

Министерство образования и науки Республики Башкортостан
Государственное автономное профессиональное образовательное учреждение
Уфимский колледж статистики, информатики и
вычислительной техники

УТВЕРЖДАЮ
Заместитель директора
по учебной работе
_____ 3.3. Курмашева
«___» _____ 2023 г.

РАЗРАБОТКА ВЕБ-ПРИЛОЖЕНИЯ ДЛЯ ВЕДЕНИЯ ЧИТАТЕЛЬСКОГО
ДНЕВНИКА

Пояснительная записка к дипломному проекту

Рецензент
_____ Н.К.Валеева
«___» _____ 2023 г.

Руководитель
_____ Р.Ф.Каримова
«___» _____ 2023 г.

Выпускник гр. 19П-1
_____ А.Г.Аминов
«___» _____ 2023 г.

2023

Министерство образования и науки Республики Башкортостан
Государственное автономное профессиональное образовательное учреждение
Уфимский колледж статистики, информатики и
вычислительной техники

УТВЕРЖДАЮ

Заместитель директора
по учебной работе

_____ 3.3. Курмашева
«__» _____ 2023 г.

ЗАДАНИЕ

на дипломный проект студенту дневного отделения, группы 19П-1,
специальности 09.02.07 Информационные системы и программирование.

Фамилия, имя, отчество: Аминов Арслан Гайнетдинович.

Тема дипломного проекта: «Разработка веб-приложения для ведения
читательского дневника».

Текст задания:

при выполнении дипломного проекта должны быть решены следующие задачи:

- а) изучен и выполнен системный анализ предметной области;
- б) разработаны структура и дизайн веб-приложения;
- в) спроектирована база данных;
- г) реализованы функции для читателя: регистрация, авторизация, редактирование своего профиля, поиск книг и авторов, просмотр дополнительной информации о книгах и авторах, написание рецензий на книги, оценивание рецензий, добавление книги в «Избранное», ведение читательского дневника с заметками;

д) реализованы функции для администратора: добавление, редактирование и удаление данных о книгах, авторах, жанрах и модерирование рецензий пользователей на книги.

В результате выполнения дипломного проекта должны быть представлены:

а) пояснительная записка, состоящая из следующих разделов:

Введение

1 Постановка задачи

2 Экспериментальный раздел

3 Экономический раздел

Заключение

Приложения

Список сокращений

Список использованных источников

а) электронный носитель, содержащий разработанный программный продукт;

б) графическая часть – 3 листа в формате А4;

в) презентация дипломного проекта в электронном виде.

Список рекомендуемых источников:

1 Кумскова И.А. Базы данных: учебник / Кумскова И.А. — Москва : КноРус, 2021. — 400 с. — ISBN 978-5-406-08303-1. — URL: <https://book.ru/book/940108>. — Текст: электронный.

2 Федорова Г.Н. Разработка модулей программного обеспечения для компьютерных систем : учебник для студ. учреждений сред. проф. образования / Г.Н. Федорова. — 4-е изд., перераб. — Москва : Издательский центр «Академия», 2020. — 384 с. ISBN 978-5-4468-9443-7. - Текст : электронный. - URL: <https://academia-moscow.ru/reader/?id=473265#> – Режим доступа: по подписке

3 METANIT.COM C#: информационная система: сайт, 2023 – URL:
<https://metanit.com/sharp/tutorial/> – Режим доступа: свободный. – Текст:
электронный

Задание к выполнению получил «30» марта 2023 г.

Студент _____ Аминов Арслан Гайнетдинович

Срок окончания «08» июня 2023 г.

Руководитель дипломного проекта _____ Р.Ф. Каримова

Задание рассмотрено на заседании цикловой комиссии информатики

«25» марта 2023 г. протокол №5

Председатель цикловой комиссии информатики _____ О.В. Фатхулова

Министерство образования и науки Республики Башкортостан
Государственное автономное профессиональное образовательное учреждение
Уфимский колледж статистики, информатики и
вычислительной техники

ЗАКЛЮЧЕНИЕ

на дипломный проект

Дипломник Аминов Арслан Гайнетдинович

Группа 19П-1

Специальность 09.02.07 Информационные системы и программирование

Тема Разработка веб-приложения для ведения читательского дневника

Объем дипломного проекта:

количество листов пояснительной записки _____

количество листов графической части _____

Заключение о степени соответствия заданию на дипломный проект

Характеристика качеств, проявленных студентом при работе над проектом:
самостоятельность, дисциплинированность, умение планировать работу и
пользоваться литературным материалом и т.д.

Положительные стороны

Недостатки

Характеристика общетехнической и специальной подготовки выпускника

Заключение и предлагаемая оценка за дипломный проект

Руководитель Каримова Резеда Флюоновна

Должность преподаватель

Место работы ГАПОУ Уфимский колледж статистики, информатики и
вычислительной техники

« » 2023 г.

Подпись

Министерство образования и науки Республики Башкортостан
Государственное автономное профессиональное образовательное учреждение
Уфимский колледж статистики, информатики и
вычислительной техники

РЕЦЕНЗИЯ

на выпускную квалификационную работу

Дипломник Аминов Арслан Гайнетдинович

Группа 19П-1

Специальность 09.02.07 Информационные системы и программирование

Тема Разработка веб-приложения для ведения читательского дневника

Объем дипломного проекта:

количество листов пояснительной записки _____

количество листов графической части _____

Заключение о степени соответствия заданию на дипломный проект

Характеристика выполнения каждого раздела дипломного проекта

Перечень положительных качеств дипломного проекта, возможность его
использования на производстве

Недостатки

Оценка качества выполнения графической части дипломного проекта

Оценка качества выполнения пояснительной записки дипломного проекта

Оценка общеобразовательной и технической подготовки выпускника

Отзыв о дипломном проекте в целом, предлагаемая оценка

Рецензент

Должность

Место работы

«__» _____ 2023 г.

Подпись

АННОТАЦИЯ

Пояснительная записка к дипломному проекту содержит постановку и программу решения задачи «Разработка веб-приложения для ведения читательского дневника».

Веб-приложение Lib написано на языках HTML, CSS, C# на основе ASP.NET MVC с использованием системы управления базами данных MySQL, предназначено для работы в браузере на любой платформе, отлажено на данных контрольного примера.

					40.A-2075-2023 09.02.07 ДП-ПЗ		
Изм.	Лист	№ докум.	Подпись	Дата	Разработка веб-приложения для ведения читательского дневника		
Разраб.		Аминов А.Г.					
Провер.		Каримова Р.Ф.					
Реценз.		Валеева Н.К.					
Н. Контр.		Каримова Р.Ф.					
Утверд.		Курмашева З.З.					
					Лит.	Лист	Листов
						2	174
					УКСИВТ 19П-1		

СОДЕРЖАНИЕ

	лист
Введение	4
1 Постановка задачи	6
1.1 Описание предметной области	6
1.2 Диаграмма прецедентов	9
1.3 Описание входной информации	9
1.4 Описание выходной информации	10
1.5 Общие требования к программному продукту	10
1.6 Описание структуры базы данных	11
1.7 Контрольный пример	15
2 Экспериментальный раздел	16
2.1 Описание программы	16
2.2 Протокол тестирования программного продукта	20
2.3 Руководство пользователя	28
3. Экономический раздел	42
3.1 Расчет затрат на создание программного продукта	42
3.2 Расчет цены предложения	44
Заключение	47
Приложение А. Контрольный пример	48
Приложение Б. Исходный код	70
Приложение В. Результат работы веб приложения	169
Список сокращений	171
Список использованных источников	172

ВВЕДЕНИЕ

Рецензия на книгу — это текст, в котором рецензент (человек, написавший рецензию) описывает свои впечатления от прочитанной им книги. Рецензент обычно анализирует сюжет, персонажей, стиль письма и другие аспекты книги, а также выражает свое мнение о ней.

Рецензии на книги могут быть полезными:

- определить, подходит ли книга их вкусам и интересам;
- содержанием объективной информации о книге, такую как стиль письма, сюжет и персонажи;
- помочь читателям определить, насколько хорошо написана книга и насколько она интересна;
- содержать рекомендации от других читателей, которые могут быть полезными при выборе книги;
- создать сообщество читателей, которые делятся своими мыслями и впечатлениями о книгах.

Рецензии могут быть опубликованы, как в журналах, газетах, так и на сайтах. Сайт с рецензиями на книги остается актуальным в наше время, так как очень много людей интересуются чтением. Благодаря такому сайту пользователи могут удобно получить информацию о книгах, которые им интересны, и принять решение о покупке или чтении. Кроме того, сайты с рецензиями на книги часто предлагают обзоры новых книг и рекомендации от других читателей, что может помочь пользователям найти новые интересные авторов и жанры.

Целью дипломного проекта является предоставление возможности ведения читательского дневника, просмотр и написание рецензий на книги.

Для достижения поставленной цели необходимо решить следующие задачи:

- изучить предметную область;
- спроектировать базу данных;
- разработать структуру и дизайн веб-приложения;

					40.А-2075-2023 09.02.07 ДП-ПЗ	Лист
						4
Изм.	Лист	№ докум.	Подпись	Дата		

– реализовать функции для читателя: регистрация, авторизация, редактирование своего профиля, поиск книг и авторов, просмотр дополнительной информации о книгах и авторах, написание рецензий на книги, оценивание рецензий, добавление книги в «Избранное», ведение читательского дневника с заметками;

– реализовать функции для администратора: изменение информационных справочников и снятие с публикации рецензии пользователей на книги;

– разработать и протестировать веб-приложение.

Аналогами веб-приложения являются:

– «LiveLib» - книжный рекомендательный сервис, обладает крупнейшей базой пользовательских рецензий, освещает литературные новости и цены на книжные новинки, позволяет зарегистрированным участникам вести учёт и анализ прочитанных произведений.

– «Книгогид» - рекомендательный сервис, предоставляющий персональные рекомендации в самых различных областях литературы – художественных, технических и научных. Можно публиковать рецензии, а еще составлять подборки книг, делиться любимыми отрывками и вести читательский дневник.

– «Goodreads» - сайт предоставляет свободный доступ к обширной базе данных книг, аннотаций, различных обзоров. Каждый пользователь может свободно зарегистрироваться сам, зарегистрировать книги, которые будут включены в библиотечный каталог и в перечень рекомендуемой литературы. У зарегистрированных пользователей есть возможность создания своих групп по интересам, обращаясь к Книге предложений и обсуждений.

					40.А-2075-2023 09.02.07 ДП-ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		5

1 Постановка задачи

1.1 Описание предметной области

Требуется разработать информационную систему для ведения читательского дневника с возможностью написания рецензий на книги. Поиск книг должен осуществляться по форме поиска по названию и жанру, в различных подборках или на страницах об авторах и жанрах. Должна быть возможность добавления книги себе в «Избранное».

В базе данных должны храниться следующие справочники: жанры, роли пользователя, пользователи, авторы, книги, избранные книги, рецензии, лайки, заметки.

В таблице «Жанр» содержится следующая информация:

- идентификатор жанра;
- название жанра.

В таблице «Роль пользователя» содержится следующая информация:

- идентификатор роли;
- название роли.

Для того, чтобы пользоваться функционалом программы, клиент должен зарегистрироваться в системе, заполнив свои данные. В таблице «Пользователь» содержится следующая информация:

- идентификатор пользователя;
- логин;
- пароль;
- имя;
- дата регистрации;
- роль.

В таблице «Книга» содержится следующая информация:

- идентификатор книги;
- название;

					40.А-2075-2023 09.02.07 ДП-ПЗ	Лист
						6
Изм.	Лист	№ докум.	Подпись	Дата		

- описание;
- фотография;
- средняя оценка (высчитывается автоматически из оценок рецензий);
- дата написания.

В таблице «Автор» содержится следующая информация:

- идентификатор автора;
- имя;
- фотография;
- биография.

Книга может относиться к нескольким жанрам. В таблице «Жанр – книга» содержится следующая информация:

- идентификатор связи;
- идентификатор жанра;
- идентификатор книги.

У книги может быть несколько авторов. В таблице «Автор – книга» содержится следующая информация:

- идентификатор связи;
- идентификатор автора;
- идентификатор книги.

