МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждения образования «БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ

ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет Информационных технологий

Кафедра Программной инженерии

Специальность 1-40 01 01 Программное обеспечение информационных технологий

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

**КУРСОВОГО ПРОЕКТА:**

по дисциплине «Объектно-ориентированные технологии программирования и стандарты проектирования »

Тема «Программное средство «Читательский дневник»»

Исполнитель

студент 2 курса 6 группы Мамаева Диана Александровна

(Ф.И.О.)

Руководитель работы Пахолко А. С

(подпись, Ф.И.О.)

Курсовой проект защищен с оценкой

(подпись)

Минск 2022

**Содержание**

[Введение 4](#_Toc104265914)

[1. Обзор литературы и аналогов 5](#_Toc104265915)

[1.1 All My Books 5](#_Toc104265916)

[1.2 BOOKSHELF 6](#_Toc104265917)

[1.3 Bookmate 6](#_Toc104265918)

[1.4 WPF | ListBox (professorweb.ru) 7](#_Toc104265919)

[1.5 Fb2Library 7](#_Toc104265920)

[1.6 WCF RIA Services 8](#_Toc104265921)

[2. Моделирование предметной области и разработка функциональных требований 9](#_Toc104265922)

[2.1 Определение требований к программному средству 9](#_Toc104265923)

[2.2 Описание средств разработки 9](#_Toc104265924)

[2.3 Описание функциональности программного средства 10](#_Toc104265925)

[2.4 Спецификация функциональных требований 10](#_Toc104265926)

[3. Проектирование программного средства 12](#_Toc104265927)

[3.1 Разработка базы данных 12](#_Toc104265928)

[3.2 Обобщенная структура программы 13](#_Toc104265929)

[3.2.1 Структура хоста 13](#_Toc104265930)

[3.2.2 Структура сервиса 13](#_Toc104265931)

[3.3.3 Структура клиента 14](#_Toc104265932)

[4. Архитектура проектируемого программного средства 15](#_Toc104265933)

[4.1.1 Реализация MVVM 15](#_Toc104265934)

[4.1.2 Схема и описание авторизации 16](#_Toc104265935)

[4.1.2 Создание современного дизайна в WPF 16](#_Toc104265936)

[4.1.2 Чтение Fb2 файлов 16](#_Toc104265937)

[4.1.4 Подключение к БД 18](#_Toc104265938)

[5. Описание интерфейсов программного средства, его тестирование и анализ полученных данных 19](#_Toc104265939)

[5.1 Описание интерфейсов программного средства 19](#_Toc104265940)

[5.1.1 Вход в качестве пользователя 20](#_Toc104265941)

[5.1.2 Вход в качестве администратора 25](#_Toc104265942)

[5.2 Тестирование программное средства и анализ полученных данных 25](#_Toc104265943)

[6. Руководство по использованию 28](#_Toc104265944)

[Заключение 29](#_Toc104265945)

[Список использованных источников 30](#_Toc104265946)

[Приложение А 31](#_Toc104265947)

[Приложение Б 32](#_Toc104265948)

## Введение

Все знают, что чтение определенно приносит много пользы для человека. Люди стали очень мобильными: одни читают новости "на ходу", другие слушают аудиокнигу по пути на работу, у третьих не хватает времени на литературу.

Сегодня на свой смартфон можно загрузить целую библиотеку, вести читательский дневник, оценивать прочитанные книги, писать рецензии и даже публиковать свои произведения.

Все больше любителей книг заводят читательские дневники, как электронные, так и рукописные. Читательский дневник — это блокнот или тетрадь, в которую человек заносит необходимые для него пометки о прочитанных книгах.

Книга не пропадает в пучине мыслей, а встаёт на полочку воспоминаний. Текст о книге похож на расследование: пытаешься копнуть глубже, чем обычно.

Такой дневник могут использовать дети и взрослые. Для детей будет проще читать и запоминать большой объём школьной литературы. А для взрослых глубже понимать литературы и читать с удовольствием.

Но нам не всегда хватает собственных мыслей что бы понять весь глубокий замысел той или иной книги.

А что делать, когда не знаешь, что почитать? Как удобно почитать или послушать скаченные книжки? Где арты, которые передадут весь антураж книги?

Есть множество приложений, которые помогут с этими вопросами, но хочется ли иметь на своем рабочем столе множество разных приложений, и быть зарегистрированным на множестве форумов? Думаю нет.

При написании курсового проекта задачей являлось создать многопользовательское приложения с простым, удобным и приятным интерфейсом, для пользователей любых возрастов.

Целью данной работы является создание приложения, которое работает с общей базой данных, доступной для всех пользователей. Также функционал приложения позволяет:

* добавлять/изменять/удалять данные из базы данных;
* получать статистику прочитанных книг;
* добавлять отзывы;
* изменять статус чтения;
* добавлять файлы/аудиофайлы содержащие книги;
* просматривать/добавлять арты;
* поиск книг по жанрам/тегам/название/авторам/циклам;
* просматривать/изменять книги в закладках;
* просматривать информацию о книгах/авторах;

Структура приложения: приложение, написанное на С#. Оно устанавливается на ПК и представляет собой интерфейс.

ПК приложение для оптимальной работы требует:

* установленный Microsoft SQL Server 2019 или DataGrip 2021.3.4;
* установленный .Net framework 5.0.202 или более поздней версии.

1. Обзор литературы и аналогов

Изучая, программы-прототипы, были найдены схожие по теме и функционалу приложения. Большая часть программ представляет собой приложение для смартфонов. Так же были найдены статьи, которые помогли реализовать некоторые функции. Результат анализа представлен ниже.

## All My Books

Данное приложения было разработано для практически всех операционных систем. Но мы рассмотрим версию для Windows.

Приложение имеет следующие возможности:

* автоматическая загрузка информации о книге из Озона, Лабиринта, MyBookList;
* каталогизация всех вариантов книг - печатных, электронных и аудио;
* полезная и удобная статистика по вашей коллекции;
* возможность [вести список прочитанных книг ("читательский дневник")](https://www.bolidesoft.com/rus/allmybooks/keep-track-book.html) с комментариями.

All My Books — одна из лучших программ для учета книг, ведь ручная работа по созданию каталога в программе сведена к минимуму. Вам не потребуется вводить информацию о книге вручную, All My Books загрузит всю доступную информацию из онлайн-источников в считанные секунды. Интерфейс приложения предоставлен на рисунке 1.1

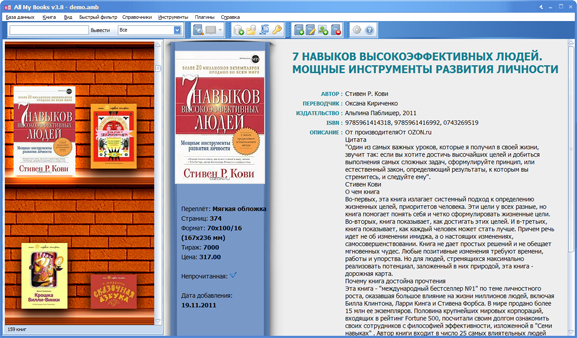


Рисунок 1.1 − Интерфейс приложения All My Books

Но все же дизайн данного приложения является устаревшим.

## BOOKSHELF

Данное приложение так же было разработано под мобильные операционные системы IOS и Android.

