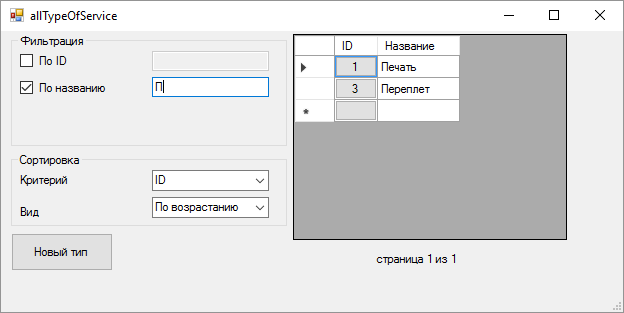


Рисунок 27. Фильтрация

Для создания новой услуги, заполним все необходимые поля.

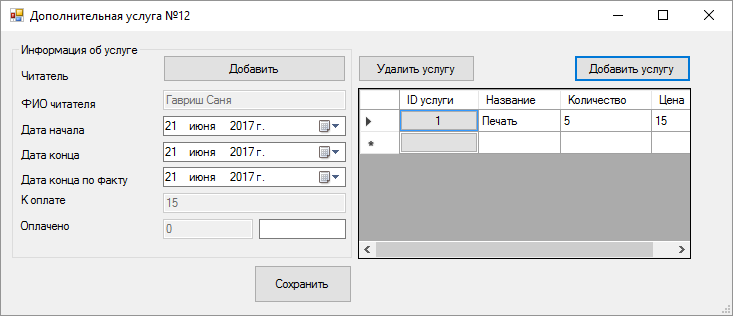


Рисунок 28. Новая услуга

При нажатии кнопки «Добавить услугу», открывается список всех видов услуг, выбрав нужную, вводим количество. При нажатии кнопки «Выбрать» услуга появляется в списке.

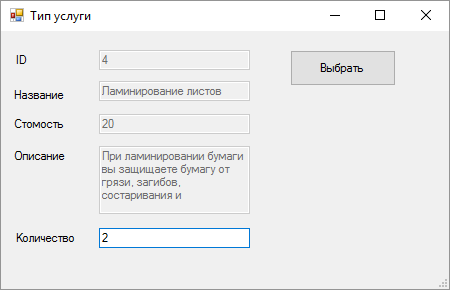


Рисунок 29. Добавление услуги

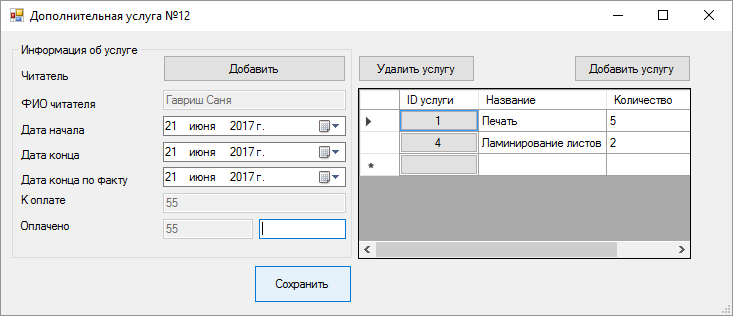


Рисунок 30. Новая услуга (2)

При нажатии кнопки «Сохранить» услуга добавляется в список всех прокатов.

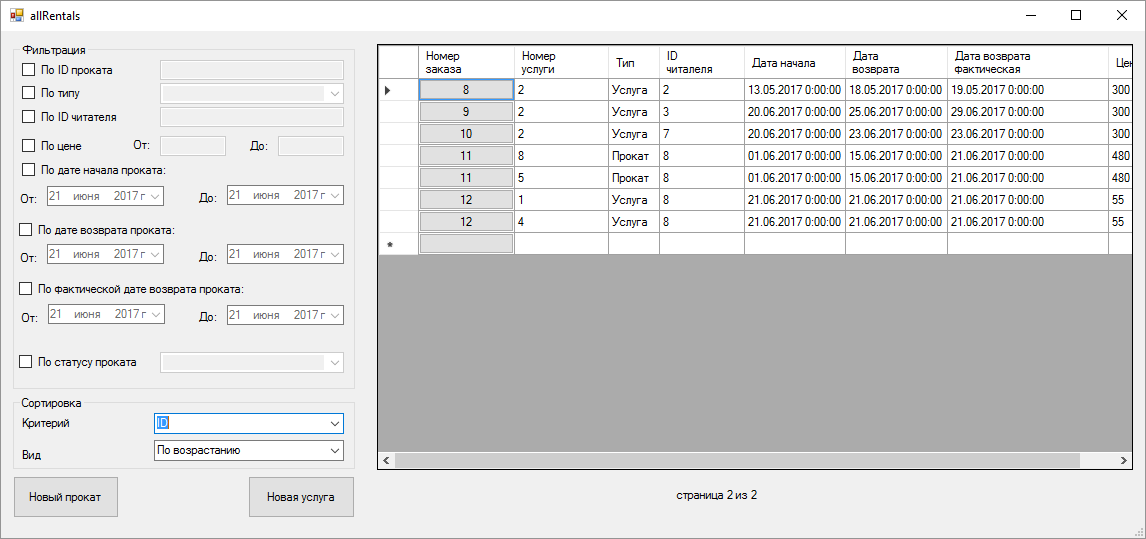


Рисунок 31. Список услуг и прокатов

На главной форме нажатие на кнопку «Отчет» вызывает окно отчетов о прибыли за указанный период времени.

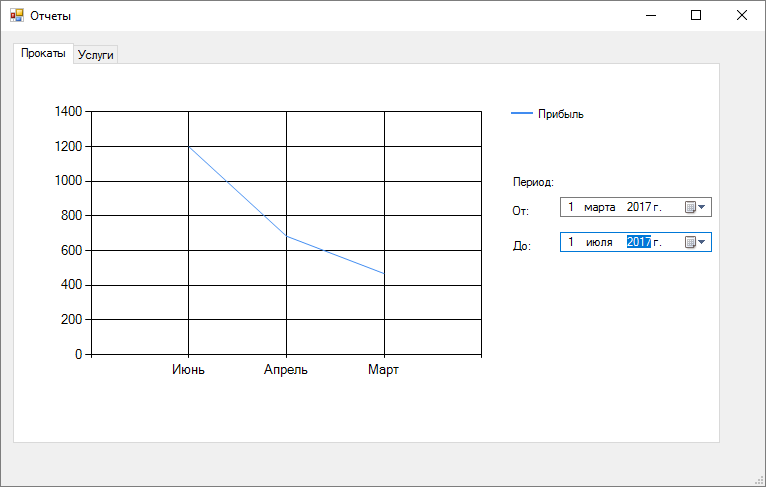


Рисунок 32. Отчет

Зайдем в таблицу всех читателей, откроем карточку читателя. При нажатии кнопки «Список всех прокатов» открывается история всех обращений читателя.