В таблице «Избранная книга» содержится следующая информация:

- идентификатор;
- идентификатор пользователя;
- идентификатор книги;
- дата добавления;
- метка (например «Прочитано»).

В таблице «Рецензия» содержится следующая информация:

- идентификатор;
- идентификатор пользователя;

					40.А-2075-2023 09.02.07 ДП-ПЗ	Лист
						7
Изм.	Лист	№ докум.	Подпись	Дата		

- идентификатор книги;
- дата создания;
- оценка;
- контент.

Пользователь может оценить рецензию (поставить лайк). В таблице «Лайк» содержится следующая информация:

- идентификатор;
- идентификатор рецензии;
- идентификатор пользователя;
- дата добавления.

В таблице «Заметка» содержится следующая информация:

- идентификатор;
- идентификатор пользователя;
- дата создания;
- название;
- контент.

Данная информационная система предполагает наличие двух групп пользователей: читатель, администратор.

Читатель должен иметь возможности:

- регистрироваться и авторизовываться на сайте;
- изменять информацию профиля;
- пользоваться поиском книг по библиотеке;
- просматривать дополнительную информацию о книгах;
- просматривать дополнительную информацию об авторах;
- просматривать рецензии на книги, писать их самому;
- оценивать рецензии, просматривать список оценивших рецензию.
- добавлять книги себе в «Избранное»;
- вести читательский дневник с заметками;

					40.А-2075-2023 09.02.07 ДП-ПЗ	Лист
						8
Изм.	Лист	№ докум.	Подпись	Дата		

Администратор, помимо возможностей читателя, также должен иметь возможности:

- работать с информационными справочниками;
- снимать с публикации отзывы пользователей на книги.

В процессе работы системы выходной информацией будут являться:

- список популярных книг;
- список книг с самыми высокими оценками;
- список книг пользователя.

1.2 Диаграмма прецедентов

Диаграмма прецедентов (англ. use case diagram) — диаграмма, отражающая отношения между актерами и прецедентами и являющаяся составной частью модели прецедентов, позволяющей описать систему на концептуальном уровне. Диаграмма прецедентов представлена на рисунке 1.2.1.

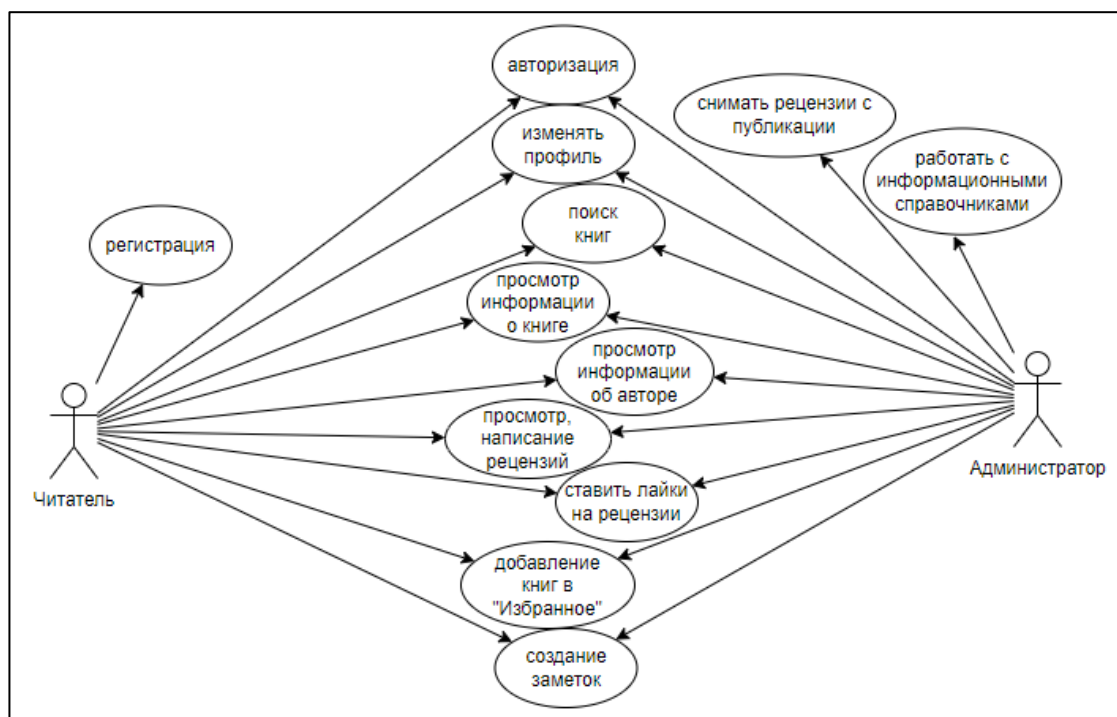


Рисунок 1.2.1 – Диаграмма прецедентов

1.3 Описание входной информации

В качестве входной информации используются следующие данные:

- жанры;

- авторы;
- книги;
- пользователи;
- рецензии пользователей;
- лайки пользователей на рецензии;
- книги в «Избранном» пользователей;
- заметки пользователей.

1.4 Описание выходной информации

В процессе работы системы выходной информацией будут являться: список популярных книг, список книг с самыми высокими оценками, список книг пользователя. Описание выходных документов отображено в таблице 1.4.1.

Таблица 1.4.1 — Описание выходной информации

Наименование (шифр)	Периодичность выдачи	Кол-во экз.
Список популярных книг	По мере необходимости	1
Список книг с самыми высокими оценками	По мере необходимости	1
Список книг пользователя	По мере необходимости	1

1.5 Общие требования к программному продукту

Общее наименование продукта «Lib». В процессе эксплуатации с веб-приложением работают читатель, администратор, которые должны обладать базовыми навыками работы с ПК и браузером.

Веб-приложение должно соответствовать следующим требованиям:

- удобный и визуально привлекательный интерфейс;
- адаптивный дизайн для совместимости на разных устройствах;
- надежные меры безопасности защиты пользовательских данных;
- оптимальная производительность и масштабируемость;
- интеграция с другими системами и сервисами;
- отказоустойчивость в непредвиденных ситуациях;

– соответствие законодательным и нормативным требованиям.

Для эксплуатации веб-приложения «Lib» необходимо устройство с доступом к сети Интернет и браузером поддерживающий технологии HTML5, CSS3, JavaScript.

В состав технических средств должен входить IBM-совместимый персональный компьютер, включающий в себя процессор, оперативную память, видеокарту, монитор, мышь, клавиатура.

1.6 Описание структуры базы данных

При проектировании базы данных использовалась СУБД MySQL. Описание структуры базы данных приведено в таблицах 1.6.1 – 1.6.12.

Таблица 1.6.1 – author (автор)

Имя поля	Описание поля	Тип данных	Размер поля	Тип ключа (PK–первичный, FK– внешний)
id	ID автора	INT	4	PK
name	Название	VARCHAR	256	
photo	Фото	VARCHAR	512	
biography	Биография	VARCHAR	4096	

Таблица 1.6.2 – author_book (автор-книга)

Имя поля	Описание поля	Тип данных	Размер поля	Тип ключа (PK–первичный, FK– внешний)
id	ID связи	INT	4	PK
author_id	ID автора	INT	4	FK
book_id	ID книги	INT	4	FK

Таблица 1.6.3 – book (книга)

Имя поля	Описание поля	Тип данных	Размер поля	Тип ключа (PK–первичный, FK– внешний)
id	ID книги	INT	4	PK
name	Название	VARCHAR	256	
description	Описание	VARCHAR	256	
photo	Фото	VARCHAR	256	
avg_rating	Средний рейтинг	DOUBLE	8	
date_of_creation	Дата написания	VARCHAR	128	

Таблица 1.6.4 – featured_book (избранная книга)

Имя поля	Описание поля	Тип данных	Размер поля	Тип ключа (PK–первичный, FK– внешний)
id	ID связи	INT	4	PK
user_id	ID пользователя	INT	4	FK
book_id	ID книги	INT	4	FK
date_of_add	Дата добавления	DATETIME	8	
mark_id	ID метки	INT	4	FK

Таблица 1.6.5 - genre (список жанров)

Имя поля	Описание поля	Тип данных	Размер поля	Тип ключа (PK–первичный, FK– внешний)
id	ID жанра	INT	4	PK
name	Название	VARCHAR	256	

Таблица 1.6.8 – mark (метка)

Имя поля	Описание поля	Тип данных	Размер поля	Тип ключа (PK–первичный, FK– внешний)
id	ID метки	INT	4	PK
name	Название	VARCHAR	128	

Таблица 1.6.6 – genre_book (жанр-книга)

Имя поля	Описание поля	Тип данных	Размер поля	Тип ключа (PK–первичный, FK– внешний)
id	ID связи	INT	4	PK
genre_id	ID жанра	INT	4	FK
book_id	ID книги	INT	4	FK

Таблица 1.6.7 – like (лайк)

Имя поля	Описание поля	Тип данных	Размер поля	Тип ключа (PK–первичный, FK– внешний)
id	ID лайка	INT	4	PK
review_id	ID рецензии	INT	4	FK
user_id	ID пользователя	INT	4	FK
date	Дата добавления	DATETIME	8	

Таблица 1.6.9 – note (заметка)

Имя поля	Описание поля	Тип данных	Размер поля	Тип ключа (PK–первичный, FK– внешний)
id	ID заметки	INT	4	PK
user_id	ID пользователя	INT	4	FK
date_of_creation	Дата создания	DATETIME	8	
name	Название	VARCHAR	256	
content	Контент	VARCHAR	4096	

Таблица 1.6.11 – role_user (роль пользователя)

Имя поля	Описание поля	Тип данных	Размер поля	Тип ключа (PK–первичный, FK– внешний)
id	ID роли	INT	4	PK
name	Название	VARCHAR	128	

Таблица 1.6.10 – review (рецензия)

Имя поля	Описание поля	Тип данных	Размер поля	Тип ключа (PK–первичный, FK– внешний)
id	ID рецензии	INT	4	PK
user_id	ID пользователя	INT	4	FK
book_id	ID книги	INT	4	FK
date_of_creation	Дата создания	DATETIME	8	
rating	Оценка	INT	4	
content	Контент	VARCHAR	2048	

Таблица 1.6.12 – user (пользователь)

Имя поля	Описание поля	Тип данных	Размер поля	Тип ключа (PK–первичный, FK– внешний)
id	ID пользователя	INT	4	PK
login	Логин	VARCHAR	256	
password	Пароль	VARCHAR	256	
name	Имя	VARCHAR	256	
date_of_registration	Дата регистрации	DATETIME	8	
role_id	ID роли	INT	4	

Схема данных представлена на рисунке 1.6.1.