Настоящая находка для тех, кто любит вести учет прочитанного. Здесь можно:

* записывать прочитанные книги, писать собственные рецензии, создавать виртуальные книжные полки по жанрам (или придумать собственное деление) и добавлять книги в список желаний;
* выводит статистику: вы можете узнать, сколько прочитали за год или месяц, сколько в среднем страниц в день читаете и какова средняя толщина ваших книг;

Интерфейс данного приложения предоставлен на рисунке 1.2

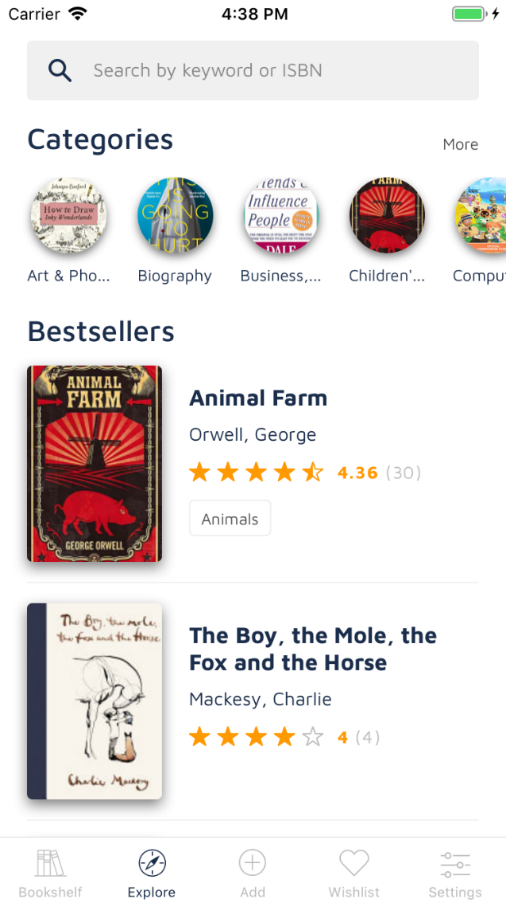


Рисунок 1.2 − Интерфейс приложения BOOKSHELF

## Bookmate

Платформы: Web, Windows.

Поддерживаемые форматы: FB\*2, ePub.

Этот сервис, как и «Google Play Книги», предлагает владельцам компьютеров читать произведения на сайте. Кроме того, пользователи Windows могут установить настольный клиент Bookmate, позволяющий добавлять тексты в личную библиотеку и открывать их офлайн.

Интерфейс приложения представлен на рисунке 1.3

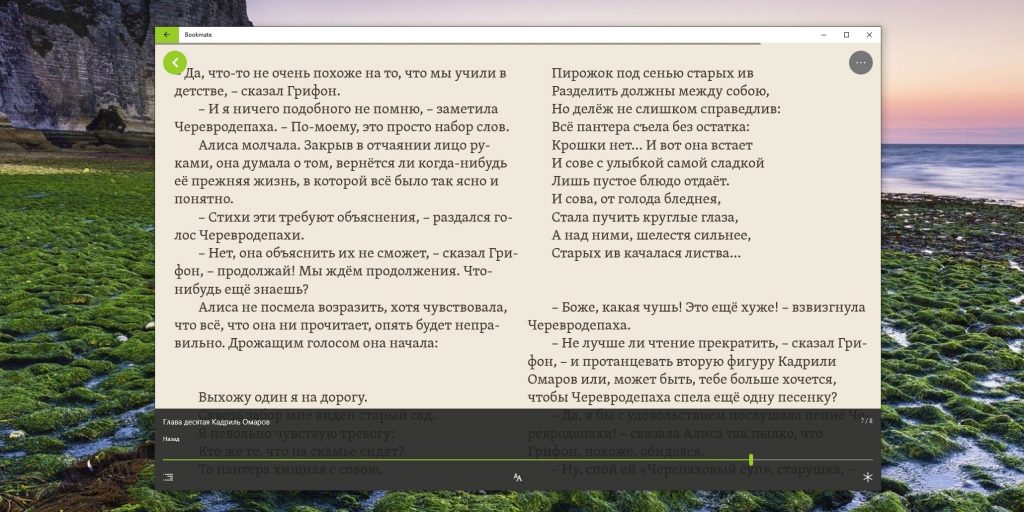
[[](https://cdn.lifehacker.ru/wp-content/uploads/2021/01/VirtualBox_Windows-10_29_01_2021_14_26_21_1611908808-e1611908837836.jpg)](https://cdn.lifehacker.ru/wp-content/uploads/2021/01/VirtualBox_Windows-10_29_01_2021_14_26_21_1611908808-e1611908837836.jpg" \o ")

Рисунок 1.3 − Интерфейс приложения Bookmate

В обеих версиях программы к вашим услугам настройки шрифта, фона, отступов и других визуальных элементов. Закладки, позиции чтения и прочие метаданные синхронизируются между устройствами. Приложение может слегка подтормаживать, но в целом пользоваться им комфортно.

## WPF | ListBox (professorweb.ru)

ListBox – как средство вывода списка данных. В отличии от более нового элемента DataGrid, отображать данные можно как угодно. ListBox является довольно гибким элементом управления. Он может хранить не только объекты ListBoxItem, но и любые произвольные элементы. Ведь класс ListBoxItem является наследником класса ContentControl, который позволяет хранить фрагменты вложенного содержимого. Если такой фрагмент является классом, порожденным от UIElement, то он будет отображен в элементе ListBox. Если же это другой тип объекта, ListBox вызовет метод ToString() и выведет полученный текст.

Для вывода данных был выбран ListBox, из-за его гибкости и удобства.

## Fb2Library

Этот проект предоставляет библиотеку .Net, помогающую разработчикам загружать FB2 файлы электронных книг. Этот формат файлов электронных книг широко распространен в России.

Библиотека считывает формат fb2-файлы и преобразует его в объект, представляющий дерево данных. В данном объекте описывается вся информация, о файле. Дерево данных представляет собой несколько узлов, хранящих названия и дочерние элементы. Данное дерево можно преобразовать в XML формат и работать уже с ним.

## WCF RIA Services

Для реализации клиент серверного приложения отлично подойдет WCF RIA Services. Он упростит реализацию свези между сервисом и клиентом.

WCF RIA Services предлагает автоматизацию пересылки данных и создания многоуровневых приложения на основе Silverlight, фокусируя внимание разработчика именно на логике работы. Вы пишите только один раз экземпляр серверного кода, а он автоматически переносится и становится доступным на клиентской стороне, без необходимости ручного дублирования или применения других ухищрений со стороны разработчика. Так же на клиенте остаются доступны все серверные возможности, такие как валидация, права доступа и многие др.

В этом цикле статей были описаны основные возможности WCF RIA Services, и показан потомное создание WCF RIA Services.На диаграмме ниже показано, за что именно отвечает WCF RIA Services:

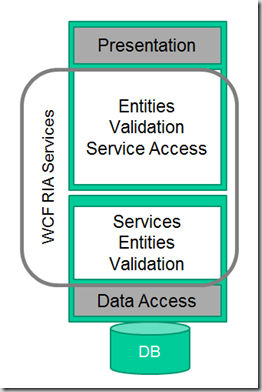


Рисунок 1.3 − Диаграмма работы WCF

# Моделирование предметной области и разработка функциональных требований

При разработке курсового проекта были определены тебования к программному средству.

## Определение требований к программному средству

Функционально ПС должно выполнять следующие задачи:

* регистрация и авторизация пользователей;
* сохранение информации в базе данных;
* для администратора одобрение или отклонение предложенных книг;
* для пользователя изменение личных данных;
* для пользователя вывод сохраненных книг;
* для пользователя просмотр списка книг;
* для пользователя добавлять отзывы и оценки к книгам;
* для пользователя добавление картинок;
* для пользователей добавление книг;
* для пользователей чтение файлов.

## 2.2 Описание средств разработки

При разработке приложения были использованы:

* интегрированная среда разработки Microsoft Visual Studio 2022;
* программная платформа .NET Framework 4.7.2;
* язык программирования C#;
* расширяемый язык разметки XAML;
* технология WPF;
* технология ADO.NET EDM;
* Microsoft SQL Server;
* Платформа WCF.

В качестве интерфейса прикладного программирования был выбран обширный API-интерфейс – Windows Presentation Foundation (WPF), предназначенный для создания настольных программ с графически насыщенным пользовательским интерфейсом. В основе графической технологии WPF лежит мощная инфраструктура, основанная на DirectX. Это является одним из основных отличий WPF от более ранней технологии создания пользовательских интерфейсов – Windows Forms. Для работы с WPF использовался объектно-ориентированный язык программирования с С-подобным синтаксисом – С#, разработанный для создания приложений на платформе Microsoft .NET Framework.