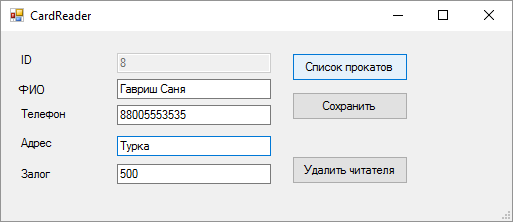


Рисунок 33. Карточка читателя (2)

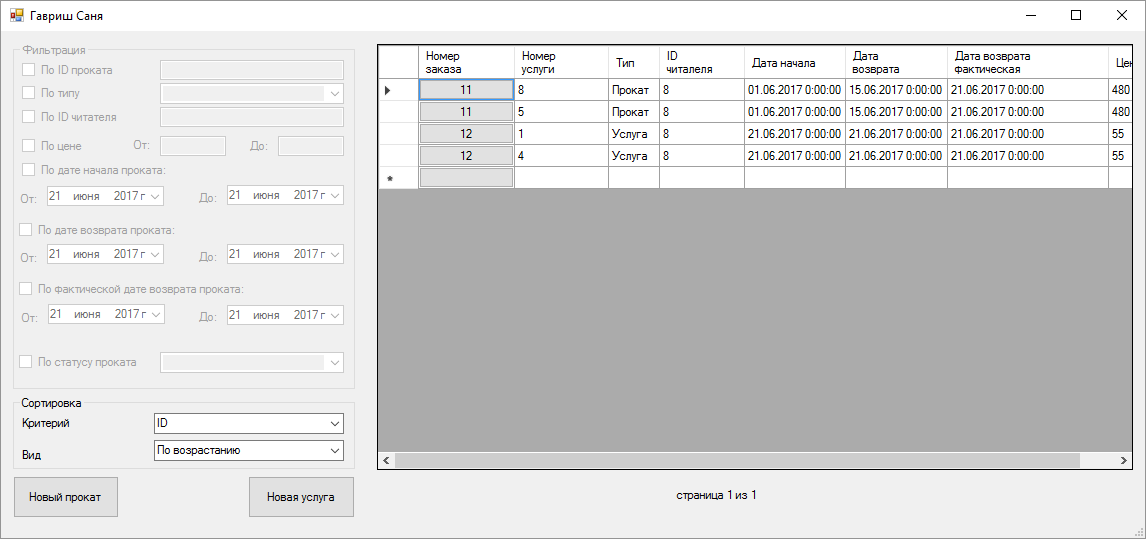
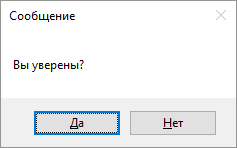


Рисунок 34. История всех обращений читателя

В карточке проката, при нажатии кнопки «Удалить», система просит подтверждения. При согласии прокат удаляется из списка.



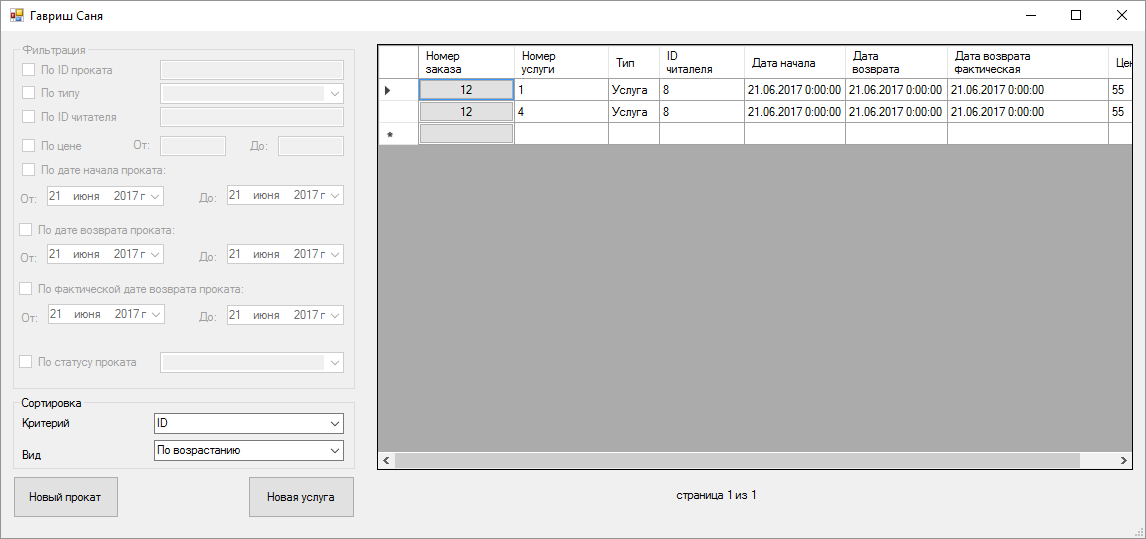


Рисунок 35. Удаление

# Заключение

В процессе выполнения курсовой работы изучены взаимосвязи в информационной структуре библиотеки, проанализированы различные запросы к базе данных определенных таблиц. На основе полученных данных было разработано клиентское приложение.

В теоретической части курсовой работы рассмотрены основы разработки автоматизированной информационной системы: процесс создания, проектирования и использования базы данных MySQL.

В практической части реализована структура базы данных библиотеки и описан процесс создания клиентского приложения с использованием программы Visual Studio.

После тестирования программы можно сделать вывод о том, что созданная в ходе курсового проекта информационная система управления фондами библиотеки для ООО «Букинист» удовлетворяет всем изложенным требованиям и может использоваться по назначению. Использование данной программы позволяет автоматизировать большой объем работы и вести необходимый учет. Программа является удобной для пользователей с различным уровнем знаний компьютера и позволяет за короткие сроки овладеть необходимыми навыками для работы с ней. Также данная программа является достаточно гибкой для расширения функционала и усовершенствований. В дальнейшей разработке можно ускорить работу самой системы и дать возможность пользователю настраивать интерфейс исходя из своих предпочтений.

# Список использованных источников

1. ГОСТ 34.602-89. Техническое задание на создание автоматизированной системы.  — Москва: Изд-во стандартов, 1989. — 14с.
2. Катаев, М.Ю. Объектно-ориентированное программирование: учебное пособие / М.Ю. Катаев.  — Томск: Изд-во томский межвузовский центр дистанционного образования, 2000.  — 145 с.
3. Гамма, Э. Приемы объектно-ориентированного проектирования. Паттерны проектирования / Э. Гамма — СПб.: Питер, 2010.  — 366 с.
4. Ларман, К. Применение UML и шаблонов проектирования. / К. Ларман  — Москва: Изд-во Вильямс, 2004.  — 624 с.
5. Основы программирования на C# [электронный ресурс] / режим доступа: http://www.intuit.ru/studies/courses/2247/18/info
6. Методы и средства инженерии программного обеспечения [электронный ресурс] / режим доступа: http://www.intuit.ru/studies/courses/2190/237/info
7. Нотация и семантика языка UML [электронный ресурс] / режим доступа: http://www.intuit.ru/studies/courses/32/32/info
8. Объектно-ориентированное программирование и программная инженерия [электронный ресурс] / режим доступа: http://www.intuit.ru/studies/courses/2309/609/info
9. Основы офисного программирования и документы Word [электронный ресурс] / режим доступа: http://www.intuit.ru/studies/courses/2251/113/info
10. Основы программирования на C# [электронный ресурс] / режим доступа: http://www.intuit.ru/studies/courses/2247/18/info
11. Основы тестирования программного обеспечения [электронный ресурс] / режим доступа: http://www.intuit.ru/studies/courses/48/48/info

# Приложения

## **Приложение 1. Техническое задание**

УТВЕРЖДАЮ УТВЕРЖДАЮ

Директор ООО «Букинист» Директор ЗАО «Дантес»

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_А.С. Пушкин \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_И.И.Бессмертный

«\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_2017г. «\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2017г.

Наименование вида АС

*Автоматизированная система “Управление фондом библиотеки"*

Сокращённое наименование

***АС «УФБ»***

***Техническое задание***

На 15 листах

Действует с «4» Апреля 2017г.

СОГЛАСОВАНО

Директор ООО «Букинист»

Александр Сергеевич Пушкин

\_\_\_\_\_\_\_\_\_\_\_\_\_\_А.С. Пушкин

Иркутск

2017

1. **Общие положения**
   1. **Полное наименование системы и ее условное обозначение**
      1. **Полное наименование:** Автоматизированная система “Управление фондом библиотеки".
   2. **Краткое наименование:** АС «УФБ».
   3. **Номер договора (контракта)**

Договор №1234 от 04.04.2017 на разработку ПО

* 1. **Наименование организаций Заказчика и Разработчика:**
     1. Заказчик: ООО «Букинист». Адрес фактический: г. Иркутск, ул. Бесы, 14. Телефон: +7(3952) 111111
     2. Разработчик: ЗАО «Дантес». Адрес фактический: г. Иркутск, ул. Бесмертная. Телефон: +7 (3952) 3333333
  2. **Плановые сроки начала и окончания работы:**
     1. Дата начала: 20.04.2017
     2. Дата окончания: 19.05.2017
  3. **Источники и порядок финансирования работ:**

Финансирование проекта осуществляется за счёт средств Заказчика. Порядок финансирования работ определяется условиями Договора № 1234 от 04.04.2017

* 1. **Порядок оформления и предъявления заказчику результатов работ по созданию системы:**

Работы по созданию АС “УФБ” сдаются Разработчиком поэтапно в соответствии с календарным планом Проекта. По окончании каждого из этапов работ Разработчик сдает Заказчику соответствующие отчетные документы этапа, состав которых определены Договором.