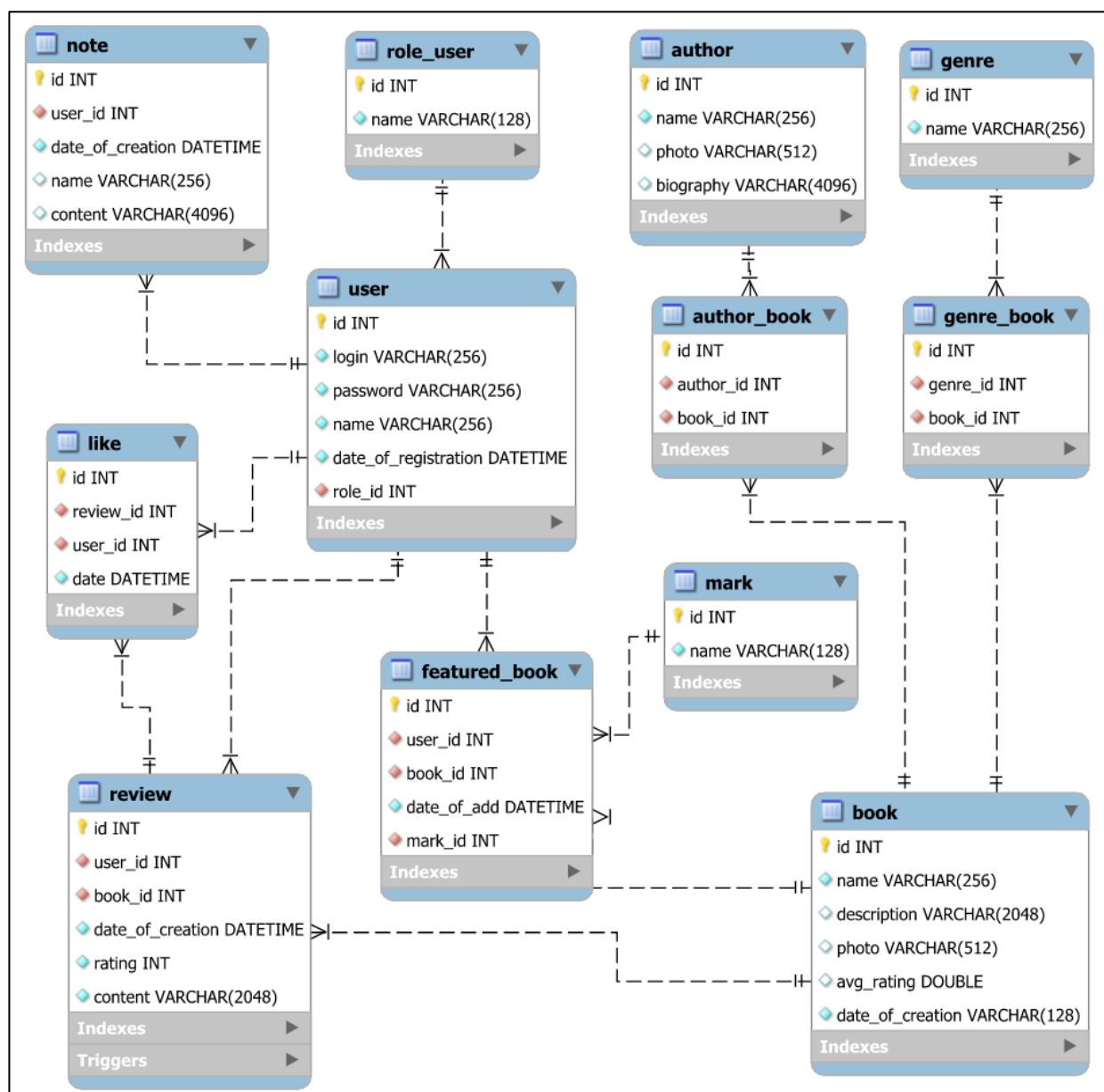


Рисунок 1.6.1 – Схема данных

1.7 Контрольный пример

Контрольный пример является ручным подсчетом задачи. Он представляет собой вариант задачи с исходными данными и используется для проверки правильности решения на ЭВМ. Входные данные для контрольного примера представлены в приложении А таблицах А.1 – А.12. Выходные данные контрольного примера представлены в рисунках В.1 – В.3 приложения В.

2 Экспериментальный раздел

2.1 Описание программы

Разработка информационной системы на основе ASP.NET MVC и MySQL предполагает использование комбинации языков программирования HTML, CSS, C#, фреймворков и технологий для создания веб-приложения, которое может хранить данные в базе данных MySQL и управлять ими.

ASP.NET MVC это веб-фреймворк, который используется для создания динамических веб-приложений и веб-сайтов, в то время как MySQL является популярной системой управления реляционными базами данных с открытым исходным кодом.

MVC расшифровывается как «модель-представление-контроллер» (от англ. model-view-controller). Это способ организации кода, который предполагает выделение блоков, отвечающих за решение разных задач. Один блок отвечает за данные приложения, другой отвечает за внешний вид, а третий контролирует работу приложения.

Физическая структура представлена на рисунке 2.1.1. Исходный код представлен в приложении Б.

					40.А-2075-2023 09.02.07 ДП-ПЗ	Лист
						16
Изм.	Лист	№ докум.	Подпись	Дата		

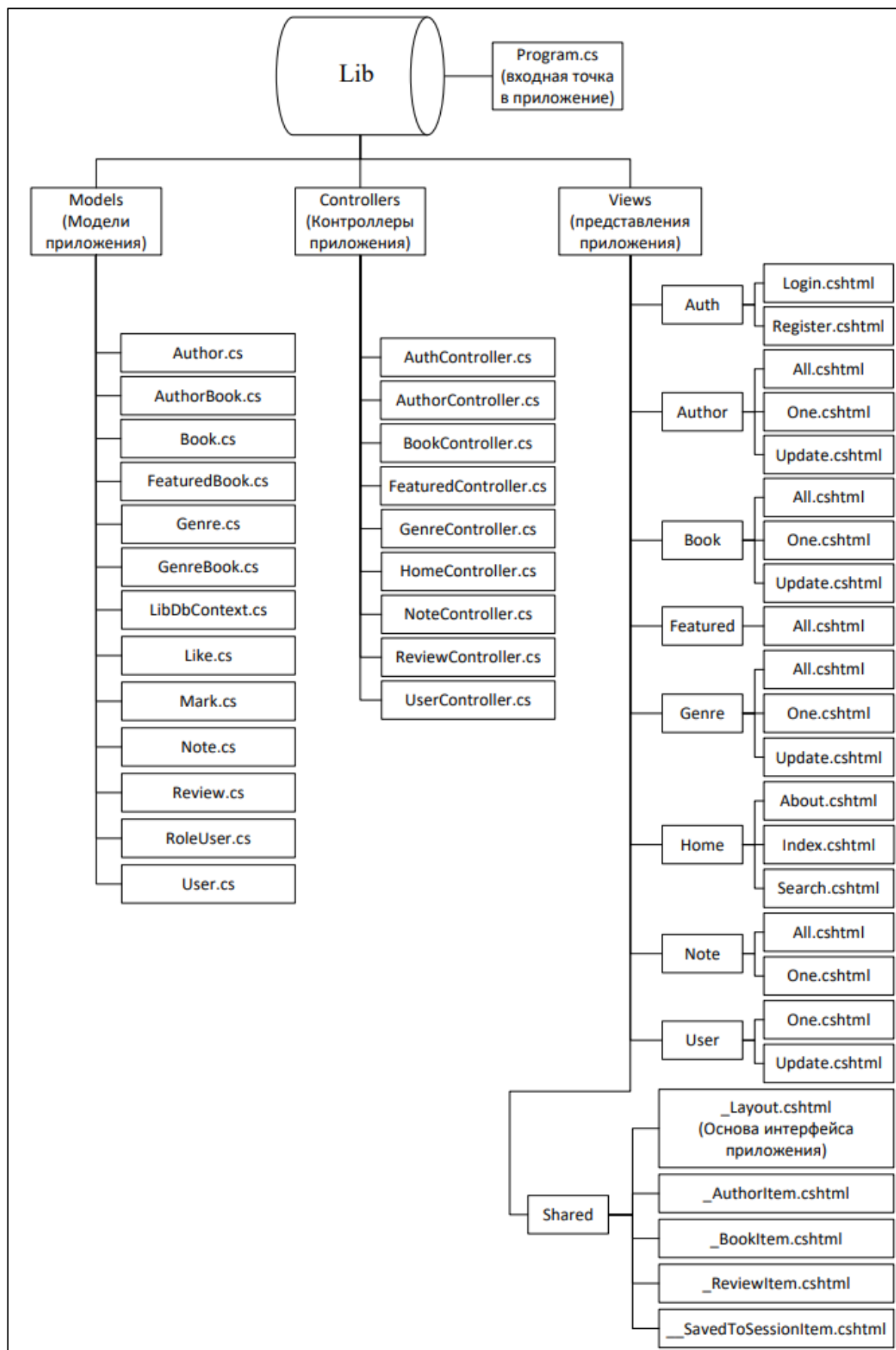


Рисунок 2.1.1 – Физическая структура

Описание структуры сайта представлена в таблице 2.1.1.

Таблица 2.1.1 - Описание структуры сайта

Название	Описание
Program.cs	Определяет входную точку в приложение
Models	Содержит модели сущностей БД
Models/Author.cs	Сущность «автор»
Models/AuthorBook.cs	Сущность «автор-книга»
Models/Book.cs	Сущность «книга»
Models/FeaturedBook.cs	Сущность «избранная книга»
Models/Genre.cs	Сущность «жанр»
Models/GenreBook.cs	Сущность «жанр-книга»
Models/LibDbContext.cs	Подключение к БД
Models/Like.cs	Сущность «лайк»
Models/Mark.cs	Сущность «метка»
Models/Note.cs	Сущность «заметка»
Models/Review.cs	Сущность «рецензия»
Models/RoleUser.cs	Сущность «роль пользователя»
Models/User.cs	Сущность «пользователь»
Controllers	Содержит контроллеры, методы которых содержат определенную логику и перенаправляют на соответствующую страницу
Controllers/AuthController.cs	Обработка запросов авторизации/регистрации
Controllers/AuthorController.cs	Обработка запросов вывода всех авторов или конкретного, с выводом книг автора
Controllers/BookController.cs	Обработка запросов вывода всех книг, или конкретного, с подробной информацией
Controllers/FeaturedController.cs	Обработка запросов для избранных книг
Controllers/GenreController.cs	Обработка запросов вывода всех жанров или конкретного, с выводом соответствующих книг
Controllers/HomeController.cs	Обработка запросов вывода основных страниц
Controllers/NoteController.cs	Обработка запросов для страниц с заметками
Controllers/ReviewController.cs	Обработка запросов для рецензий

Продолжение таблицы 2.1.1

Controllers/UserController.cs	Обработка запросов, связанных с профилем пользователя
Views	Содержит интерфейсы тела страницы приложения
Views/Auth/Login.cshtml	Форма авторизации
Views/Auth/Register.cshtml	Форма регистрации
Views/Author/All.cshtml	Список всех авторов по алфавиту
Views/Author/One.cshtml	Страница автора со списком его книг
Views/Author/Update.cshtml	Форма создания/редактирования автора
Views/Book/All.cshtml	Список книг
Views/Book/One.cshtml	Страница автора со списком его книг
Views/Book/Update.cshtml	Форма создания/редактирования книги
Views/Featured/All.cshtml	Список книг пользователя в «Избранном»
Views/Genre/All.cshtml	Список жанров
Views/Genre/One.cshtml	Список книг по жанру
Views/Genre/Update.cshtml	Форма создания/редактирования жанра
Views/Home/About.cshtml	Страница «О сайте»
Views/Home/Index.cshtml	Главная страница сайта
Views/Home/Search.cshtml	Форма поиска книг и авторов
Views/Note/All.cshtml	Список заметок пользователя
Views/Note/One.cshtml	Страница заметки
Views/User/One.cshtml	Профиль пользователя
Views/User/Update.cshtml	Форма редактирования пользователя
Views/Shared/_Layout.cshtml	Основа интерфейса приложения
Views/Shared/_AuthorItem.cshtml	Элемент списка с информацией об авторе
Views/Shared/_BookItem.cshtml	Элемент списка с информацией о книге
Views/Shared/_ReviewItem.cshtml	Элемент списка с информацией о рецензии
Views/Shared/_SavedToSessionItem.cshtml	Список книг и авторов, сохраненных в истории просмотра

2.2 Протокол тестирования программного продукта

В ходе тестирования веб-приложения на корректных и некорректных данных не было обнаружено ошибок, которые влияли бы на работу самого веб-приложения и всей системы.

Данное веб-приложение удовлетворяет всем предъявленным требованиям, имеет комфортный интерфейс и интуитивно понятный функционал, исключает появления системных ошибок.

В таблице 2.2.1 представлена общая информация о тестировании. В таблицах 2.2.2 – 2.2.7 представлены протоколы тестирования: тестирование регистрации на корректных/некорректных данных, тестирование авторизации на корректных данных, тестирование сохранения заметки на корректных данных, тестирование написания рецензии на корректных/некорректных данных.

Таблица 2.2.1 – Общая информация о тестировании

Название проекта	Lib
Номер версии	1.9
Имя тестера	Аминов Арслан Гайнетдинович
Даты тестирования	1.06.2023

Таблица 2.2.2 – Протокол тестирования регистрации на корректных данных

Описание информационных полей для тестирования	
1	2
Наименование	Описание
Test Case #	Lib_test_1
Приоритет тестирования	Высокий
Название тестирования	Регистрация
Резюме испытания	Необходимо добиться корректного поведения веб-приложения при регистрации пользователя.
Шаги тестирования	Заполнить поля, необходимыми для регистрации, корректными данными.