Есть различные механизмы работы с БД в С#. Из-за большого количества таблиц в моей базе данных, был выбран ADO.Net EDM, который предоставляет самый прямой способ доступа к данным в .Net Framework.

Для разделения клиентской и сервисной области была выбрана платформа WСF. Windows Communication Foundation (WCF) — это платформа для создания приложений, ориентированных на службы. С помощью WCF можно передавать данные в виде асинхронных сообщений из одной конечной точки службы в другую. Конечная точка службы может входить в постоянно доступную службу, размещаемую в IIS, или представлять службу, размещаемую в приложении. Конечная точка может быть клиентом службы, которая запрашивает данные от конечной точки службы. Сообщения могут представлять одиночный символ или одно слово, отправляемое в формате XML, или иметь вид сложного потока двоичных данных.

## 2.3 Описание функциональности программного средства

Описание функциональности программного средства представлено с помощью UML-диаграммы вариантов использования (Приложение А).

Спецификация функциональности программного средства представлена в следующем пункте.

## 2.4 Спецификация функциональных требований

Для функциональности ПС необходимо создание базы данных для хранения информации приложения. Подробно о базе данных описано в главе 3.

В программном средстве «VioletBookDiary» при в запуске программы необходима авторизация или регистрация, если пользователь еще не зарегистрирован в программе. Для авторизации необходимо указать почту и пароль. Регистрация отличается лишь тем, что нужно подтвердить пароль, повторив его в другом поле. Если пороли не совпадут пользователь не сможет зарегистрироваться. Другие данные пользователя, такие как: имя, фото профиля, и описание пользователя, генерируются автоматически. В дальнейшем пользователь сможет изменить некоторые данные своего профиля.

При авторизации пользователя, ему становятся доступно окно, визуализирующие имеющийся функционал. На главном окне пользователь сможет увидеть все имеющиеся в базе книги и при нажатии на её, откроется страница «Информация о книге» с доступной о ней информацией, такой же функцией обладают и другие страницы со списком книг. На данной странице пользователь сможет оставить отзыв, дав оценку этой книге, добавить картинку и сохранить книгу в личную библиотеку, отсортировав её на 4 категории: «Читаю», «Буду читать», «Прочитано» и «Брошено». Категория «Брошено» не отображается на странице «Информация о пользователе», так что можно считать, что пользователь удалит книгу из собственной библиотеки. Так же пользователю доступно чтение книги, это происходит на отдельной странице, где можно выбрать главу из списка или перемещаться по главам через кнопки.

Нажав на фото профиля пользователя, оставившего отзыв, можно увидеть информацию об этом пользователе. Так же на главной странице пользователь сможет просмотреть картотеку картинок. На странице «Каталог» можно найти книгу по названию или отфильтровать по доступным полям таким как: авторы, жанры, тэги.

На странице «Информации о пользователе», пользователь сможет изменить информацию о своем аккаунте. Для изменения доступны поля: имени, описания и фото профиля. Так же на этой странице пользователь сможет увидеть сохраненные в собственной библиотеки книги, разбиты на упомянутые ранее категории.

Так же пользователю доступно окна добавление книги, где он может описать книгу, некоторые поля будут автоматически генерировать новые записи в БД. При нажатии кнопки «Добавить книгу» пользователю выдаст сообщение об успешной операции. В ином случае выдаст сообщение об ошибке. На некоторых полях есть валидация и поэтому если поля красные кнопка «Добавить книгу» будет заблокирована.

В окне настроек пользователь может сменить язык или тему, изменения будут видны сразу же.

При авторизации администратора, ему кроме функций пользователя так же будет доступна страница с предложенными пользователями книгами, где администратор может просмотреть, одобрить или удалить предложение. Если администратор удаляет запись, то автоматически, кроме самой книге удаляются связи и бессвязные данные.

# Проектирование программного средства

## Разработка базы данных

Для обеспечения работоспособности программного средства была разработана база данных, что являлось обязательным условием курсового проекта. База данных была тщательно сконструирована до начала написания самого приложения. Для ее создания использовалась реляционная система управления базами данных (РСУБД) MS SQL.

Таким образом была разработана база данных ViolesDiary, которая включает в себя таблицы AUTHORIZED, USERS, BOOKS, PAINT, AUTHORS, FEEDBACK, GENRE, TAG, USER\_BOOKMARKS, BOOK\_AUTHORS, BOOK\_GENRE, BOOK\_TAG. Схема данных предоставлена на рисунке 3.1

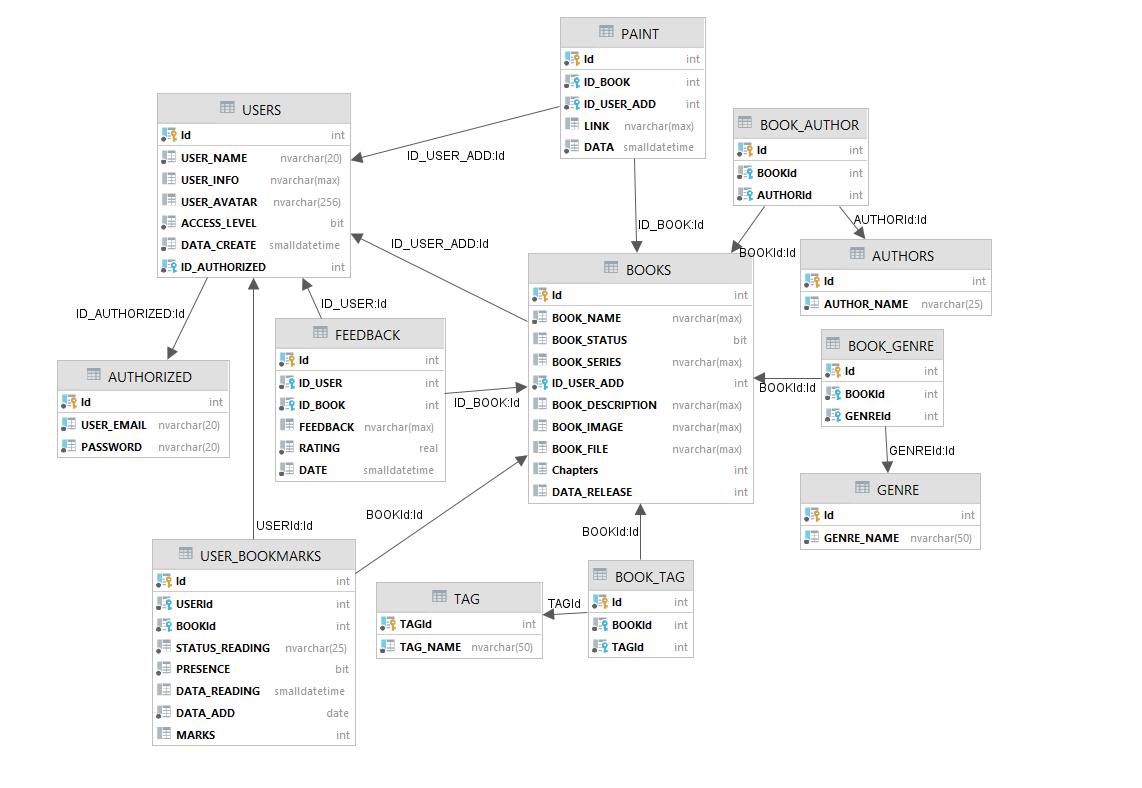


Рис. 3.1 – Схема базы данных VioletDiary

Для хранения данных для авторизации была создана таблица AOUTHORIZADE. Которая включает 3 столбца: Id – id авторизации, USER\_EMAIL – почта пользователя, PASSWORD – пароль.

Таблица USERS хранит информацию о пользователе, доступную и другим пользователям. В данной таблице можно отметить поле ACCESS\_LEVEL, которая идентифицирует, кем является данный пользователь: администратором – 1 или пользователем – 0.

Таблица PAINT используется для хранения картинок, определенной книги. Таблица связанна с таблицей USER и таблицей BOOK, данные связи соответственно показывают: какой пользователь добавит картинку и к какой книге эта картинку добавлена.