1. **Назначение и цели создания системы**

## **Назначение системы** Основной вид деятельности, для автоматизации которого предназначена ПО, согласно коду ОКВЭД 91.01- Деятельность библиотек и архивов

## **Цели создания системы АС «УФБ»**

* Улучшение качества деятельности библиотеки ООО «Букинист».
* Повышение эффективности работы сотрудников библиотеки ООО «Букинист».

# Характеристика объекта автоматизации

Таблица . Характеристика объекта автоматизации

|  |  |  |  |
| --- | --- | --- | --- |
| **Структурное подразделение** | **Наименование процесса** | **Возможность автоматизации** | **Решение об автоматизации в ходе проекта** |
| Отдел организационно-финансовой деятельности | Финансовые операции. Составление соответствующей документации | Автоматизация возможна | Будет автоматизирован |
| Отдел обслуживания | Организация библиотечного и информационного обслуживания пользователей; | Автоматизация возможна | Будет автоматизирован |
| Отдел обработки и комплектования книжных фондов | Управление книжным фондом | Автоматизация возможна | Будет автоматизирован |

Библиотека ООО «Букинист» представляет собой двухэтажное здание с двумя читальными залами, компьютерным классом, мультимедийной аудиторией и 5-ю архивами: художественной, научной, отраслевой, детской литературы, отделом искусств. Кроме проката книг могут предоставляться дополнительные услуги:

Таблица . Прейскурант цен доп. услуг

|  |  |
| --- | --- |
| Ксерокопия документов | 4 руб. 1 лист |
| Аренда мультимедийной аудитории | 1 час — 1000 руб. |
| Пользование компьютерным классом | 1 час — 100 руб. |
| Доставка книг (только по городу) | 300 руб. |

Доставка книг осуществляется только по городу, цена не зависит от расстояния. Арендовать мультимедийную аудиторию можно только в рабочее время библиотеки. Пользования компьютерным классом возможно только в рабочее время библиотеки, информация об аренде компьютера фиксируется по окончанию сеанса.

Часть книг сдается в прокат платно, цена проката определяется процентом от первоначальной стоимости книги.

Библиотека работает каждый день с 11 до 19.

Численность персонала библиотеки 9 человек: 5 библиотекарей, 2 библиографа, 1 охранник, 1 курьер.

|  |  |
| --- | --- |
| **Должность** | **Обязанности** |
| Библиотекарь(5) | Информирование читателей, организация и контроль сдачи/возврата книг, предоставление дополнительных услуг |
| Библиограф(2) | Управление книжным фондом |
| Охранник(1) | Предотвращение хищений на охраняемом объекте, а также обеспечение безопасности и общественного порядка на его территории |
| Курьер(1) | Доставка книг по городу |

# Требования к системе

* 1. **Требования к системе в целом**
     1. **Требования к режимам функционирования АС «УФБ»**

Автоматизированная система должна поддерживать следующие режимы функционирования:

* Основной режим, в котором подсистемы выполняют все свои основные функции;
* Профилактический режим, в котором одна или все подсистемы не выполняют своих функций.
* В основном режиме функционирования автоматизированная система должна обеспечивать:
* Работу пользователей в режиме – 8 часов в день, 7 дней в неделю;
* Выполнение своих функций – добавление, обработка и сохранение данных, хранение данных, предоставление отчетности.
* В профилактическом режиме автоматизированная система «УФБ» должна обеспечивать:
* Техническое обслуживание;
* Устранение ошибок.
  + 1. **Требования к структуре и функционированию системы**

Система должна иметь структуру, обеспечивающую реализацию основных функций в соответствии требованиям данного технического задания и состоять из следующего состава функциональных подсистем:

* Учет читателей;
* Учет книжного фонда;
* Анализ и отчётность.

Подсистема учета читателей должна обеспечивать учет и ведение данных по читателям и условиям их обслуживания.

Подсистема учета книжного фонда должна обеспечивать учет и ведение данных по книгам.

Подсистема анализа и отчетности должна обеспечивать осуществление выбора определенного набора данных по объектам Системы, формирования и визуализации отчетности по заданным параметрам.

* + 1. **Требования к численности и квалификации персонала системы**

#### Требования к численности персонала

АС «УФБ» не требует особых навыков для управления. Для работы с данными требуется 3 рабочих места: Библиотекарь, Библиограф, Системный администратор.

#### Требования к квалификации работы персонала

Весь персонал должен иметь базовые навыки работы с ПК, пройти инструктаж по работе с ПО.

* + 1. **Требования к надежности**

#### Требования к мероприятиям по обеспечению надежности

Надежность Системы должна достигаться комплексом организационных и технических мер.

Организационные меры по обеспечению надежности должны быть направлены на минимизацию ошибок пользователей и обслуживающего персонала, а также минимизацию времени восстановления работоспособности системы за счёт:

* обеспечения требуемого уровня квалификации обслуживающего персонала;
* регламентации, нормативного обеспечения и контроля выполнения работ обслуживающего персонала;
* своевременного оповещения пользователей о случаях нештатной работы компонентов системы;
* обеспечения и контроля работ по сервисному обслуживанию и поддержке компонентов комплекса технических средств, или поддержки собственными специалистами.

Под аварийной ситуацией понимается аварийное завершение процесса, выполняемого той или иной компонентой Системы, а также «зависание» этого процесса.

#### Состав показателей надежности для системы в целом

Уровень надежности должен достигаться согласованным применением организационных, организационно-технических мероприятий и программно-аппаратных средств.

Надежность должна обеспечиваться за счет:

* применения технических средств, системного и базового программного обеспечения, соответствующих классу решаемых задач;
* своевременного выполнения процессов администрирования Системы;
* регулярное выполнение требований ГОСТ 51188-98. «Защита информации. Испытания программных средств на наличие компьютерных вирусов», а именно порядка и методов, проведения испытаний программных средства наличие компьютерных вирусов, эксплуатация в соответствии с руководством пользователя;
* соблюдения правил эксплуатации и технического обслуживания программно-аппаратных средств;
* предварительного обучения пользователей и обслуживающего персонала.

#### Требования к надежности программного обеспечения

Система должна обеспечивать возможность резервирования данных на внешние носители данных, а также возможность восстановления данных с внешнего носителя в течение установленного временного интервала.

Система должна обеспечивать защиту от ошибочных действий пользователей, приводящих к аварийному состоянию объекта или системы и ввода некорректных данных.  
Система должна соответствовать следующим параметрам:

* вероятность безотказной работы[[1]](#footnote-1)-0,95
* вероятность восстановления[[2]](#footnote-2)-0,97
  + 1. **Требования к эргономике и технической эстетике**

ПО должно обеспечивать удобный интерфейс для пользователей, отвечающий следующим требованиям:

* единый стиль оформления для пользовательских интерфейсов;
* должна быть удобная и понятная навигация;
* взаимодействие пользователя с системой должно осуществляться на русском языке;
  + 1. **Требования к эксплуатации, техническому обслуживанию, ремонту и хранению компонентов системы**

Программа должна функционировать в бесперебойном режиме, техническое обслуживание производит системный администратор, при возникновении проблем в программе.

* + 1. **Требования к защите информации от несанкционированного доступа**

Доступ к программе должен быть только у сотрудников компании. При запуске программы будет необходимо ввести код доступа. У каждого сотрудника разный уровень доступа к программе. Таким образом, библиотекарь может только работать с клиентами, библиограф может добавлять и удалять книги в каталог, системный администратор имеет полный доступ к программе.

* + 1. **Требования к патентной частоте**

Программа должна использоваться в компании ООО «Букинист», для которой она была создана и функционал программы не должен затрагивать интересы чужих патентов.