Продолжение таблицы 2.2.2

1	2
Данные тестирования	Login: neverket2@gmail.com; Password: 123456; Confirm password: 123456; Name: Aminov.
Ожидаемый результат	Веб-приложение должно открыть главную страницу с приветствием.
Фактический результат	Веб-приложение открывает главную страницу с приветствием.
Предпосылки	На шапке сайта нажата кнопка «Регистрация».
Постусловия	Система не зависает, находится в состоянии полной работоспособности.
Статус (Pass/Fail)	Pass

Проведем тестирование регистрации на корректных данных. Результаты изображены на рисунках 2.2.1 – 2.2.2.

Регистрация

Рисунок 2.2.1 – Форма регистрации с корректными данными

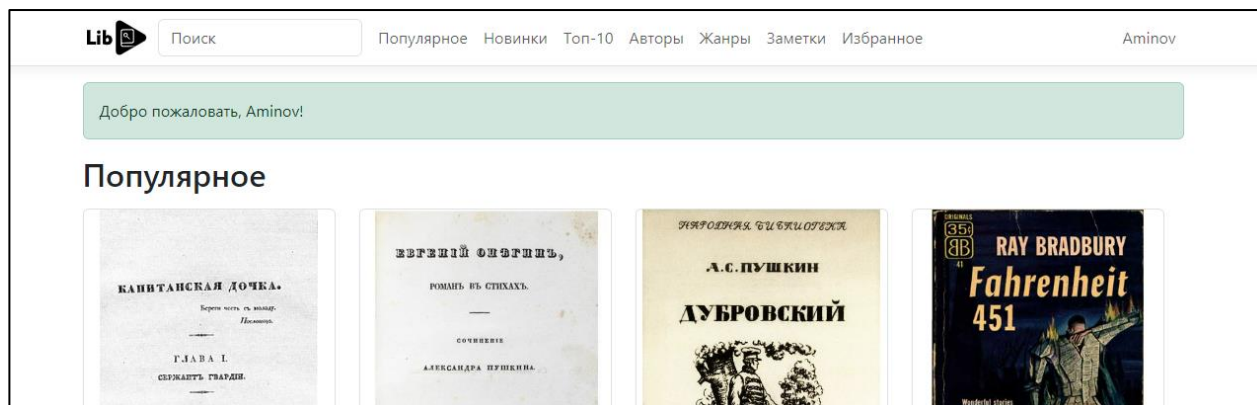


Рисунок 2.2.2 – Результат регистрации с корректными данными

Таблица 2.2.3 – Протокол тестирования регистрации на некорректных данных

Описание информационных полей для тестирования	
Наименование	Описание
Test Case #	Lib_test_2
Приоритет тестирования	Высокий
Название тестирования	Регистрация
Резюме испытания	Необходимо добиться корректного поведения веб-приложения при регистрации пользователя.
Шаги тестирования	Заполнить поля, необходимыми для регистрации, не корректными данными.
Данные тестирования	Login: neverket2@gmail.com; Password: 123456; Confirm password: 123; Name: Aminov.
Ожидаемый результат	Веб-приложение должно уведомить пользователя об ошибке.
Фактический результат	Веб-приложение уведомляет пользователя об ошибке.
Предпосылки	На шапке сайта нажата кнопка «Регистрация»
Постусловия	Система не зависает, находится в состоянии полной работоспособности.
Статус (Pass/Fail)	Pass

Проведем тестирование регистрации на некорректных данных. Результаты изображены на рисунках 2.2.3 – 2.2.4.

Рисунок 2.2.3 – Форма регистрации с некорректными данными

Регистрация

Пароли не совпадают

Зарегистрироваться

Рисунок 2.2.4 – Результат регистрации с некорректными данными

Таблица 2.2.4 – Протокол тестирования авторизации на корректных данных

Описание информационных полей для тестирования	
Наименование	Описание
Test Case #	Lib_test_3
Приоритет тестирования	Высокий
Название тестирования	Авторизация
Резюме испытания	Необходимо добиться корректного поведения веб-приложения при авторизации пользователя.
Шаги тестирования	Заполнить поля, необходимыми для авторизации с корректными данными.
Данные тестирования	Login: neverket@gmail.com; Password: 123456.
Ожидаемый результат	Веб-приложение должно открыть главную страницу с приветствием.
Фактический результат	Веб-приложение открывает главную страницу с приветствием.
Предпосылки	На шапке сайта нажата кнопка «Войти».
Постусловия	Система не зависает, находится в состоянии полной работоспособности.
Статус (Pass/Fail)	Pass

Проведем тестирование авторизации на корректных данных. Результаты изображены на рисунках 2.2.5 – 2.2.6.

Авторизация

neverket2@gmail.com

.....

Войти

Рисунок 2.2.5 – Форма авторизации с корректными данными

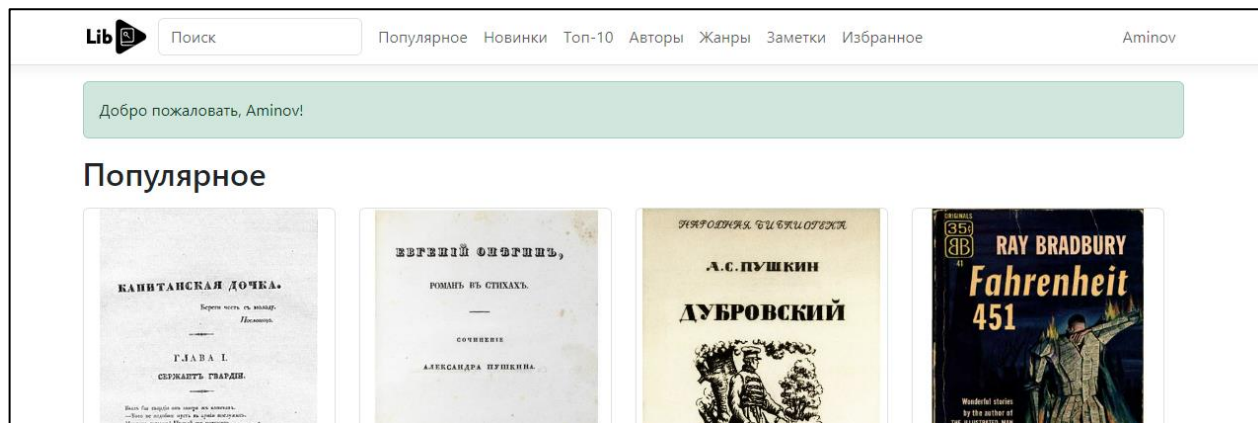


Рисунок 2.2.6 – Результат авторизации с корректными данными

Таблица 2.2.5 – Протокол тестирования сохранения заметки на корректных данных

Описание информационных полей для тестирования	
1	2
Наименование	Описание
Test Case #	Lib_test_4
Приоритет тестирования	Высокий
Название тестирование	Сохранение заметки
Резюме испытания	Необходимо добиться корректного поведения веб-приложения при сохранении заметки.
Шаги тестирования	Заполнить поля заметки корректными данными, нажать кнопку «Сохранить».
Данные тестирования	Заголовок: Тест; Текст: Текст заметки
Ожидаемый результат	Веб-приложение должно сохранить заметку, должно отобразиться сообщение об успешном сохранении.
Фактический результат	Веб-приложение сохраняет заметку, на странице отображается сообщение об успешном сохранении.
Предпосылки	Пользователь авторизовался, создал новую заметку.

Продолжение таблицы 2.2.5

1	2
Постусловия	Система не зависает, находится в состоянии полной работоспособности.
Статус (Pass/Fail)	Pass

Проведём тестирование сохранения заметки на корректных данных. Результат изображен на рисунке 2.2.7.

The screenshot shows the 'Lib' application interface. At the top, there is a search bar and navigation links: 'Популярное', 'Новинки', 'Топ-10', 'Авторы', 'Жанры', 'Заметки', and 'Избранное'. The user 'Aminov' is logged in. Below the navigation bar, there is a 'Назад' button. The main form contains a 'Тест' input field and a 'Текст заметки' text area. At the bottom of the form, there are two buttons: 'Сохранить' (green) and 'Удалить' (red).

Рисунок 2.2.7 – Форма сохранения заметки с корректными данными

The screenshot shows the 'Lib' application interface after saving a note. A green message bar at the top indicates 'Сохранено'. The rest of the interface, including the navigation bar, user name 'Aminov', and the form fields, is the same as in the previous screenshot.

Рисунок 2.2.8 – Результат сохранения заметки с корректными данными

Таблица 2.2.6 – Протокол тестирования написания рецензии на корректных данных

Описание информационных полей для тестирования	
1	2
Наименование	Описание
Test Case #	Lib_test_5

Продолжение таблицы 2.2.6

1	2
Приоритет тестирования	Высокий
Название тестирования	Написание рецензии
Резюме испытания	Необходимо добиться корректного поведения веб-приложения при написании рецензии
Шаги тестирования	Заполнить поля рецензии корректными данными, нажать кнопку «Сохранить».
Данные тестирования	Оценка: 5; Текст: помнится, в школе это было одно из немногих произведений, действительно тронувших моё сердце. Теперь же, при повторном прочтении, впечатления получились ещё более яркими.
Ожидаемый результат	Веб-приложение должно сохранить рецензию, показать ее на странице книги.
Фактический результат	Веб-приложение сохраняет рецензию, отображает ее на странице книги.
Предпосылки	Пользователь авторизировался, перешел на страницу книги.
Постусловия	Система не зависает, находится в состоянии полной работоспособности.
Статус (Pass/Fail)	Pass

Проведём тестирование написания рецензии на корректных данных. Результаты изображены на рисунках 2.2.9 – 2.2.10.

Александр Пушкин

Жанр

★

Рецензия

Оценка

5

Текст

помнится, в школе это было одно из немногих произведений, действительно тронувших моё сердце. Теперь же, при повторном прочтении, впечатления получились ещё более яркими

Отмена Сохранить

Рецензии 9

Рисунок 2.2.9 – Форма написания рецензии с корректными данными

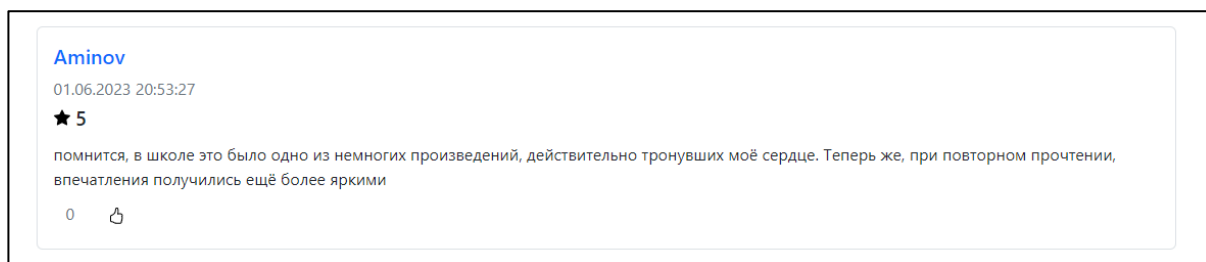


Рисунок 2.2.10 – Результат написания рецензии с корректными данными

Таблица 2.2.7 – Протокол тестирования написания рецензии на некорректных данных

Описание информационных полей для тестирования	
Наименование	Описание
Test Case #	Lib_test_6
Приоритет тестирования	Высокий
Название тестирования	Написание рецензии
Резюме испытания	Необходимо добиться корректного поведения веб-приложения при написании рецензии
Шаги тестирования	Заполнить поля рецензии некорректными данными, нажать кнопку «Сохранить».
Данные тестирования	Оценка: 5; Текст: «пробел».
Ожидаемый результат	Веб-приложение должно показать сообщение об ошибке на странице книги.
Фактический результат	Веб-приложение показывает сообщение об ошибке на странице книги.
Предпосылки	Пользователь авторизовался, перешел на страницу книги.
Постусловия	Система не зависает, находится в состоянии полной работоспособности.
Статус (Pass/Fail)	Pass

Проведём тестирование написания рецензии на корректных данных. Результаты изображены на рисунках 2.2.11 – 2.2.12.