Таблицы TAG, GENRE, AUTHORS хранят соответственно тэги, жанры и авторов и связанны с таблицей книги по принципу многие ко многим. У каждой книге может быть несколько авторов, жанров и тэгом, как и наборов. Для реализации связей многие ко многим были созданы таблицы BOOK\_AUTHORS, BOOK\_GENRE, BOOK\_TAG.

Так же существует таблица FEEDBACK которая осуществляет функцию добавления отзывов для конкретной книги.

Таблице USER\_BOOKMARKS нужна для сохранения пользователем книге в собственной библиотеке. Так же в этой таблице хранится информация о последней открытой главе файла книги, что дает возможность вернутся к чтению, без запоминания главы.

## Обобщенная структура программы

Данное программное средство является многопользовательским, поэтому структура программы разделена на 3 проэкта: VDService(Сервис), DiaryHost(Хост), VioletBookDiary(клиент). 3 проекта для удобства помещенный в одно решение – VioletDiary, структура решения предоставлена на рисунке 3.2. Далее подробнее о назначении каждого.

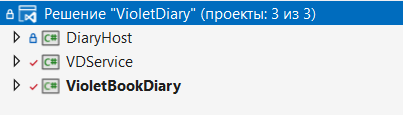


Рис. 3.2 – Структура решения VioletDiary

### **3.2.1 Структура хоста**

Хост (host) — в первую очередь это любое устройство, которое выполняет функции клиент-сервера. Проект выполняющий эту функцию называется DiaryHost и имеет стандартную структуру, с минимальным функционалом.

Для создания локального узла. Хост связывает другие два проекта настраивая между ними передачу данных.

### **3.2.2 Структура сервиса**

VDService – проект реализующий WCF сервис. Прежде всего, веб-сервисы (или веб-службы) — это технология. И как и любая другая технология, они имеют довольно четко очерченную среду применения.

По сути, веб-сервисы — это реализация абсолютно четких интерфейсов обмена данными между различными приложениями, которые написаны не только на разных языках, но и распределены на разных узлах сети.

Для программного средства VioletBookDiary, WCF выполняет функции, связанные с БД. Что дает возможность реализации передачи данных из БД в локальной сети.

Для реализации WCF сервиса были использованы паттерны Repository и UnitOfWork. Данные паттерны реализуют удобный и кратчайший способ доступа к данным БД.

### **3.3.3 Структура клиента**

VioletDiaryBook – проект реализующий интерфейс клиента. В проете реализован паттерн MVVM. Паттерн MVVM (Model-View-ViewModel) позволяет отделить логику приложения от визуальной части (представления). Данный паттерн является архитектурным, то есть он задает общую архитектуру приложения. Из этого следует что проект имеет следующую структуру:

* Model, View, ViewModel - компоненты реализующие MVVM;
* Current - статические классы, хранящие глобальные данные;
* Commands – реализация паттерна ICommand.

Взаимосвязь между классами хорошо видна в диаграмме классов (Приложение Б).

Приложение включает в себя 5 окон и 12 страниц. Связи между ними представлены на рисунке 3.3.

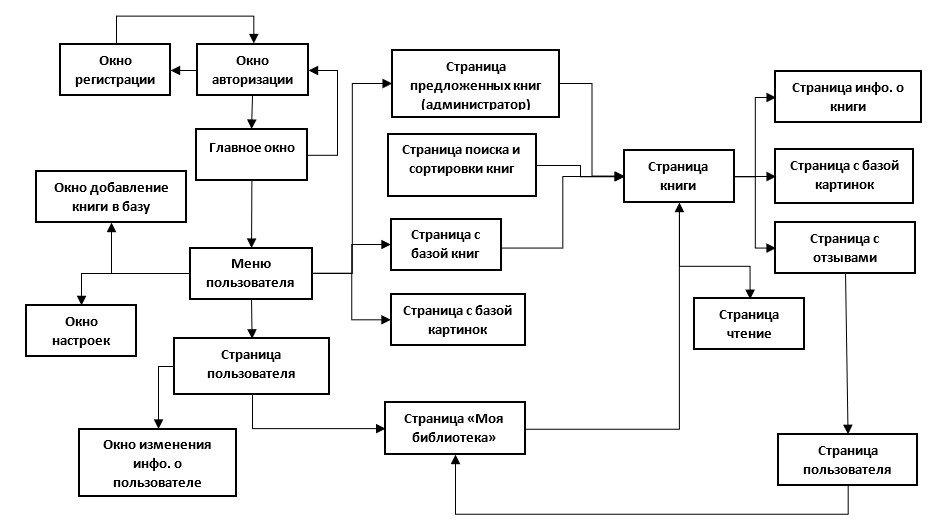


Рис. 3.3 – Схема связи между окнами

# Архитектура проектируемого программного средства

При создании программного средства использовался архитектурный паттерн Model-View-ViewModel (MVVM). Он был представлен архитектором программного обеспечения компании Microsoft Джоном Госсманом в 2005 году. Одной из главных целей шаблона является отделение и абстрагирование видимого пользовательского интерфейса от логики представления. MVVM состоит из трех компонентов: модель (Model), представление (View) и модель представления (ViewModel). Наглядное представление паттерна представлено на рисунке 4.1.

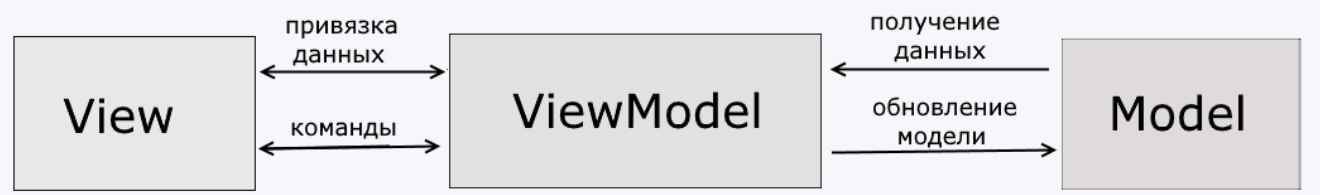


Рисунок 4.1 − Наглядное представление MVVM

Модель (Model) — это субъект, который описывает используемые в приложении данные. Это может быть что угодно, будь то обычный продукт или же сложный рецепт. Представление (View) — это графический элемент управления или набор элементов управления, отвечающих за отображение данных на экране. Модель представления (ViewModel) — это и есть та магия, стоящая за всем. Она содержит логику пользовательского интерфейса, команды, события и ссылку на модель. В MVVM модель преставления не обязана обновлять непосредственно данные представления. Благодаря мощному механизму привязки, представленному в WPF, пользовательский интерфейс может обновляться, как только обновляется модель преставления. Для этого, в модели преставления должен быть реализован интерфейс INotifyPropertyChanged и запустущенно событие OnPropertyChanged [4].

### **4.1.1** **Реализация MVVM**

Для работы с паттерном реализован класс наследующий интерфейс INotifyPropertyChanged. На рисунке 4.1 предоставлена реализация данного класса.

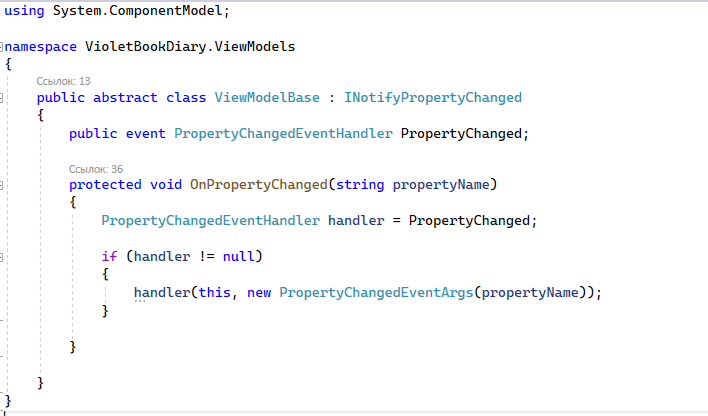


Рис. 4.1 – Реализация класса ViewModelBase

Данный класс реализует функцию OnPropertyChanged по вызове которой будут обновлять данные в представлении.