## **Требования к функциям (задачам), выполняемым системой**

1. *Табличное представление данных*

Выходные данные программы должны быть организованы в виде таблиц.

1. *Ведение карточки читателя*

Система должна иметь возможность изменения и просмотра данных о клиенте, а также сохранять информацию о всех выданных клиенту книгах с датами выдачи и возврата.

1. *Ведение справочника читателей*

Система должна иметь возможность добавления и удаления читателей из системы.

1. *Ведение каталога книг*

Система должна иметь возможность добавления, изменения и удаления книг из системы. Составления списка авторов, издательств.

1. *Ведение справочника пользователей системы*

Система должна иметь возможность просмотра, добавления, изменения и удаления пользователя из системы.

1. *Ведение справочника дополнительных услуг*Система должна иметь возможность просмотра, добавления, изменения и удаления услуги из системы.
2. *Ведение списка всех прокатов/возвратов*

Система должна иметь возможность просмотра, добавления, изменения и удаления проката из списка.

1. *Ведение списка всех оказанных дополнительных услуг*

Система должна иметь возможность просмотра, добавления, изменения и удаления услуги из списка.

1. *Составление отчетов:*

Автоматическое построение отчетности в виде графиков  
Программа должна автоматически строить графики в созданном разделе: “Отчет”. Графики должны быть:  
Графический отчет прибыли от проката книг за указанный период;   
График прибыли от дополнительных услуг за указанный период.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Функция** | **Форма представление выходной информации** | **Характеристика точности и времени выполнения** | **Критерии отказа** | **Время восстановления** |
| Составление\фильтрация\сортировка списка читателей | Таблица | не более 60 секунд | Отсутствие данных | Пока не будет добавлена хотя бы одна запись |
| Добавление нового читателя | Добавление читателя в конец таблицы клиентов | не более 30 секунд | Неверный формат введенных данных | Пока не будут добавлены корректные данные |
| Удаление читателя | Сообщение об удалении читателя из БД | не более 30 секунд после подтверждения | Удаление несуществующего читателя | Пока не будет выбран существующий читатель |
| Изменение данных о читателе | Изменение данных в таблице читателей | не более 30 после сохранения | Неверный формат введенных данных | Пока не будут добавлены корректные данные |
| Составление\фильтрация\сортировка списка книг | Таблица | не более 60 секунд | Отсутствие данных | Пока не будет добавлена хотя бы одна запись |
| Составление\фильтрация\сортировка списка авторов | Таблица | не более 60 секунд | Отсутствие данных | Пока не будет добавлена хотя бы одна запись |
| Составление\фильтрация\сортировка списка типов книг | Таблица | не более 60 секунд | Отсутствие данных | Пока не будет добавлена хотя бы одна запись |
| Добавление новой книги | Добавление книги в конец списка книг | не более 30 секунд | Неверный формат введенных данных | Пока не будут добавлены корректные данные |
| Изменение данных о книге | Изменение данных в списке книг | не более 30 после сохранения | Неверный формат введенных данных | Пока не будут добавлены корректные данные |
| Удаление книги | Сообщение об удалении | не более 30 секунд после подтверждения | Удаление несуществующей книги | Пока не будет выбрана существующая книга |
| Составление\фильтрация\сортировка списка пользователей | Таблица | не более 60 секунд | Отсутствие данных | Пока не будет добавлена хотя бы одна запись |
| Добавление нового пользователя | Добавление пользователя в конец списка книг | не более 30 секунд | Неверный формат введенных данных | Пока не будут добавлены корректные данные |
| Удаление пользователя | Сообщение об удалении | не более 30 секунд после подтверждения | Удаление несуществующего пользователя | Пока не будет выбран существующий пользователя |
| Изменение данных о пользователе | Изменение данных в таблице пользователей | не более 30 после сохранения | Неверный формат введенных данных | Пока не будут добавлены корректные данные |
| Новый прокат/ Возврат книги | Изменение данных в списке прокатов/возвратов | не более 30 после сохранения | Наличие просроченных книг/добавление книги, которой нет в наличии | Пока не будет возвращена книга, оплачен штраф/ выбрана книга в наличии |
| Удаление проката/возврата книги | Изменение данных в списке прокатов/возвратов | не более 30 после сохранения | Удаление несуществующего проката/возврата | Пока не будет выбран существующий прокат/возврат |
| Добавление/изменение дополнительной услуги | Изменение данных в списке всех оказанных дополнительных услуг | не более 30 после сохранения | Добавление несуществующей услуги /некорректные данные | Пока не будет выбрана существующая услуга |
| Удаление дополнительной услуги | Изменение данных в списке всех оказанных дополнительных услуг | не более 30 после сохранения | Удаление несуществующей услуги | Пока не будет выбрана существующая услуга |
| Составление отчета | Изменение графика | не более 30 секунд | Отсутствие данных | Пока не будут добавлены корректные данные |

## **Требования к видам обеспечения**

* + 1. **Требования к математическому обеспечению системы:**

Стоимость проката книги = цена проката в день \* количество дней

Общая сумма проката = сумма стоимостей проката книги за указанный период

Штраф = цена проката в день\*количество просроченных дней

* + 1. **Требования информационному обеспечению системы:**

К данным читателя относятся:

* ФИО
* ID
* Адрес
* Телефон
* История прокатов

К данным книги относятся:

* Название
* ID
* Место хранения
* Тип
* Издательство
* Год
* Автор
* Жанр
* Аннотация
* Количество томов
* Переводчик
* Цена книги
* Цена проката
* Оригинальное название
* Количество страниц
* Дата приобретения библиотекой
* История прокатов

К данным пользователя относятся:

* ФИО
* Телефон
* ID
* Должность
* Код доступа

К данным нового проката относится:

* ID читателя
* ФИО читателя
* Дата
* Список книг и дополнительных услуг
* Доставка
* Цена

Список всех прокатов/возвратов включает:

* Клиент
* Дата
* Книга
* Статус
* Цена
* Доставка

Список всех доп. услуг включает:

* Дата
* Клиент
* ID услуги
* Количество
* Стоимость

Дополнительные услуги:

* Название услуги
* ID услуги
* Описание
* Цена

Базы данных будут созданы с помощью MySQL.

* + 1. **Требования к программному обеспечению системы**
* ОС: Windows 7/8/10
* Версия.NET: 4.6
* Повсеместное использование кодировки UTF-8
  + 1. **Требования к техническому обеспечению**

Технические средства, требуемые для функционирования ПО: настольный компьютер:

* Процессор с тактовой частотой 1,6 ГГц или большей;
* 2 ГБ ОЗУ;
* 10 ГБ доступного пространства на жестком диске;
* Жесткий диск не менее 5400 об/мин;
* Монитор;
* Клавиатура;
* Мышь;

# Состав и содержание работ по созданию ПО

|  |  |  |
| --- | --- | --- |
| **№** | **Содержание** | **Срок выполнения** |
| **1** | Разработать техническое задание на программное решение задачи. | 07.04.2017 |
| **2** | Разработать диаграммы прецедентов и диаграмму классов UML по техническому заданию 1-й работы. | 14.04.2017 |
| **3** | Разработать диаграммы действий, последовательностей и состояний UML по техническому заданию и диаграммам предыдущих работ. | 21.04.2017 |
| **4** | Определить интерфейсы классов, разработать диаграммы компонентов и размещения UML по проекту АИС | 28.04.2017 |
| **5** | Разработать проект интерфейса программной системы по техническому заданию и диаграммам предыдущих работ. | 05.05.2017 |
| **6** | Реализовать приложение с помощью методик объектно-ориентированного программирования. | 19.05.2017 |
| **7** | Использовать реляционную БД для хранения данных программы (MsSQL, MySQL или др.). | 02.06.2017 |

# Порядок контроля и приемки системы

Для каждого этапа приёма работы должны проводиться опытные эксплуатации и тестирования и лишь затем приём её со стороны заказчика:

Порядок сдачи работ:

* Сдача работ производится при присутствии комиссии. Комиссия состоит из представителей заказчика и исполнителя.
* Если в ходе приемки работ обнаруживается несоответствие требованиям, представленным в настоящем документе, заказчик вправе требовать доработки системы.
* При необходимости техническое задание может быть дополнено другими разделами, или сокращено путем удаления нецелесообразных пунктов при обоюдном согласии сторон

# Требования к составу и содержанию работ по подготовке объекта автоматизации к вводу системы в действие

Для обеспечения готовности объекта к вводу системы в действие провести комплекс мероприятий:

* Приобрести компоненты технического и программного обеспечения, заключить договора на их лицензионное использование;
* Завершить работы по установке технических средств;
* Провести обучение пользователей.