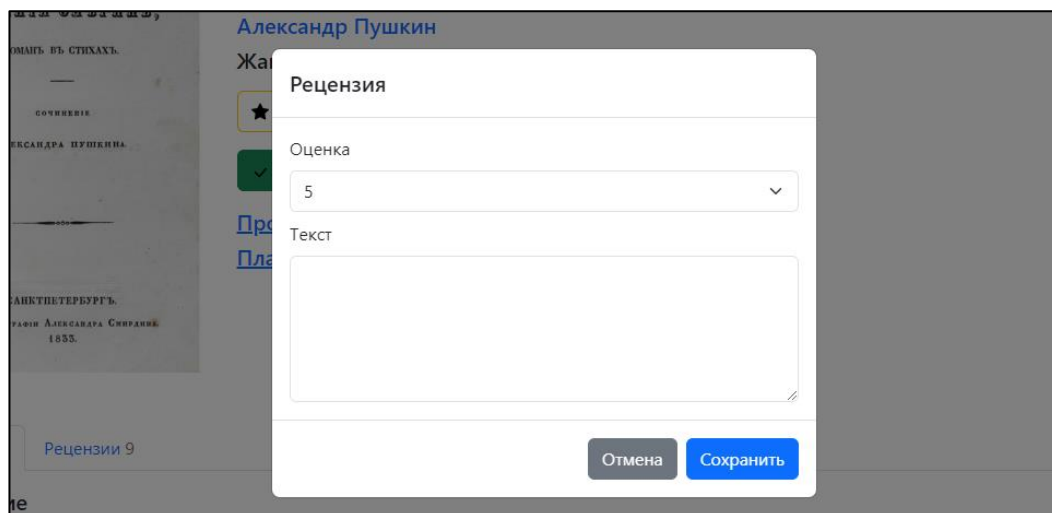


Рисунок 2.2.11 – Форма написания рецензии с некорректными данными

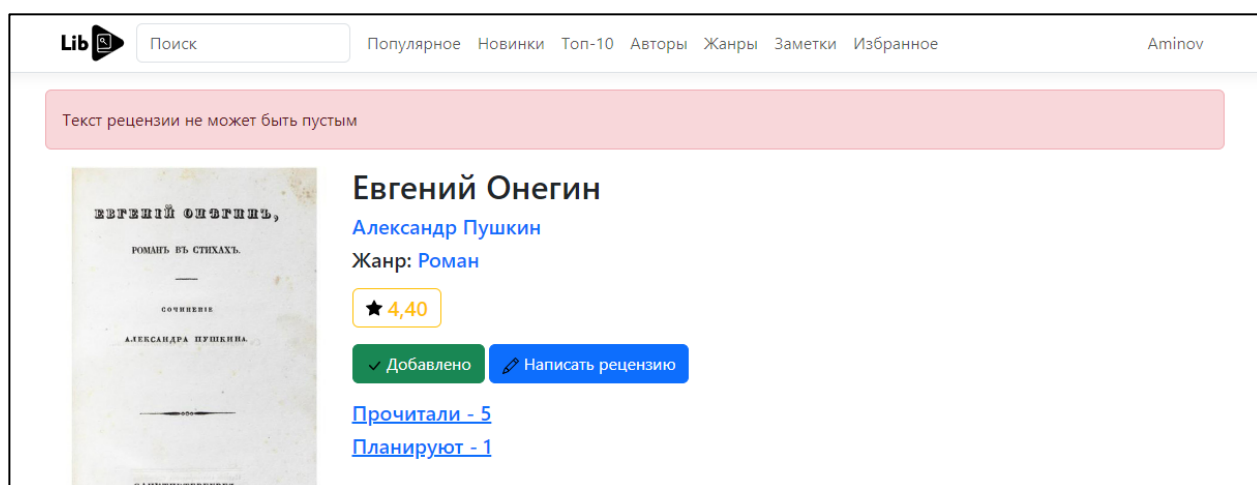


Рисунок 2.2.12 – Результат написания рецензии с некорректными данными

2.3 Руководство пользователя

Цель руководства заключается в расписании подробной инструкции и информации для пользователя, чтобы он смог самостоятельно пользоваться веб-приложением и правильно его эксплуатировал.

Веб-приложение должно быть размещено и эксплуатировано в сети Internet. Конечными пользователями веб-приложения являются читатель и администратор.

В зависимости от уровня доступа пользователя в системе ему доступны различные возможности.

Для читателей веб-программа позволяет пользоваться поиском книг по системе, просматривать подробную информацию о них. Просматривать рецензии на книги, также писать собственные. Добавлять книги себе в «Избранное» с выбранной меткой, например «Прочитано».

Для администраторов веб-программа позволяет изменять информацию книг, жанров, авторов. Модерировать рецензии на книги.

Для того, чтобы получить доступ к системе «Lib», необходимо открыть браузер и в адресной строке набрать: «https://localhost:7297/».

При переходе на сайт, открывается главная страница сайта, изображена на рисунке 2.3.1.

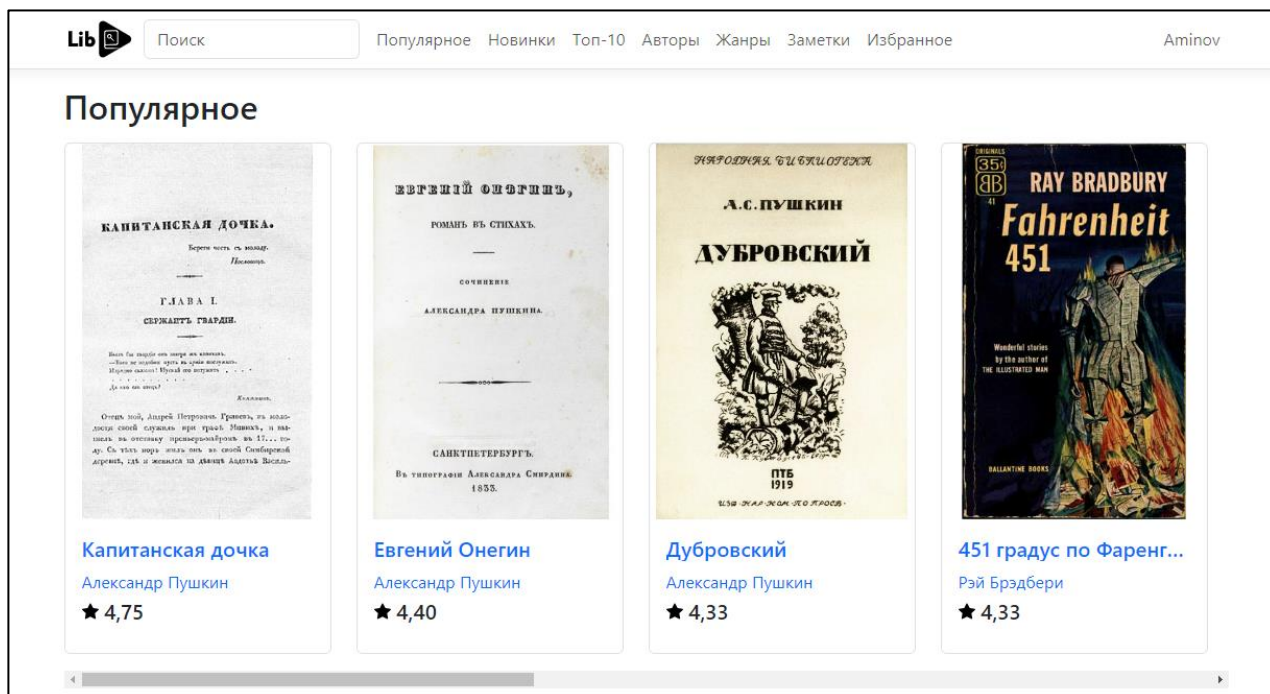


Рисунок 2.3.1 – Страница «Главная»

При нажатии на кнопку «Регистрация» открывается форма регистрации, изображено на рисунке 2.3.2.

Рисунок 2.3.2 – Страница «Регистрация»

При нажатии на кнопку «Войти» открывается форма авторизации, изображено на рисунке 2.3.3.

Рисунок 2.3.3 – Страница «Авторизация»

После успешной авторизации на сайте открывается главная страница с приветствием пользователя, изображено на рисунке 2.3.4.

Рисунок 2.3.4 – Страница «Главная» с приветствием пользователя

После нажатия по имени пользователя на шапке сайта открывается профиль пользователя, изображено на рисунке 2.3.5.

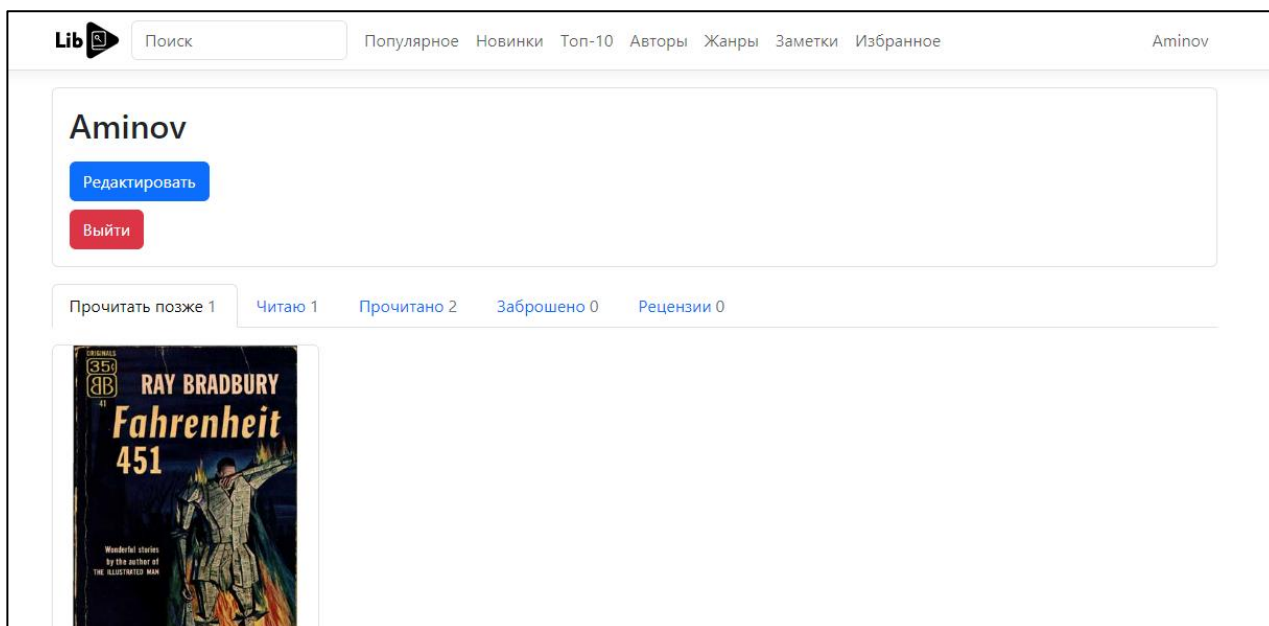


Рисунок 2.3.5 – Страница «Профиль»

После нажатия кнопки «Редактировать» открывается редактирование профиля, изображено на рисунке 2.3.6.

Рисунок 2.3.6 – Страница «Редактирование профиля»

После нажатия кнопки «Популярное» в шапке страницы открывается страница с популярными книгами, изображено на рисунке 2.3.7.