Для реализации модели представления, был реализован паттерн ICommand. Применение данного паттерна представлено на рисунке 4.2

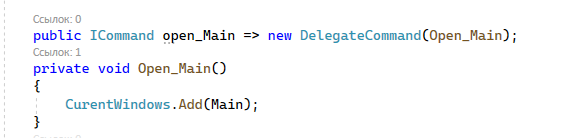


Рис. 4.2 – Пример использования ICommand

Объект типа ICommand привязывается к кнопке, что дает возможность вызова функции, находящейся в объекте.

### **4.1.2 Схема и описание авторизации**

Сразу после запуска программу пользователь попадает в окно авторизации. Если поля логина и пароля будут пустыми, в окне будет выведено сообщение об ошибке. После программа обращается к базе данных и проверяет существует ли пользователь с соответствующим логином и паролем. Если пользователь не найден, будет выведено сообщение об ошибке, в противном случае вход в приложение.

### **4.1.2 Создание современного дизайна в WPF**

Для упрощения создания дизайна интерфейса, была использована библиотека Material Design in Xaml. Данная библиотека автоматически подгружает большую часть дизайна и меняет его на более современный и интерактивный, добавляя визуальные эффекта, например нажатия на кнопку или открытие всплывающего списка. Так же библиотека была использована для реализации интуитивно понятного интерфейса, такой эффект был достигнут с помощью иконок отображающий суть кнопки.

Библиотека решает большую часть проблем, избавляя от огромного количества кода. Что позволяет при разработке приложения уделять больше внимания логической части, чем визуальной.

### **4.1.2 Чтение Fb2 файлов**

Изучив доступные реализации чтения файлов книг в WPF, я остановилась на формате Fb2 и пользовательской библиотеке FB2Library.

Данная библиотека упрощает чтение Fb2 файлов. В ней присутствует класс имеющий структуру дерева, в которому будет хранится весь Fb2 файл.

Применение данной библиотеки представлено на рисунке 4.3

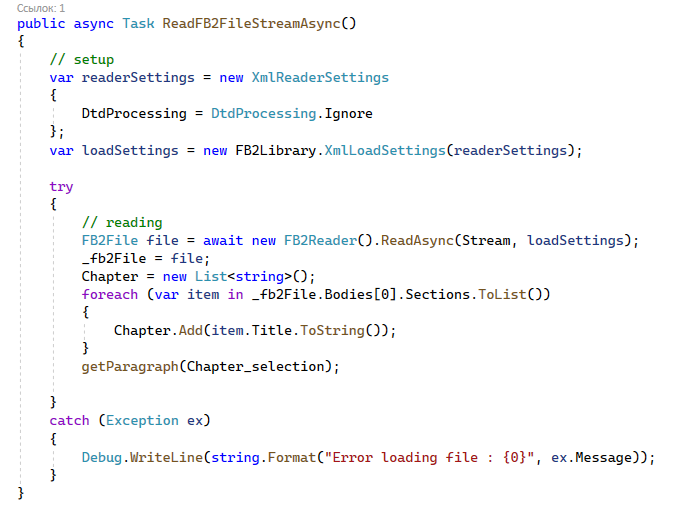


Рис. 4.3 – Функция чтения из файла

Данная функция была представлена в руководстве. Для моих задач она была модифицирована. В функцию добавлено создание списка глав и вывод текущей главы.

Функция вывода текущей главы – getParagreph с одним параметром. Параметр данной функции отражает, текущую главу, что дает возможность, быстрого перемещения по главам. А так же имеется функция запоминания последней прочтенной главы. Код данной функции предоставлен на рисунке 4.4



Рис. 4.4 – Функция чтения из файла

Функция getParagraph преобразует выбранную главу в XML формат. XML нужен для создание станицы с текстом главы. Он оборачивается в теги языка html для корректного вывода на экран. Подводя итоги, данная функция формирует html страницу с текстом выбранной главы.

### **4.1.4** **Подключение к БД**

Если различные механизмы работы с БД в С#. Т.к. моя база данных небольшая, мой выбор пал на ADO.Net EDM.

Entity Framework — это набор технологий в ADO.NET, которые поддерживают разработку программных приложений, ориентированных на данные. Архитекторам и разработчикам приложений, ориентированных на обработку данных, приходится учитывать необходимость достижения двух совершенно различных целей. Они должны моделировать сущности, связи и логику решаемых бизнес-задач, а также работать с ядрами СУБД, используемыми для сохранения и получения данных.

Данные могут распределяться по нескольким системам хранения данных, в каждой из которых применяются свои протоколы, но даже в приложениях, работающих с одной системой хранения данных, необходимо поддерживать баланс между требованиями системы хранения данных и требованиями написания эффективного и удобного для обслуживания кода приложения.

Платформа Entity Framework позволяет работать с данными в форме специфических для домена объектов и свойств (например, с клиентами и их адресами) без необходимости учитывать формат базовых таблиц и столбцов базы данных, где хранятся эти данные. Entity Framework дает разработчикам возможность работать с данными на более высоком уровне абстракции, создавать и сопровождать приложения, ориентированные на работу с данными, одновременно с этим сокращая объем кода по сравнению с традиционными приложениями. поскольку Entity Framework является компонентом платформа .NET Framework, Entity Framework приложения могут работать на любом компьютере, на котором установлена платформа .NET Framework с пакетом обновления 1 (SP1) версии 4,5.

# Описание интерфейсов программного средства, его тестирование и анализ полученных данных

## Описание интерфейсов программного средства

Сразу после запуска приложения перед пользователем открывается окно авторизации, в которое требуется ввести логин и пароль. Внешний вид окна представлен на рисунке 5.1.

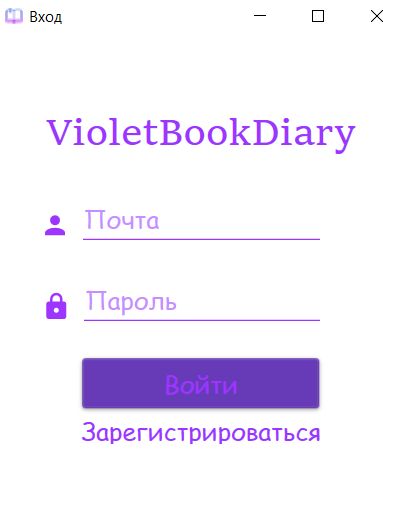


Рисунок 5.1 − Внешний вид окна авторизации

Из окна авторизации пользователи, которых нет в базе данных приложения, могут зарегистрироваться, нажав на кнопку «Регистрация». После того, как это произойдет, откроется окно регистрации новых пользователей, интерфейс которого представлен на рисунке 5.2.



Рисунок 5.2 − Интерфейс окна регистрации

После нажатия на кнопку регистрации, пользователь получает сообщение об результате регистрации, затем окно закрывается тем самым возвращая пользователя на окно авторизации.

### **5.1.1 Вход в качестве пользователя**

После успешной авторизации открывается главное окно приложения, которое встречает пользователя. На окне при входе открыта страница с базой книг и кнопкой переключения на страницу с базой картинок. Внешний вид главного окна и страницы базы книг предоставлена на рисунке 5.3 и 5.4.