# Требования к документированию

Состав программной документации должен быть согласован с Заказчиком и разработан в соответствии с ГОСТ 34.201 «Виды, комплектность и обозначение документов при создании автоматизированных систем».

1. **Источники разработки**

* Договор №1234 от 04.04.2017 на разработку АС «УФБ», заключенный между ООО «Букинист» и ЗАО «Дантес»;
* ГОСТ 34.602-89. «Техническое задание на создание автоматизированной системы»;
* ГОСТ 34.201-89 «Виды, комплектность и обозначение документов при создании автоматизированных систем»;
* ГОСТ ЕСПД – «Единая система программной документации»;

## **Приложение 2. Таблицы описания прецедентов**

Таблица 3 Описание прецедента "Просмотр списка клиентов"

|  |  |
| --- | --- |
| Имя прецедента | Просмотр списка клиентов |
| Краткое описание | Прецедент позволяет просматривать список всех клиентов |
| Актёры | Библиотекарь, библиограф, администратор |
| Предусловия | Должен быть хотя бы один клиент. Если таковых нет, то пользователю предлагается создать нового клиента. |
| Основной поток | Выводится информация обо всех клиентах в виде таблицы |
| Постусловия | Если прецедент был успешно завершен, то пользователь может выполнять другие действия. |
| Альтернативные потоки | — |

Таблица 4 Описание прецедента "Добавление клиента"

|  |  |
| --- | --- |
| Имя прецедента | Добавление клиента |
| Краткое описание | Прецедент позволяет добавлять новых клиентов |
| Актёры | Библиотекарь, администратор |
| Предусловия | Должен быть выполнен прецедент “Просмотр списка клиентов” |
| Основной поток | Открывается форма добавления нового клиента, куда заносятся все необходимые данные. |
| Постусловия | Если прецедент был успешно завершен, то форма добавления клиента закрывается, и в БД добавляется новый клиент. |
| Альтернативные потоки | Если имеются пустые поля, либо введены некорректные данные, то пользователю выдается сообщение об ошибке и дается возможность исправить данные. |

Таблица 5 Описание прецедента "Изменение данных о клиенте"

|  |  |
| --- | --- |
| Имя прецедента | Изменение данных о клиенте |
| Краткое описание | Прецедент позволяет редактировать данные о клиенте |
| Актёры | Библиотекарь, администратор |
| Предусловия | Должен быть выполнен прецедент “Просмотр списка клиентов” |
| Основной поток | Открывается форма редактирования клиента, где изменяются необходимые данные. |
| Постусловия | Если прецедент был успешно завершен, то форма редактирования клиента закрывается, и измененные данные о клиенте сохраняются |
| Альтернативные потоки | Если имеются пустые поля, либо введены некорректные данные, то пользователю выдается сообщение об ошибке и дается возможность исправить данные. |

Таблица 6 Описание прецедента " Удаление клиента"

|  |  |
| --- | --- |
| Имя прецедента | Удаление клиента |
| Краткое описание | Прецедент позволяет удалять клиента из БД |
| Актёры | Библиотекарь, администратор |
| Предусловия | Должен быть выполнен прецедент “Просмотр списка клиентов” |
| Основной поток | При нажатии на кнопку “Удалить”, текущая запись удаляется. |
| Постусловия | Если прецедент был успешно завершен, то клиент удаляется из базы данных. |
| Альтернативные потоки | — |

Таблица 7 Описание прецедента “Просмотр списка книг”

|  |  |
| --- | --- |
| Имя прецедента | Просмотр списка книг |
| Краткое описание | Прецедент позволяет просматривать список всех книг |
| Актёры | Библиограф, библиотекарь, администратор |
| Предусловия | Должна быть хотя бы одна книга |
| Основной поток | Выводится информация обо всех книгах в виде таблицы. Таблица содержится как в карточке клиента (в ней содержатся все книги конкретного клиента), так и отдельно (в ней содержится список всех книг) |
| Постусловия | Если прецедент был успешно завершен, то пользователь может выполнять другие действия. |
| Альтернативные потоки | — |

Таблица 8 Описание прецедента "Добавление книги"

|  |  |
| --- | --- |
| Имя прецедента | Добавление книги |
| Краткое описание | Прецедент позволяет добавлять новые книги |
| Актёры | Библиограф, администратор |
| Предусловия |  |
| Основной поток | Открывается форма добавления книги, куда вносятся необходимые данные. |
| Постусловия | Если прецедент был успешно завершен, то в БД добавляется новая книга. |
| Альтернативные потоки | Если имеются пустые поля, либо введены некорректные данные, то пользователю выдается сообщение об ошибке и дается возможность исправить данные. |

Таблица 9 Описание прецедента "Изменение данных о книге"

|  |  |
| --- | --- |
| Имя прецедента | Изменение данных о книге |
| Краткое описание | Прецедент позволяет редактировать данные о книге |
| Актёры | Библиограф, администратор |
| Предусловия | Должен быть выполнен прецедент “Просмотр списка книг” |
| Основной поток | Открывается форма редактирования книги, где изменяются необходимые данные. |
| Постусловия | Если прецедент был успешно завершен, то форма редактирования книг закрывается, и измененные данные сохраняются |
| Альтернативные потоки | Если имеются пустые поля, либо введены некорректные данные, то пользователю выдается сообщение об ошибке и дается возможность исправить данные. |

Таблица 10 Описание прецедента "Удаление книги"

|  |  |
| --- | --- |
| Имя прецедента | Удаление книги |
| Краткое описание | Прецедент позволяет удалять книги |
| Актёры | Библиограф, администратор |
| Предусловия | Должен быть выполнен прецедент “Просмотр списка книг” |
| Основной поток | При нажатии на кнопку “Удалить”, текущая запись удаляется. |
| Постусловия | Если прецедент был успешно завершен, то книга удаляется из базы данных. |
| Альтернативные потоки | — |

Таблица 9 Описание прецедента “Просмотр списка прокатов”

|  |  |
| --- | --- |
| Имя прецедента | Просмотр списка прокатов |
| Краткое описание | Прецедент позволяет просматривать список всех прокатов |
| Актёры | Администратор, библиотекарь |
| Предусловия | Должен быть хотя бы один прокат |
| Основной поток | Выводится информация обо всех прокатах в виде таблицы. |
| Постусловия | Если прецедент был успешно завершен, то пользователь может выполнять другие действия. |
| Альтернативные потоки | — |

Таблица 10 Описание прецедента "Добавление проката"

|  |  |
| --- | --- |
| Имя прецедента | Добавление проката |
| Краткое описание | Прецедент позволяет добавлять новые книги |
| Актёры | Библиотекарь, администратор |
| Предусловия | Должны быть выполнены прецеденты “Просмотр списка книг”, “Просмотр списка читателей” |
| Основной поток | Открывается форма добавления проката, куда вносятся необходимые данные. |
| Постусловия | Если прецедент был успешно завершен, то в БД добавляется новый прокат. |
| Альтернативные потоки | Если имеются пустые поля, либо введены некорректные данные, то пользователю выдается сообщение об ошибке и дается возможность исправить данные. |