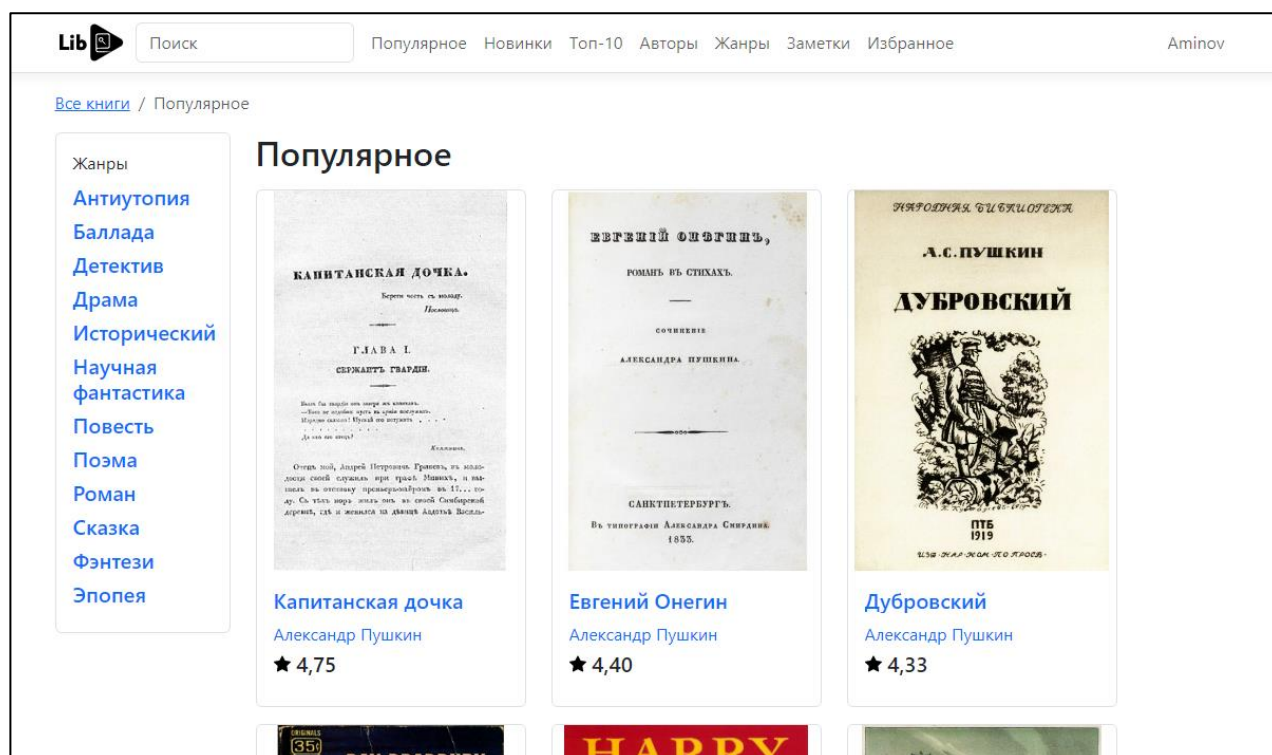


Рисунок 2.3.7 – Страница «Популярное»

После нажатия кнопки «Новинки» в шапке страницы открывается страница с новыми книгами, изображено на рисунке 2.3.8.



Рисунок 2.3.8 – Страница «Новинки»

После нажатия кнопки «Топ-10» в шапке страницы открывается страница с новыми книгами, изображено на рисунке 2.3.9.

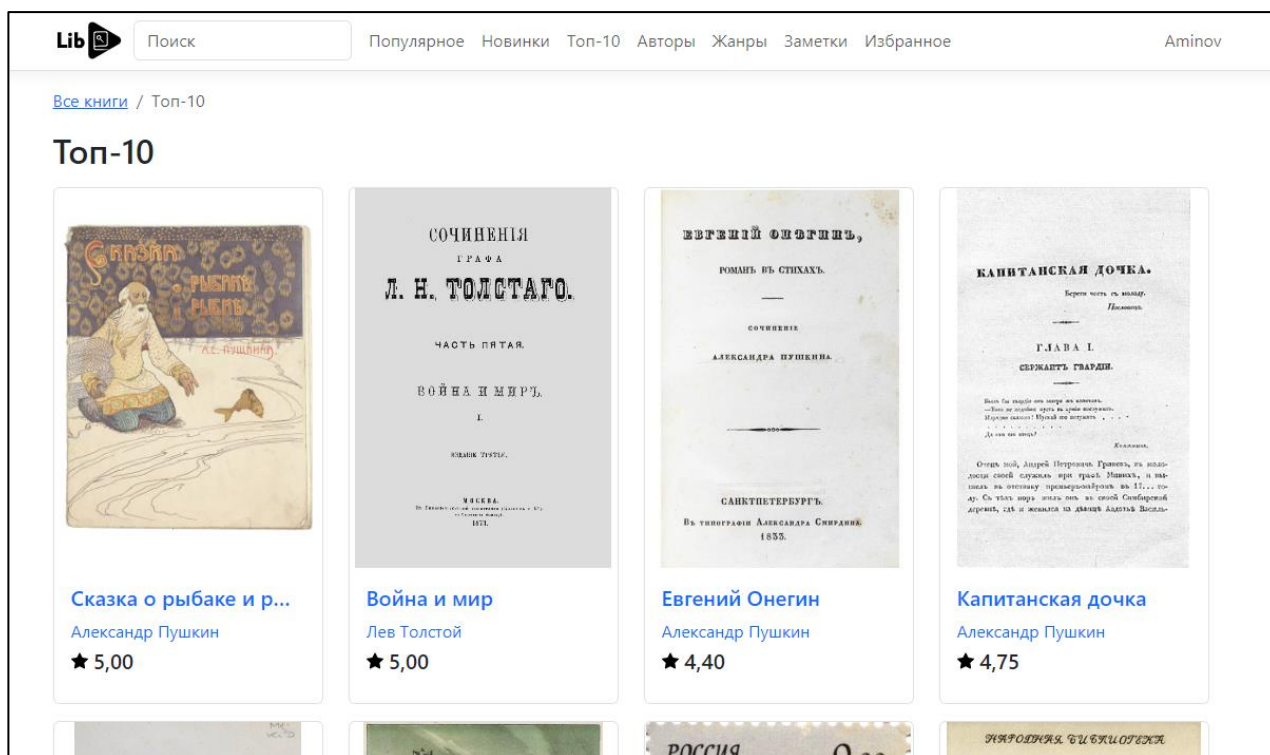


Рисунок 2.3.9 – Страница «Топ-10»

После нажатия кнопки «Заметки» в шапке страницы открывается страница с заметками текущего пользователя, изображено на рисунке 2.3.10.

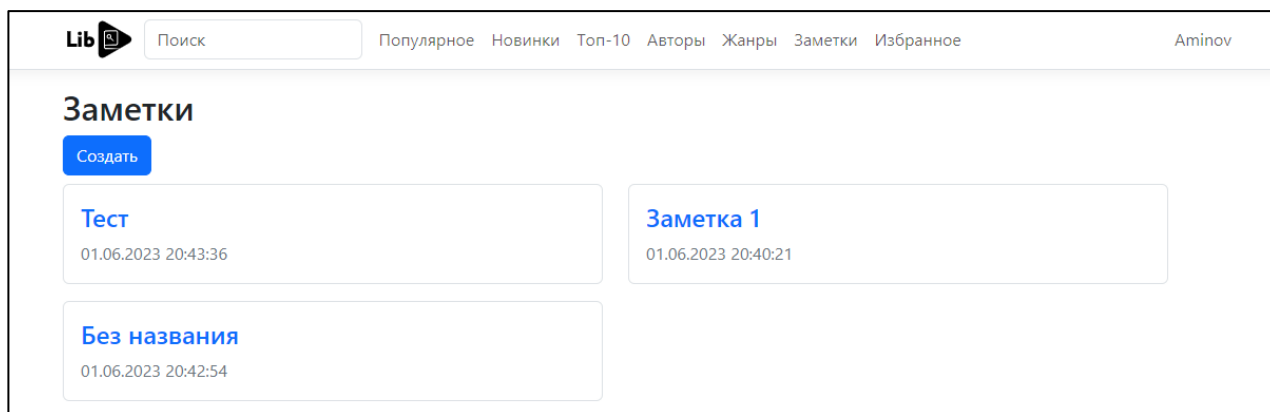


Рисунок 2.3.10 – Страница «Заметки»

После нажатия по заметке открывается страница с ее содержимым, изображено на рисунке 2.3.11.

Lib

Поиск

Популярное

Новинки

Топ-10

Авторы

Жанры

Заметки

Избранное

Aminov

Назад

Тест

Текст заметки

Сохранить

Удалить

Рисунок 2.3.11 – Страница «Заметка»

После нажатия кнопки «Избранное» в шапке страницы открывается страница с избранными книгами текущего пользователя, изображено на рисунке 2.3.12.

Lib

Поиск

Популярное

Новинки

Топ-10

Авторы

Жанры

Заметки

Избранное

Aminov

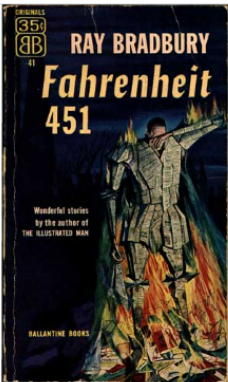
Избранное

Прочитать позже 1

Читаю 1

Прочитано 2

Заброшено 0



451 градус по Фаренг...

Рэй Брэдбери

★ 4,33

Рисунок 2.3.12 – Страница «Избранное»

При переходе на страницу автора, отображаются его книги, изображено на рисунке 2.3.13.

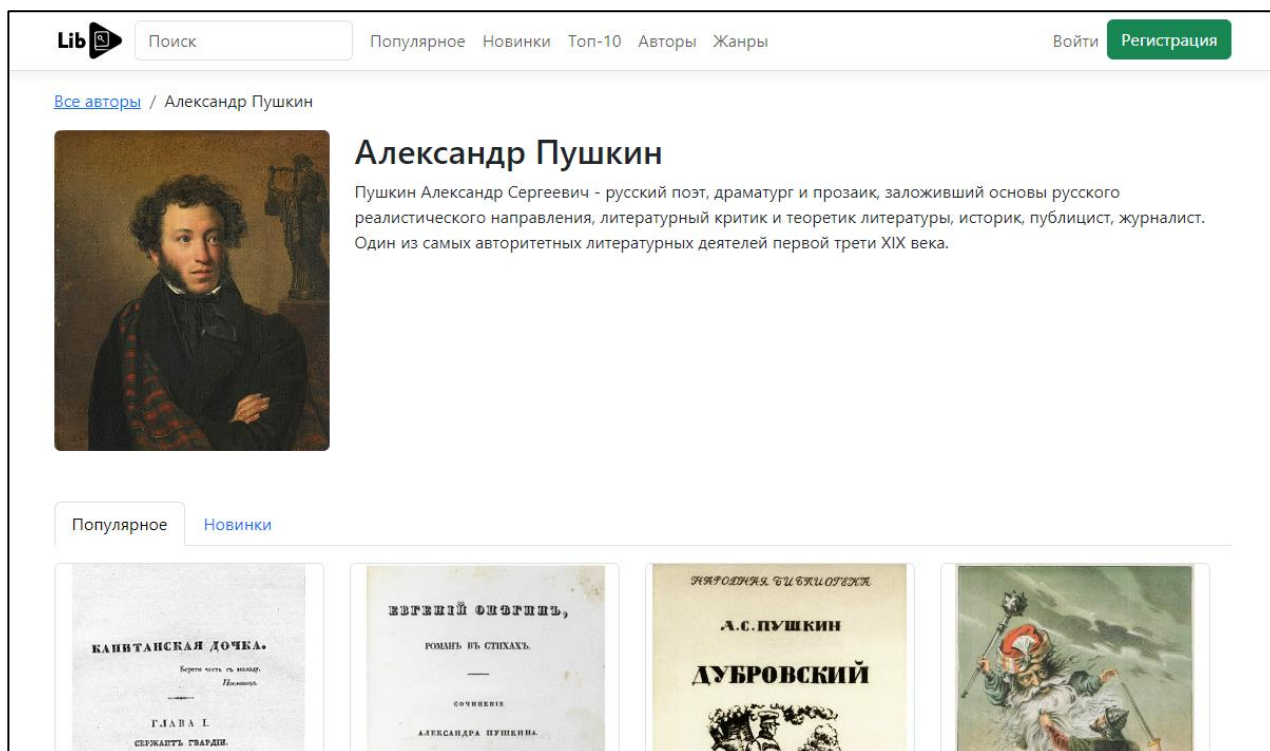


Рисунок 2.3.13 – Страница «Автор»

При переходе на страницу жанра, отображаются соответствующие книги, изображено на рисунке 2.3.14.