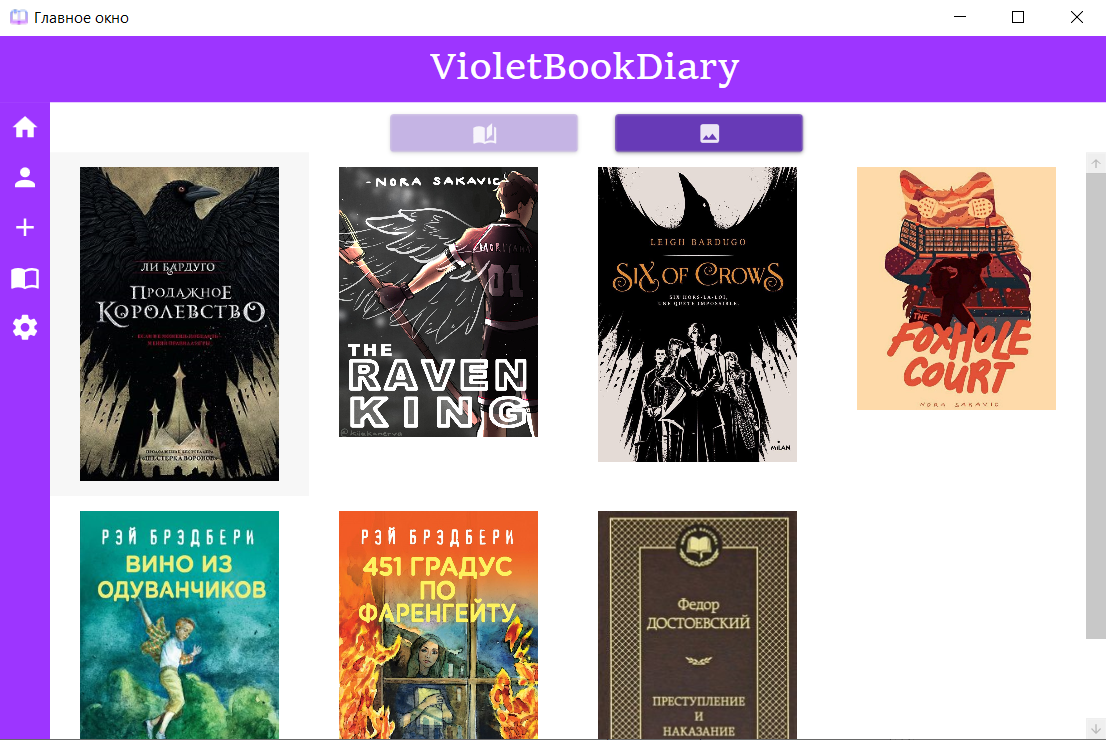


Рисунок 5.3 − Интерфейс главного окна и страницы базы книг



Рисунок 5.4 − Интерфейс главного окна и страницы базы картинок

Список книг является интерактивным, поэтому после нажатия на обложку книги, пользователь перейдет на страницу информации о книге. Интерфейс странице предоставлен на рисунке 5.5



Рисунок 5.5 − Интерфейс информации о книге

Так же на странице присутствуют пользовательские элементы управления, такие как «Отзывы о книге», интерфейс этого элемента управления предоставлен на рисунке 5.6

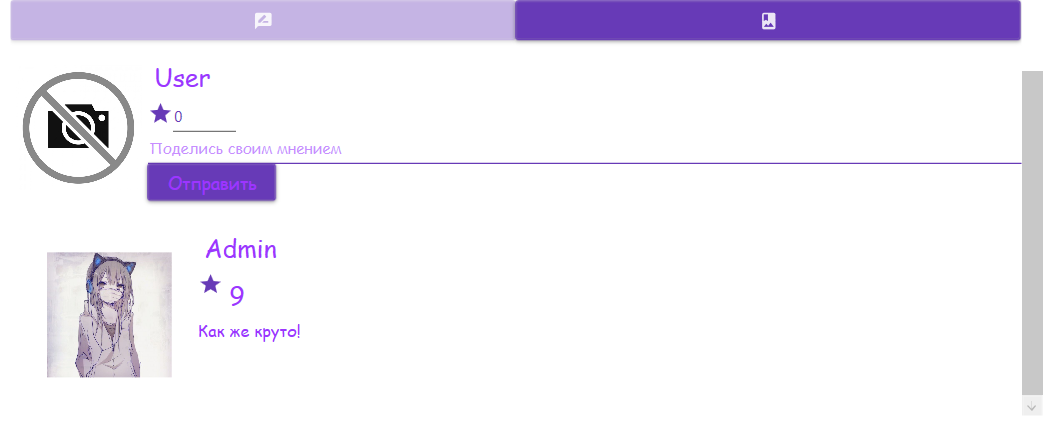


Рисунок 4.6 − Интерфейс элемента управления «Отзывы о книге»

Так же элемент управления «Картинки книги», предоставлен на рисунке 5.7.

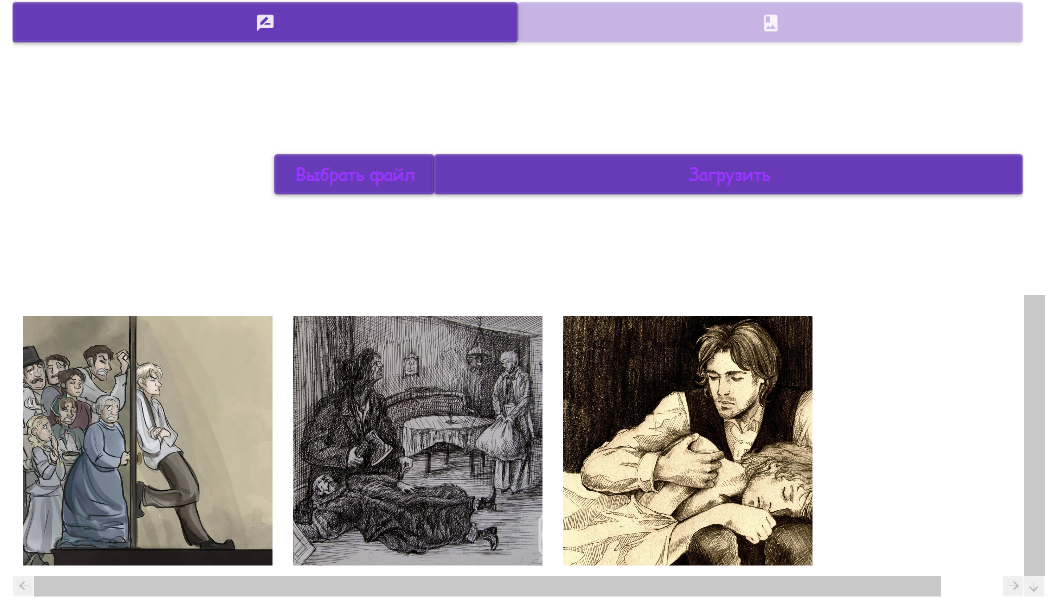


Рисунок 5.7 − Интерфейс элемента управления «Отзывы о книге»

Через элемент управления «Отзывы о книге» пользователь может перейти на страницу другого пользователя, нажав на фото профиля.

В верхнем левом углу присутствует кнопка назад по нажатию которой пользователь вернётся на предыдущую страницу.

Так же если в базе данных храниться файл книги, то будет разблокирована кнопка чтения и пользователь по нажатию сможет переместиться на страницу «Чтения книги», пример страницы предоставлен на рисунке 5.8 и имеет всё ту же кнопку назад и кнопку перемещения по главам, если таковые присутствуют.