Таблица 11 Описание прецедента "Изменение данных о прокате"

|  |  |
| --- | --- |
| Имя прецедента | Изменение данных о прокате |
| Краткое описание | Прецедент позволяет редактировать данные о прокате |
| Актёры | Библиотекарь, администратор |
| Предусловия | Должен быть выполнен прецедент “Просмотр списка прокатов” |
| Основной поток | Открывается форма редактирования проката, где изменяются необходимые данные. |
| Постусловия | Если прецедент был успешно завершен, то форма редактирования закрывается, и измененные данные сохраняются |
| Альтернативные потоки | Если имеются пустые поля, либо введены некорректные данные, то пользователю выдается сообщение об ошибке и дается возможность исправить данные. |

Таблица 12 Описание прецедента "Удаление проката"

|  |  |
| --- | --- |
| Имя прецедента | Удаление проката |
| Краткое описание | Прецедент позволяет удалять прокат |
| Актёры | Библиотекарь, администратор |
| Предусловия | Должен быть выполнен прецедент “Просмотр списка прокатов” |
| Основной поток | При нажатии на кнопку “Удалить”, текущая запись удаляется. |
| Постусловия | Если прецедент был успешно завершен, то пользователь удаляется из базы данных. |
| Альтернативные потоки | — |

Таблица 13 Описание прецедента "Составление отчета"

|  |  |
| --- | --- |
| Имя прецедента | Составление отчета |
| Краткое описание | Прецедент позволяет составлять отчеты по выбранным данным |
| Актёры | Администратор |
| Предусловия | Должна быть необходимая информация по отчетным данным |
| Основной поток | Отчет формируется в виде графика |
| Постусловия | Если прецедент был успешно завершен, то пользователь может выполнять другие действия. |
| Альтернативные потоки | — |

Остальные прецеденты аналогичны прецедентам, описанным выше.