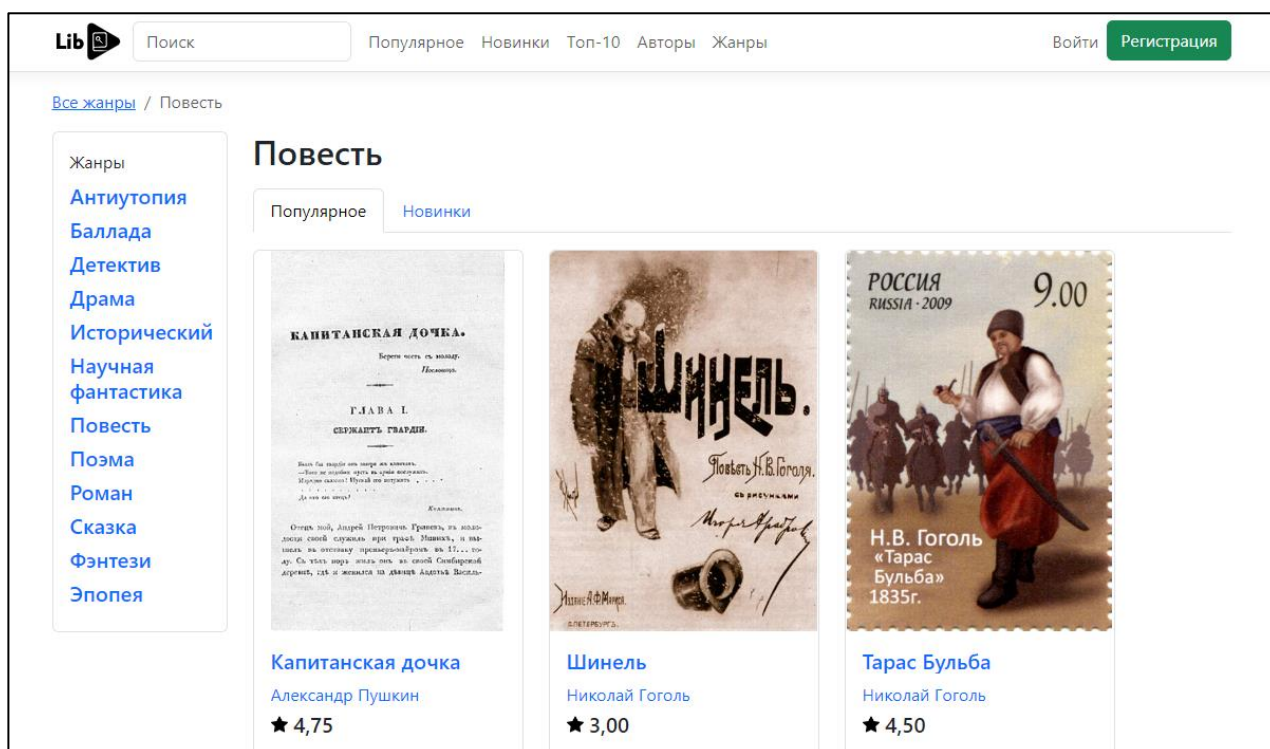


Рисунок 2.3.14 – Страница «Жанр»

При переходе на страницу поиска книги отображаются по заданному запросу и выбранному жанру, а авторы отображаются по заданному запросу, изображено на рисунке 2.3.15.



Рисунок 2.3.15 – Страница «Поиск книг»

При переходе на страницу книги, отображается ее подробная информация, изображена на рисунках 2.3.16 – 2.3.17.



Рисунок 2.3.16 – Страница «Книга» с описанием



Рисунок 2.3.17 – Страница «Книга» с рецензиями

При нажатии кнопки «Добавить» на странице книги отображается окно с формой добавления книги в «Избранное», изображено на рисунке 2.3.18.

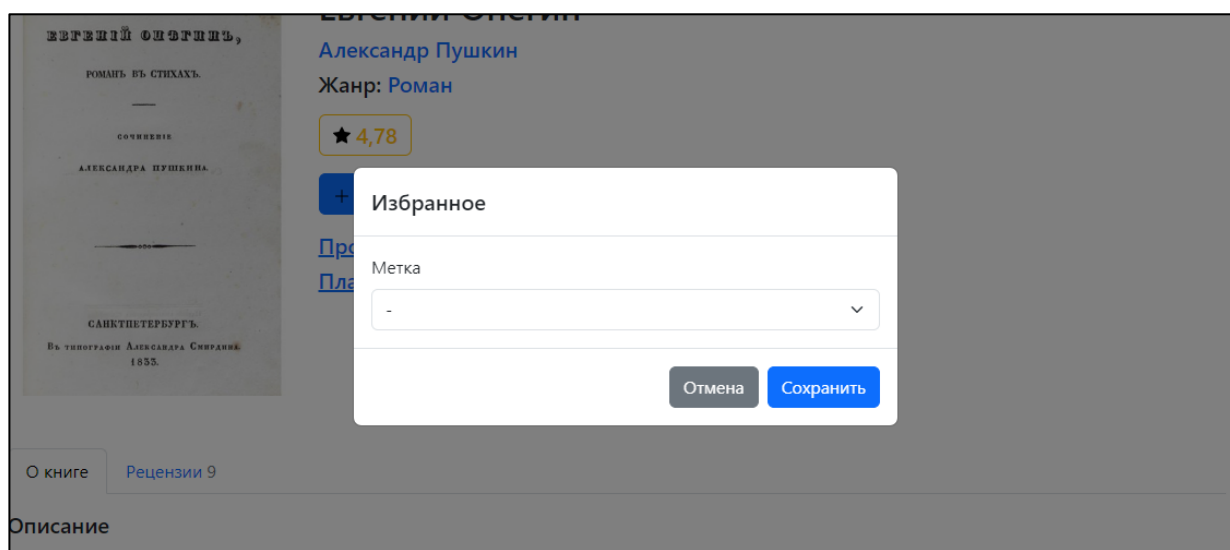


Рисунок 2.3.18 – Окно добавления книги в «Избранное»

При нажатии кнопки «Написать рецензию» на странице книги отображается окно с формой создания рецензии, изображено на рисунке 2.3.19.

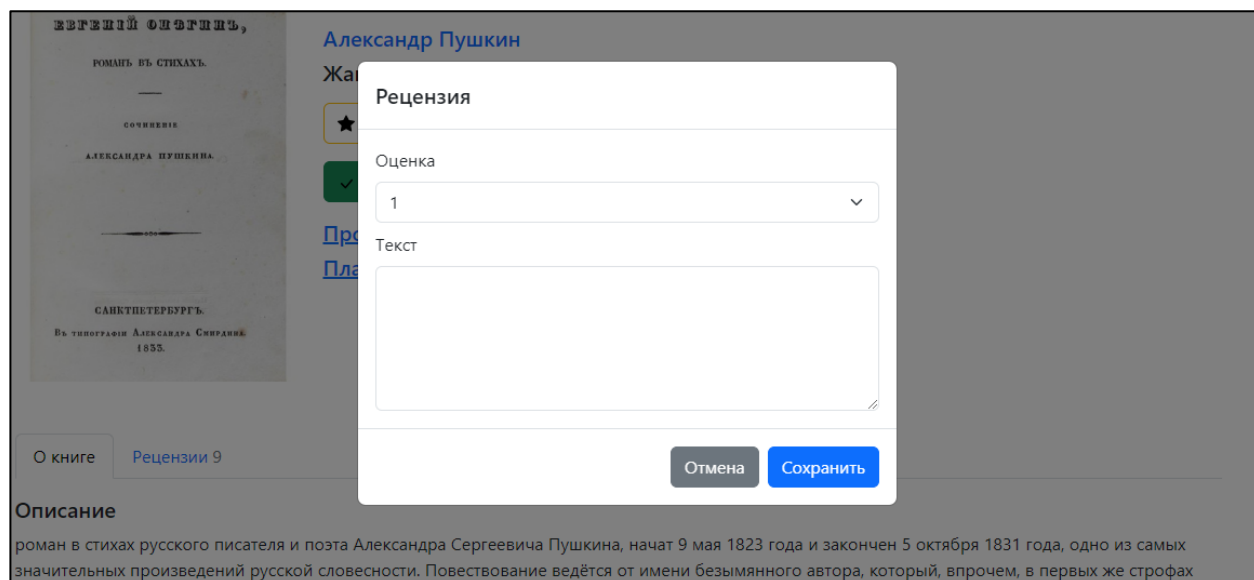


Рисунок 2.3.19 – Окно создания рецензии

При нажатии кнопки со средней оценкой книги выдвигается окно с подробной информацией об оценке книги, изображено на рисунке 2.3.20.

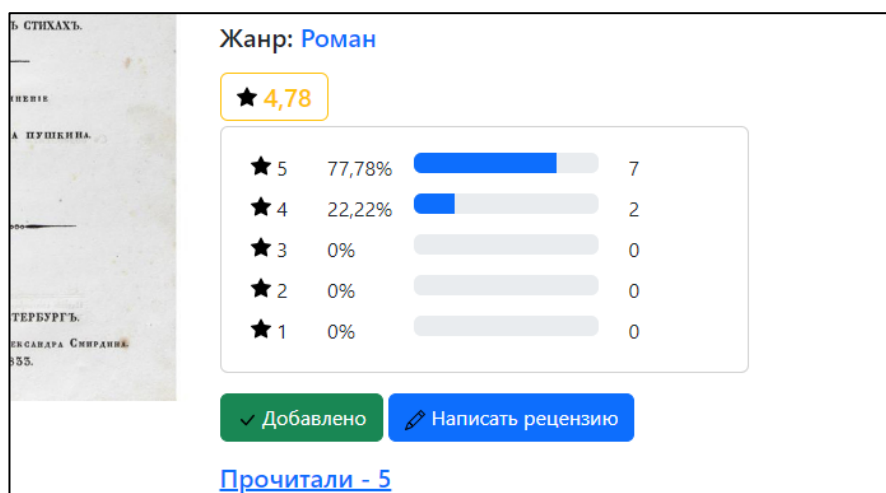


Рисунок 2.3.20 – Окно с подробной информацией об оценке книги

При переходе на страницу профиля с ролью администратора рядом с именем пользователя помечается роль администратора, также надпись есть и на шапке сайта, изображено на рисунке 2.3.21.

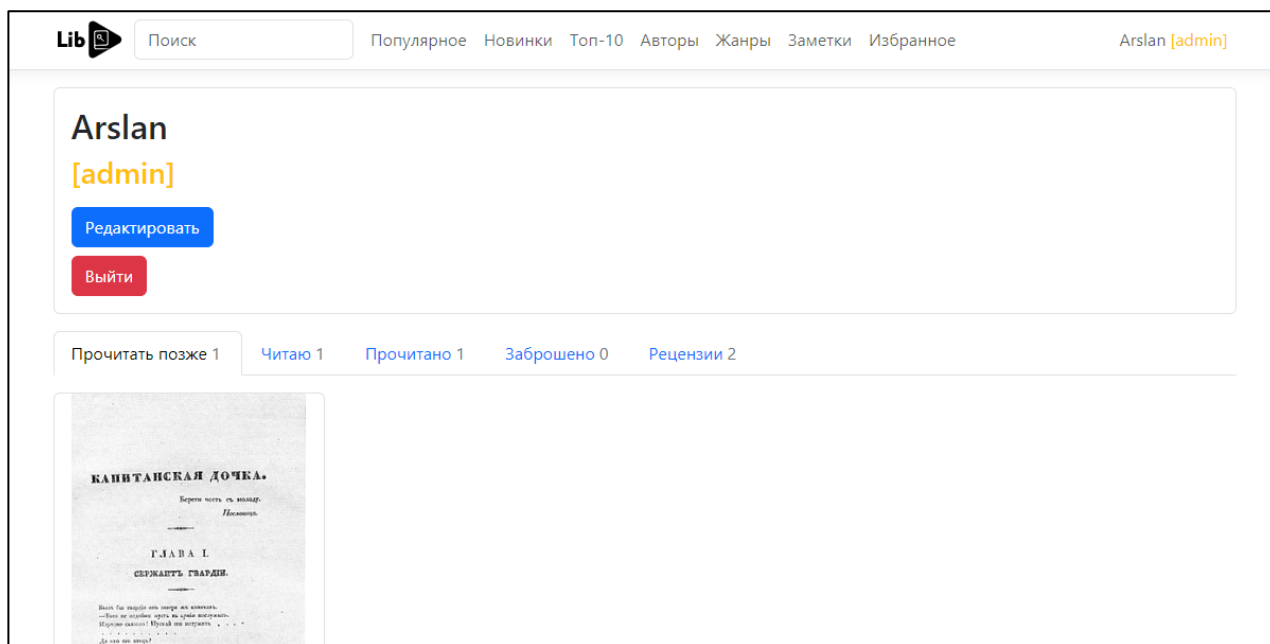


Рисунок 2.3.21 – Роль администратора

При переходе на страницу книги с ролью администратора и нажатии кнопки «Редактировать», открывается страница с формой для редактирования книги. Для добавления книги в систему, используется эта же форма. Страница изображена на рисунке 2.3.22.