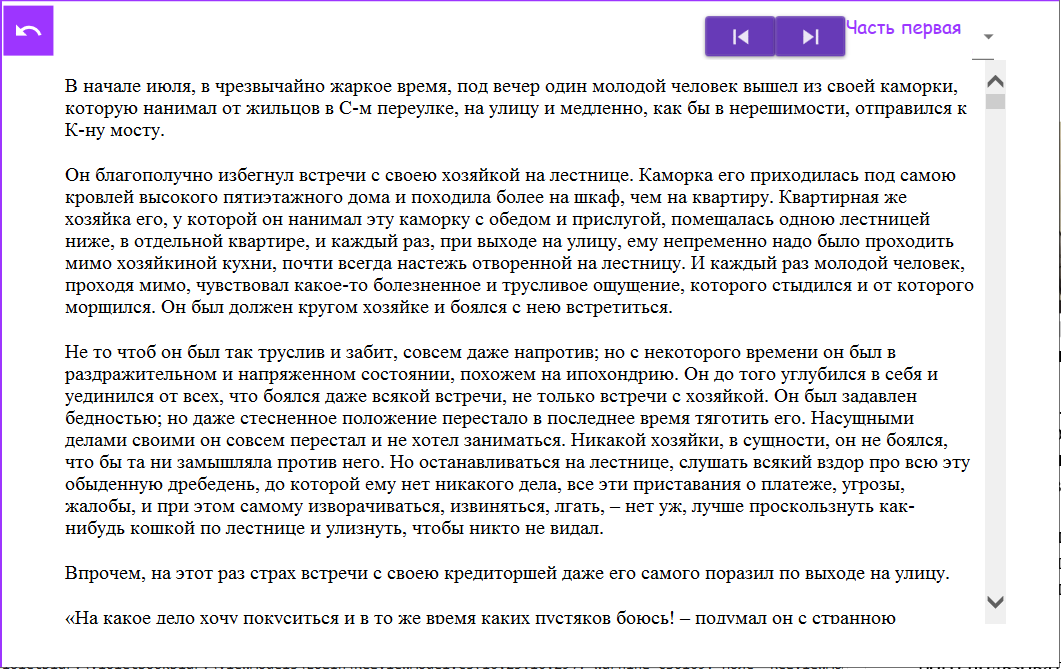


Рисунок 5.8 − Интерфейс страницы «Чтение книги»

Так же на главном окне находится интерактивное меню, с помощью которого пользователь может перемещать между страницами.

Первым после главной странице в меню находится кнопка перемещения на страницу «Информация о пользователе», интерфейс которой предоставлен на рисунке 5.9



Рисунок 5.9 − Интерфейс страницы «Информация о пользователе»

На странице описана вся доступная информация включая Id пользователя. Одной из частей информации о пользователе, является его библиотека, которая доступна и другим пользователя. Библиотека включает в себя кнопки перемещения по имеющимся спискам и сам список, он как и на других страницах является интерактивным. Так же присутствуют кнопки для изменения профиля и выхода из него. Окно изменения профиля предоставлено на рисунке 5.10

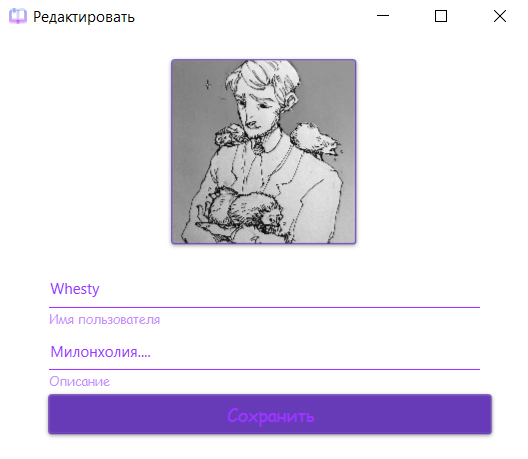


Рисунок 5.10 − Интерфейс окна «Изменения информации о пользователе»

Третья кнопка бокового меню открывает окно «Добавить книгу», где пользователь может заполнить форму, чтобы добавить книгу. Интерфейс окна предоставлен на рисунке 5.11

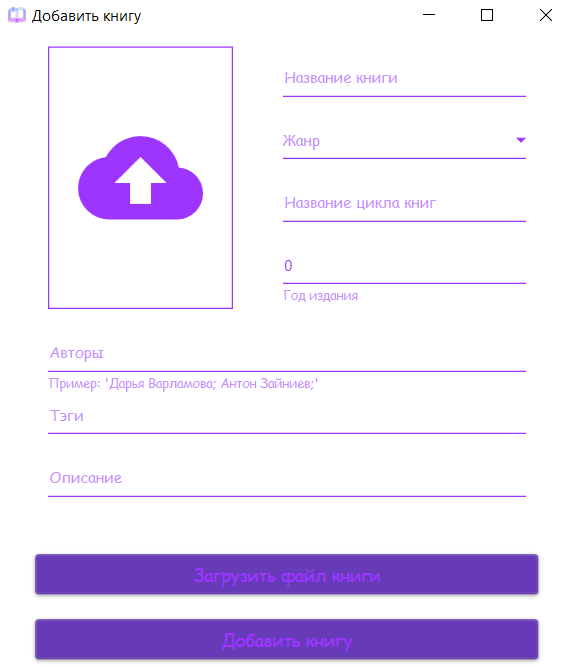


Рисунок 5.11 − Интерфейс окна «Добавить книгу»

Так же через боковое меню можно попасть на страницу «Каталога», в котором представлен список книг и возможность - поиска и фильтрация, по полям жанров, тэгов и авторов. Интерфейс страницы представлен на рисунке 5.12

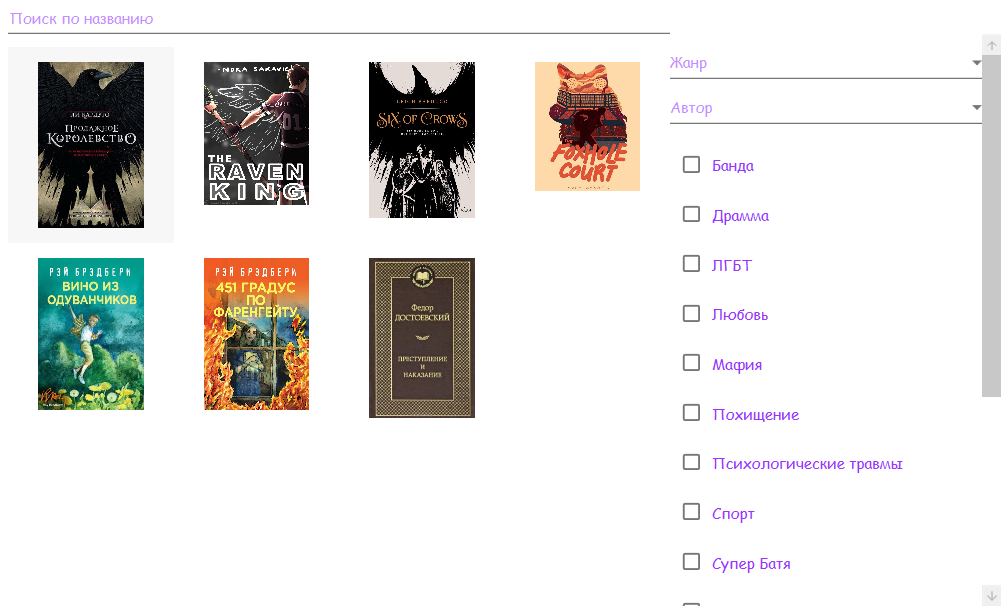


Рисунок 5.12 − Интерфейс страницы «Каталог»

Так же пользователю доступно окно настроек, где можно выбрать язык и изменить тему. Окно представлено на рисунке 5.13

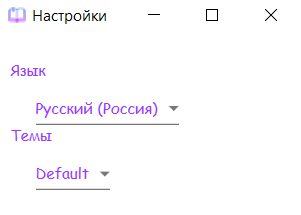


Рисунок 5.14 − Интерфейс окна «Настройки»

### **5.1.2 Вход в качестве администратора**

Единственным отличием от входа за пользователя, является страница редактирования базы книг, где администратор может подтвердить или удалить книгу предложенную пользователем. Интерфейс страницы представлен на рисунке 5.15



Рисунок 5.15 − Интерфейс страницы предложенных книг

## Тестирование программное средства и анализ полученных данных

Для проверки работоспособности программного средства было проведено его тщательное тестирование, в ходе которого осуществлялась проверка заполняемости базы данных, корректность извлекаемых из базы данных значений, а также реакция программы на вводимые, заведомо неверные, данные. Было проработано множество ситуаций, которые могут возникнуть в ходе работы с программным средством.