## **Приложение 3. Пользовательский интерфейс**

Настольное приложение

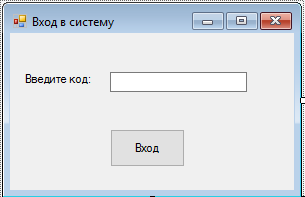


Рисунок 36. Форма входа в систему

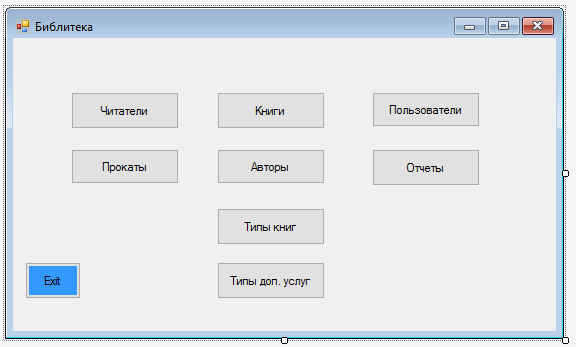


Рисунок 37. Главная форма

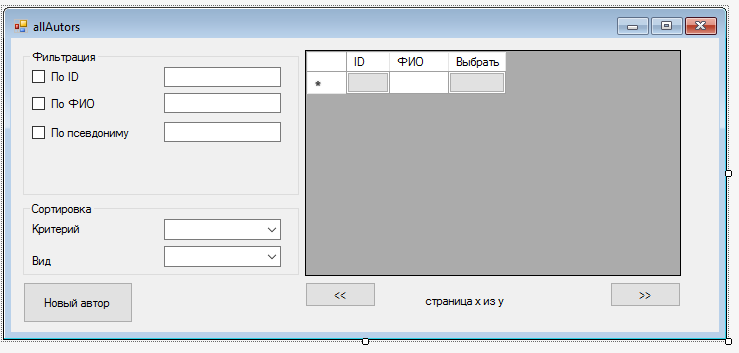


Рисунок 38. Список авторов

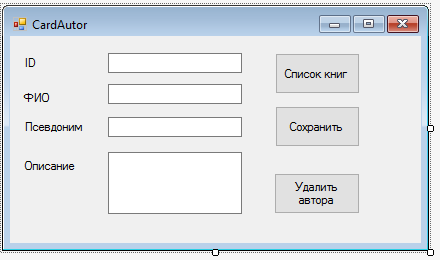


Рисунок 39. Карточка автора

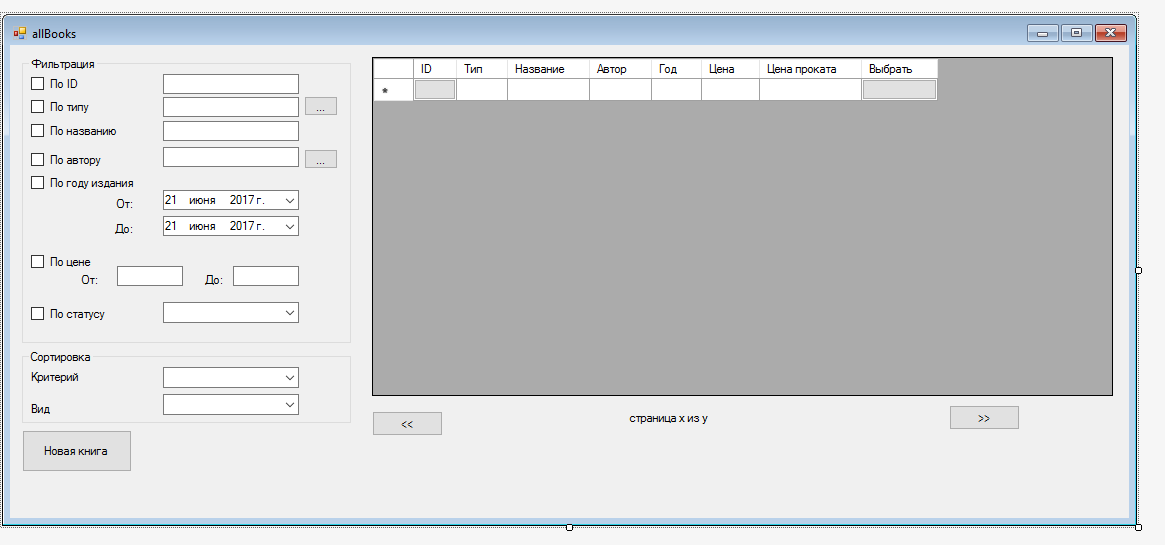


Рисунок 40. Список книг

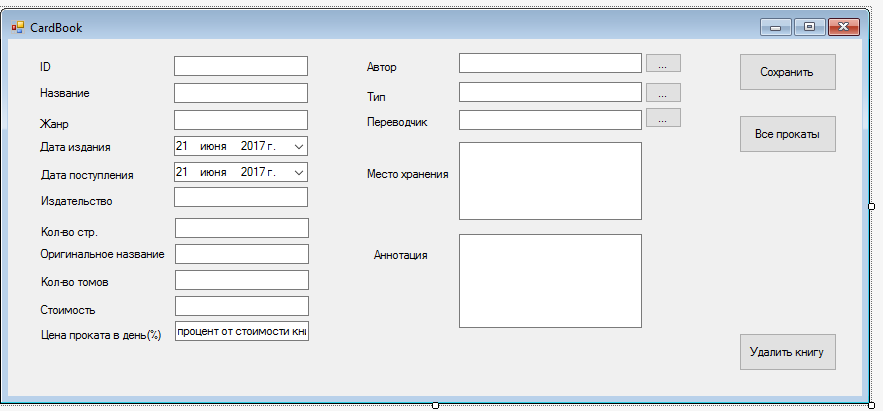


Рисунок 41. Карточка книги

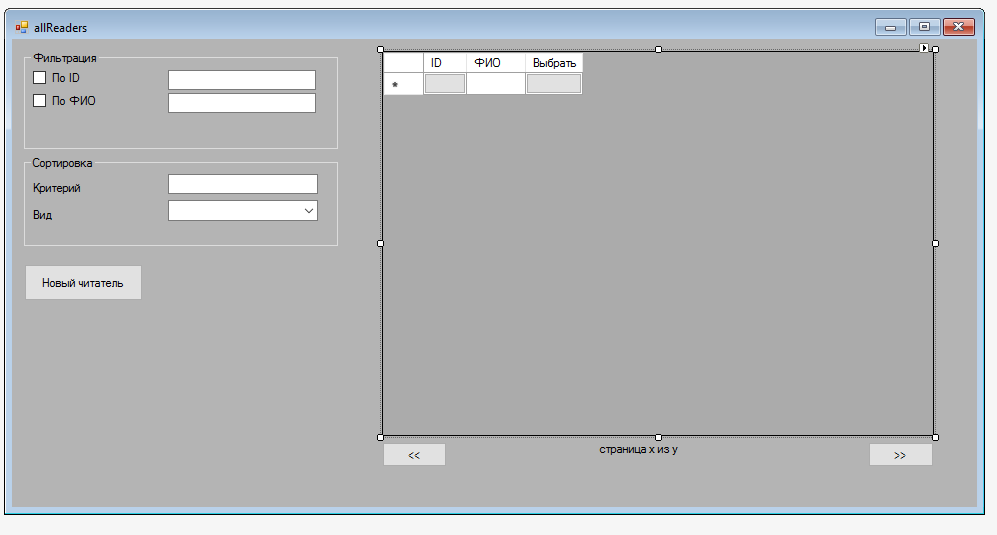


Рисунок 42. Список читателей

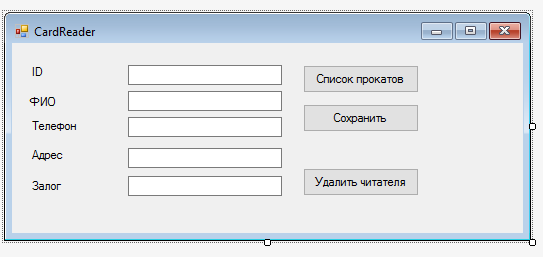


Рисунок 43. Карточка читателя

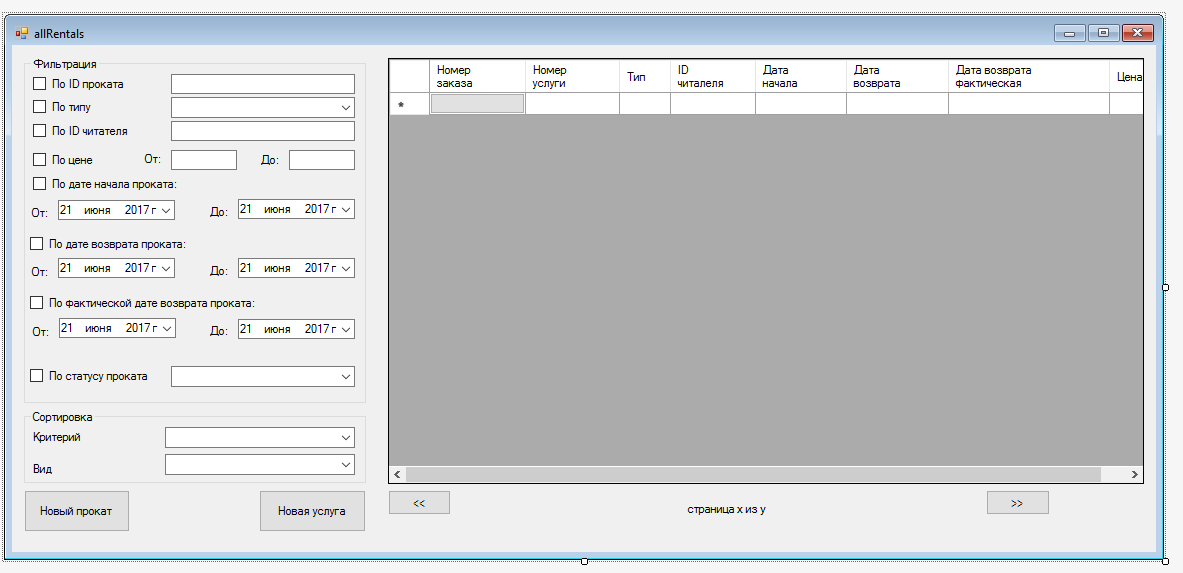


Рисунок 44. Все прокаты

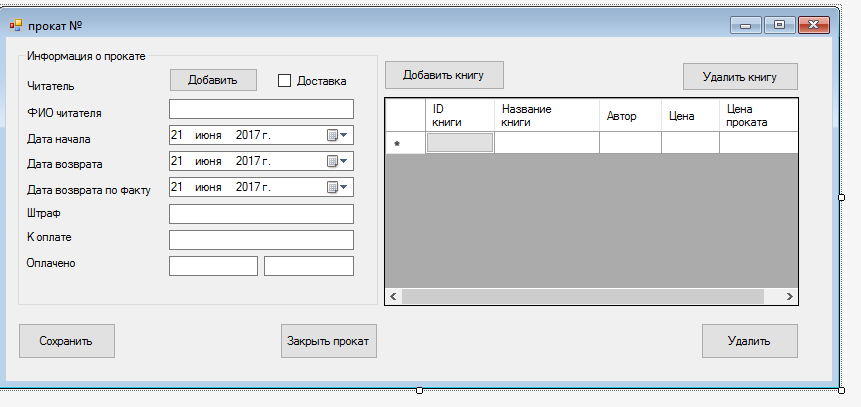


Рисунок 45. Карточка проката

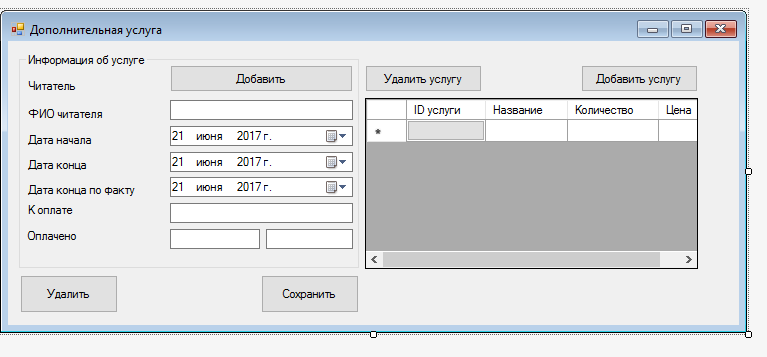


Рисунок 46. Карточка услуги

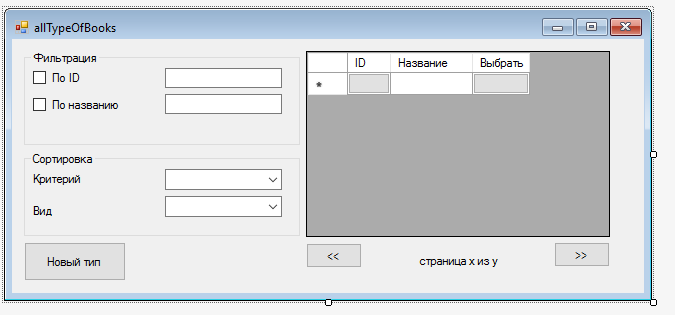


Рисунок 47. Список типов книг

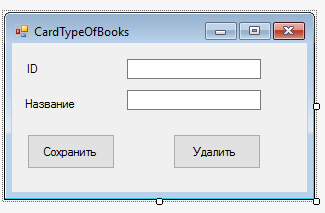


Рисунок 48. Карточка типа книги

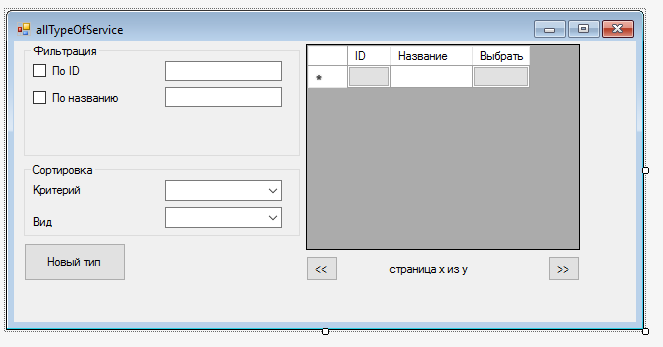


Рисунок 49. Список видов услуг

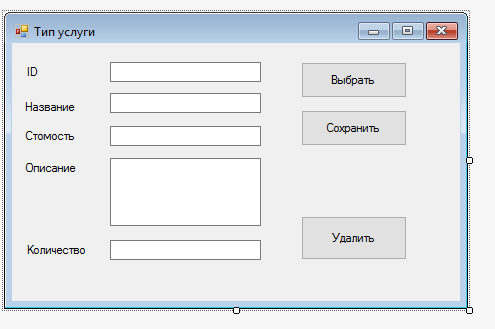


Рисунок 50. Карточка вида услуги

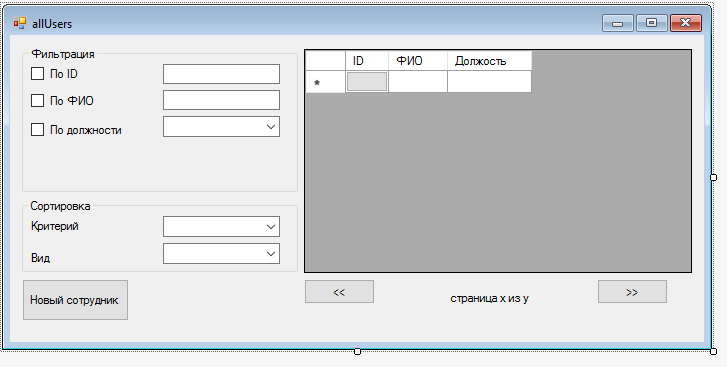


Рисунок 51. Список всех пользователей системы

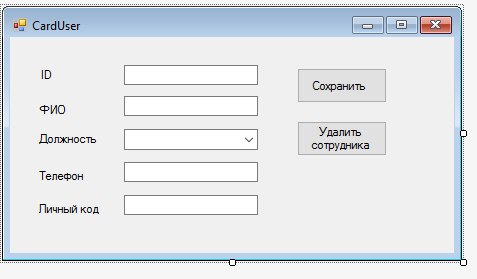


Рисунок 52. Карточка пользователя

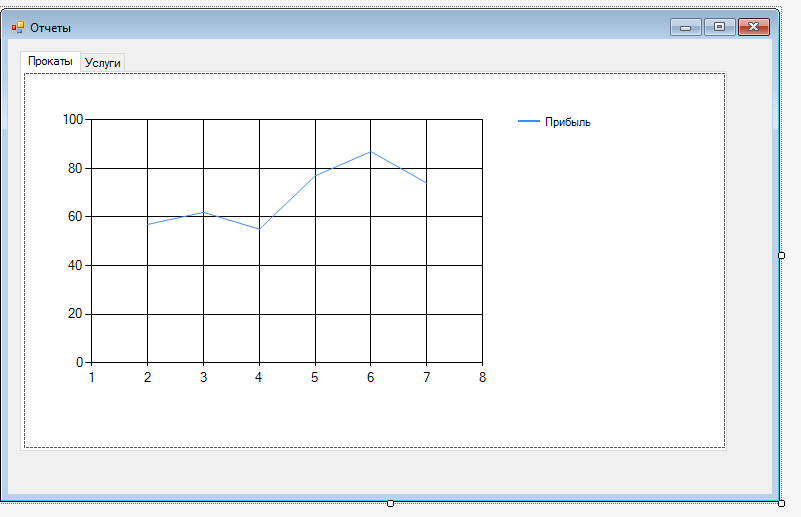


Рисунок 53. Отчет

Веб-приложение

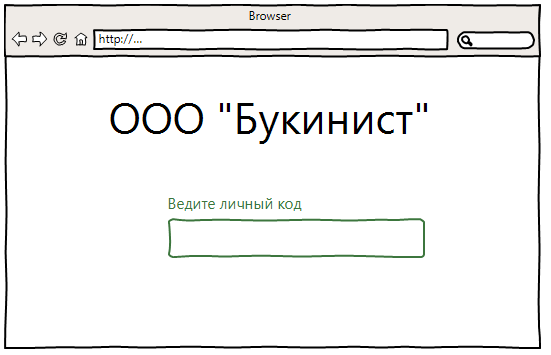


Рисунок 54. Форма входа

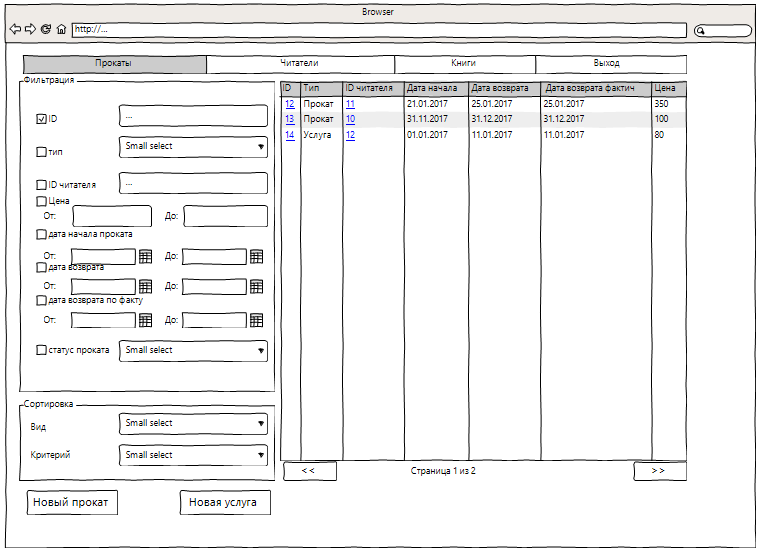


Рисунок 55. Все прокаты

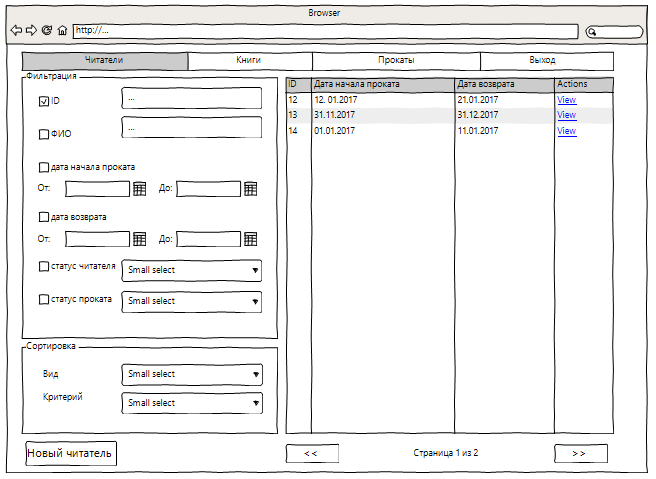


Рисунок 56. Все читатели

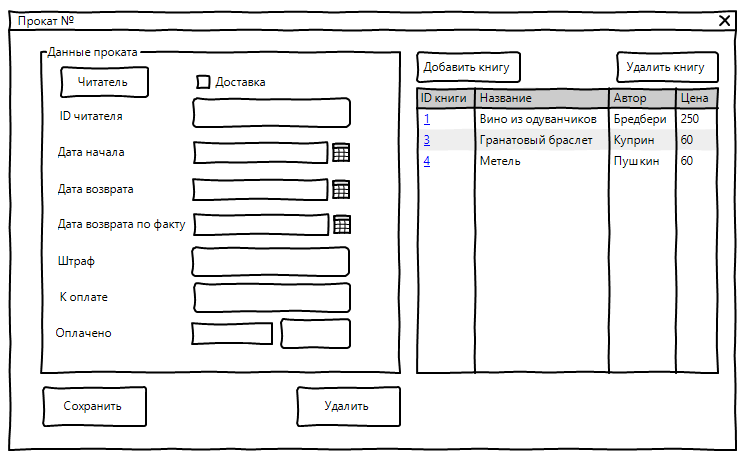


Рисунок 57. Карточка проката

1. Вероятность того, что в пределах заданной наработки (продолжительность или объем работы объекта, измеряемая в часах (56 часов в неделю)) отказ объекта не возникнет [↑](#footnote-ref-1)
2. Вероятность того, что время восстановления работоспособного состояния объекта не превысит заданное значение [↑](#footnote-ref-2)