Для начала протестируем окно входа, при неправильном вводе данных, а именно отсутствием введенных данных в базе, возникает ошибка уведомляющая пользователя о том что он ввел неправильные данные (результат на рис 5.16):

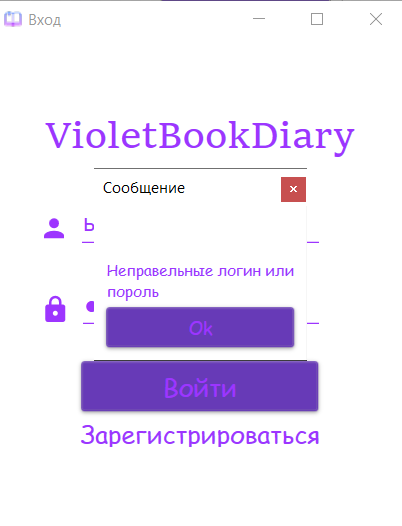


Рисунок 5.16 − Сообщение об ошибке авторизации

Далее было проверена форма для добавления книг в бд, если не ввести данные или ввести не все то откроется окно сообщения, пример ошибки предоставлен на рисунке 5.17

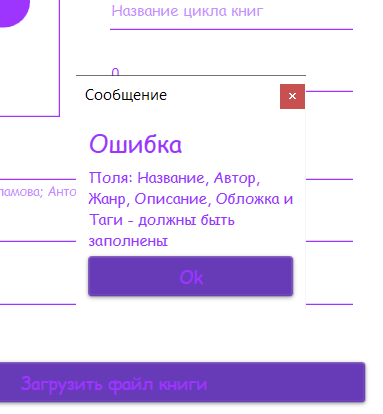


Рисунок 5.17 − Сообщение о некорректно заполненной форма

В данном окне так же присутствует загрузка изображения, поэтому при выборе файла обложки, если закрыть окна так ничего и не выбрав, появится сообщение об ошибке (рисунок 5.18)

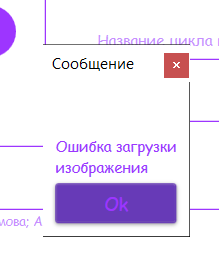


Рисунок 5.18 − Сообщение о некорректном выборе файла

Так же были проверены поля с валидацией, например поле оценки должно быть исключительно числом (результат предоставлен на рисунке 5.19)

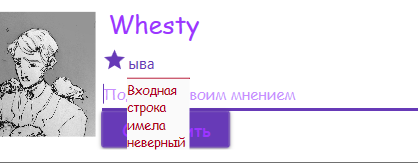


Рисунок 5.19 − Сообщение о неверных данных

В конце проверим поля регистрации, на рисунке 5.20 показано, что если поля не соблюдают свои условии, то они будут подсвечиваться красным и кнопка регистрации будет заблокирована.



Рисунок 5.20 − Сообщение о неверных данных

# Руководство по использованию

Первым делом новый пользователь должен зарегистрироваться в базе данных. Для этого при запуске программы пользователь должен нажать кнопку зарегистрироваться. В появившемся окне ввести данные для регистрации и нажать зарегистрироваться.

После регистрации пользователь должен авторизоваться, введя данные из только что зарегистрированного аккаунта. Затем нажать кнопку войти.

Авторизовавшись, пользователь попадет на главное окно, где ему будут предоставлены все возможные функции данного приложения.

Первым делом советую изменить данные в аккаунте, это можно сделать на странице «Информация о пользователе», нажав кнопку изменить. Окно изменения профиля предоставлено на рисунке 5.10.

Далее пользователь волен делать все, что ему заблагорассудится.

Что бы создать собственную библиотеку, нужно найти интересующую тебя книгу, сохранить в библиотеку, нажав на кнопку с сердечком, выбрать категорию своей библиотеки. Если же добавленная книга не понравилась или была добавлена случайно, книгу можно определить в категорию «Брошено». Книги, добавленные в библиотеку, можно увидеть на странице «Информация о пользователе».

Может случится так, что книги, которую пользователь хочет сохранить в свою библиотеку, может не оказаться. Тогда открыв окно «Добавить книгу» пользователь сможет предложить свою книгу. Когда администратор увидит и проверит заполненные поля, книга в случае корректных данных окажется в библиотеке.

Оставить отзыв или поделится картинкой, к книге можно, пролистнув страницу с информацией в низ. Там так же можно увидеть чужие отзывы.

Так же пользователь может настроить интерфейс под себя в окне настроек. Окно настроек имеет 2 выпадающих списка. С помощью которых можно изменить тему приложения и язык.

# Заключение

В ходе выполнения курсовой работы был разработан проект, предназначенный для больших любителей книг, которые готовы не только читать их, но и делиться своим мнением и рисунками. Во время выполнения данной курсовой работы было изучено немало теоретического материала, а также просмотрено и разобрано большое количество уже готовых решений тех или иных задач.

В ходе выполнения курсовой работы было выполнено следующее:

* добавлять/изменять/удалять данные из базы данных;
* получать статистику прочитанных книг;
* добавлять отзывы;
* изменять статус чтения;
* добавлять файлы/аудиофайлы содержащие книги;
* просматривать/добавлять арты;
* поиск книг по жанрам/тегам/название/авторам/циклам;
* просматривать/изменять книги в закладках;
* просматривать информацию о книгах/авторах;

В соответствии с полученным результатом работы программы можно сделать вывод, что разработанная программа работает, верно, а требования технического задания выполнены в полном объеме.

Само приложение – первый опыт в данной области. Естественно, существует ряд приложений, имеющих функционал и дизайн в разы лучше. Что дает возможность стремится к лучшему, прокачивая свои навыки в создании веб-приложений.

Подводя итоги, был получен огромный опыт, который в дальнейшей карьере поможет мне решать схожие проблемы возникшие при создании данного приложения.

## Список использованных источников

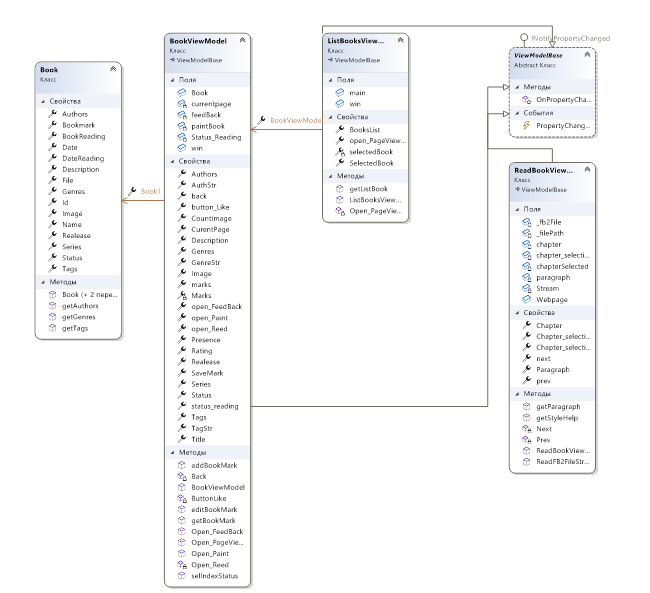
1. WPF | ListBox и ComboBoc(professorweb.ru) – Режим доступа: https://professorweb.ru/my/WPF/UI\_WPF/level6/6\_17.php?
2. Garofallo, R. Building Enterprise Applications with Windows Presentation Foundation and the Model View ViewModel Pattern / R. Garofallo − California: O’Reilly Media, − 2011 − 220 p.
3. Информационный сайт: базы данных [Электронный ресурс]. Режим доступа:[inform.sch117nn.edusite.ru/DswMedia/klyaks\_netuchitelyutestyibazyidannyix.htm](http://inform.sch117nn.edusite.ru/DswMedia/klyaks-_netuchitelyutestyibazyidannyix.htm) – Дата доступа: 24.05.2021.
4. Сайт о программировании Metanit: Введение в ADO.NET [Электронный ресурс]– Режим доступа: <https://metanit.com/sharp/adonet/1.1.php> – Дата доступа: 24.05.2021.
5. Репозиторий проекта библиотеки FB2Library(GitHub) – Режим доступа: https://github.com/wcoder/FB2Library
6. WCF RIA Services – Режим доступа: <https://habr.com/ru/post/203820/>

## Приложение А



Приложение А - UML-диаграммы вариантов использования

## Приложение Б



Приложение Б – Диаграмма классов