Рисунок 2.3.22 – Страница «Редактирование книги»

При переходе на страницу автора с ролью администратора и нажатии кнопки «Редактировать», открывается страница с формой для редактирования автора. Для добавления автора в систему, используется эта же форма. Страница изображена на рисунке 2.3.23.

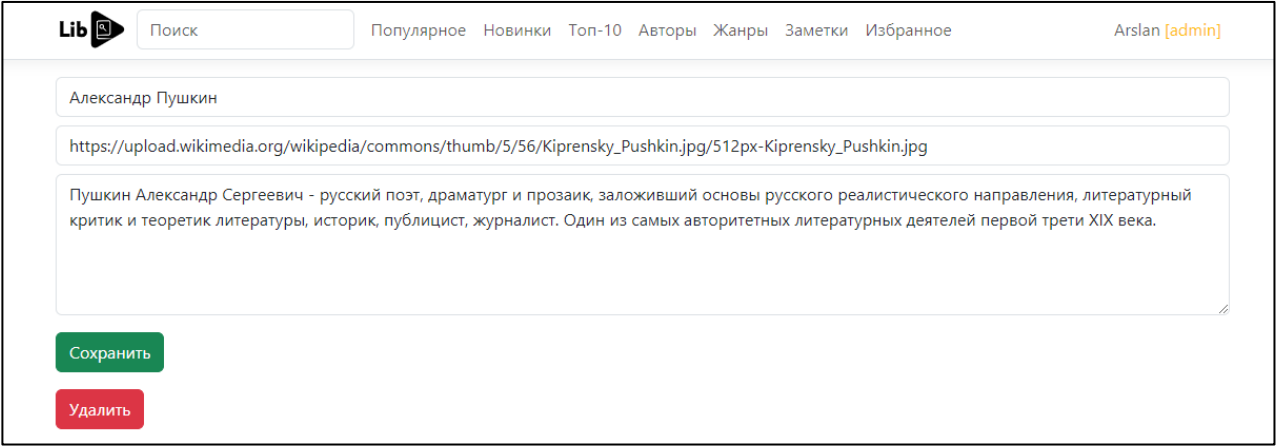


Рисунок 2.3.23 – Страница «Редактирование автора»

При переходе на страницу жанра с ролью администратора и нажатии кнопки «Редактировать», открывается страница с формой для редактирования жанра. Для добавления жанра в систему, используется эта же форма. Страница изображена на рисунке 2.3.24.

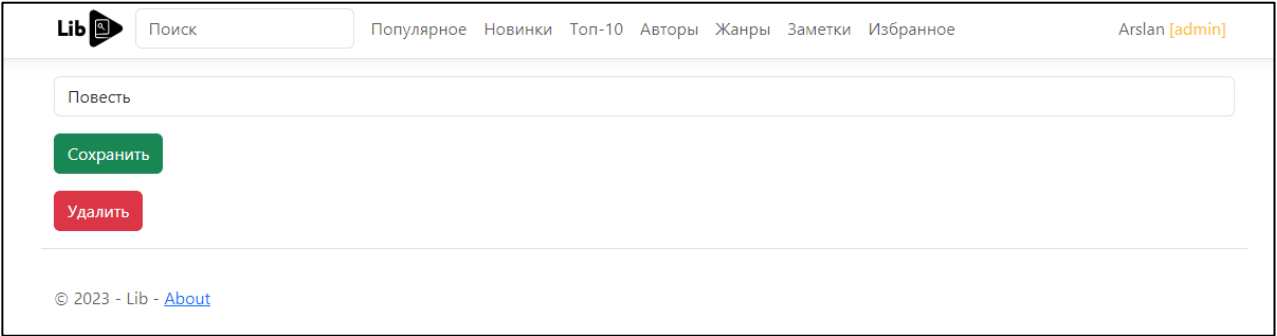


Рисунок 2.3.24 – Страница «Редактирование жанра»

При переходе на страницу книги с ролью администратора на рецензиях появляется кнопка снятия с публикации, изображено на рисунке 2.3.25.

Eugene Gross

08.02.2023 12:22:00

★ 4

эта классика обошла меня стороной в школе, а на сочинении, скорее всего же оно было, не могло не быть, спасло то, что в музыкальной школе мы проходили оперу, поэтому сам сюжет я знала) Ну да это не важно, важно то, что дошла я до чтения только сейчас, когда есть жизненный опыт и были влюбленности, безответные и не очень))) И это облегчает чтение, хоть оно и в стихах, которые всё же мне читать сложно, хотя, по идее, должно быть наоборот и читаться ещё быстрее. Но даже в стихотворной форме меня напрягали излишние подробности и уточнения, кто из героев кого читал, когда и сколько) Да, понятно, что это тоже их характеризует, но всё же).

2

👍

Снять с публикации

Lois Johnson

30.12.2022 14:09:00

★ 5

Нет, такие книги нужно читать и перечитывать, растаскивать на цитаты, запоминать наиболее важные строчки. Помнится, в школе девочки учили письмо Татьяне Онегину, так я до сих пор его помню, хоть только часть, но все же. Пушкин работал над романом 8 лет и назвал его своим подвигом. Главный герой - Евгений Онегин, молодой дворянин, которому пришлось переехать из Петербурга в деревню, чтобы проститься с дядей. Он любитель балов, женщин, долго поспать и покутить и, конечно же, в деревне ему стало скучно, да еще и соседка, Татьяна Ларина, влюбилась в него. Он не готов быть возлюбленным, мужем, отцом и отказывает ей. Жизнь потом повернулась к нему той же точкой, что и он в свое время к ней. Теперь он оказался ненужным. Мне хотелось бы финал поточнее, люблю, когда все понятно, а тут начинаешь задумываться: "а что же было дальше с Онегиным". Но тут автор оставляет нам право самим решить, что героя ждет дальше. Классика....

0

👍

Снять с публикации

Рисунок 2.3.25 – Кнопка снятия рецензии с публикации для роли администратора

При нажатии по кнопке «About» снизу страницы открывается страница с кратким описанием сайта, изображено на рисунке 2.3.26.

Lib

Поиск

Популярное Новинки Топ-10 Авторы Жанры Заметки Избранное

Arslan [admin]

About Lib

Книжный рекомендательный сервис для ведения читательского дневника, с подборками книг и рецензиями на них

Разработчик - Аминов Арслан 19П-1

Рисунок 2.3.26 – Страница «О сайте»

3 Экономический раздел

3.1 Расчет затрат на создание программного продукта

Расчет себестоимости машинного часа эксплуатации вычислительной и оргтехники (ВиОТ):

$$C_{\text{м.ч.}} = \frac{\sum_{i=1}^n 3_i}{F_{\text{п}} K_{\text{г}}}, \quad (3.1.1)$$

где $C_{\text{м.ч.}}$ - себестоимость машинного часа;

3_i - годовые затраты, связанные с эксплуатацией и обслуживанием ВиОТ;

$F_{\text{п}}$ - годовой полезный фонд времени работы единицы оборудования;

$K_{\text{г}} = 0,95$ - коэффициент готовности.

$$F_{\text{п}} = F_{\text{н}} (1 - \alpha_{\text{р}}), \quad (3.1.2)$$

где $F_{\text{н}}$ – номинальный годовой фонд рабочего времени в часах;

$\alpha_{\text{р}} = (0,05 \dots 0,2)$ – коэффициент, учитывающий время, затраченное на ремонт, настройку, обслуживание ВиОТ.

$$F_{\text{п}} = 1946 * (1 - 0,125) = 1702 \text{ (ч.)}.$$

Расчет суммарных годовых затрат.

Для расчета годовых затрат, необходимо определить балансовую стоимость $C_{\text{БАЛ}}$ условного комплекта, необходимого для создания программного продукта.

Таблица 3.1.1 - Состав условного комплекта

№	Наименование	Кол-во, шт.	Цена, руб.
1	Ноутбук	1	90000
2	Мышь	1	1000
	Итого:		91000

$$C_{\text{БАЛ}} = \sum_{i=1}^n C_i K_i + P_{\text{д}}, \quad (3.1.3)$$

где C_i - цена единицы условного комплекта;

K_i - количество единиц условного комплекта;

P_D - дополнительные расходы на доставку, установку, первоначальную наладку.

$$P_D = 0,1 \sum_{i=1}^n C_i K_i \quad (3.1.4)$$

$$P_D = 0,1 \cdot (90000 \cdot 1 + 1000 \cdot 1) = 9100 \text{ (руб.)}$$

$$C_{\text{БАЛ}} = 90000 \cdot 1 + 1000 \cdot 1 + 9100 = 100100 \text{ (руб.)}$$

Затраты на материалы:

$$Z_M = 0,02 \cdot C_{\text{БАЛ}} \quad (3.1.5)$$

$$Z_M = 0,02 \cdot 100100 = 2002 \text{ (руб.)}$$

Амортизационные отчисления ВиОТ.

$$AO_{\text{ОБОР}} = C_{\text{БАЛ}} \cdot H_A^{\text{ОБОР}}, \quad (3.1.6)$$

где $H_A^{\text{ОБОР}} = 0,2$ - норма амортизационных начислений.

$$AO_{\text{ОБОР}} = 100100 \cdot 0,2 = 20020 \text{ (руб.)}$$

Износ программных продуктов.

Условный комплект обладает необходимыми продуктами, представленными в таблице 3.1.3.

Таблица 3.1.3 - Используемые программные средства

Наименование	Цена (руб.)
Microsoft Windows 10 Home	12990
Visual Studio 2022	0
Google Chrome	0
Итого ($\sum C_{\text{ПП}}$)	12990

Амортизационные отчисления программных продуктов - $AO_{\text{ПП}}$.

$$AO_{\text{ПП}} = \sum C_{\text{ПП}} \cdot H_A^{\text{ПП}} \cdot 10^{-2}, \quad (3.1.7)$$

где $\sum C_{\text{ПП}}$ – суммарная стоимость программных продуктов.

$H_A^{\text{ПП}} = 0,5$ – норма амортизационных начислений.

$$AO_{\text{ПП}} = 12990 \cdot 0,5 \cdot 0,01 = 64,95 \text{ (руб.)}$$

Расходы на содержание и эксплуатацию оборудования.

Расходы на содержание и эксплуатацию оборудования состоят из:

– затрат на ремонт и специальное обслуживание

$$З_{РЕМ} = 0,03 * C_{БАЛ}; \quad (3.1.8)$$

– затрат на электрическую энергию

$$З_{э} = \sum M * F_{П} * K_{Г} * Ц_{кВт/ч}, \quad (3.1.9)$$

где $\Sigma M = 0,518$ кВт - суммарная мощность,

$K_{Г} = 0,95$ - коэффициент готовности.

$Ц_{кВт/ч} = 4,01$ руб. - стоимость кВт/ч,

$З_{РЕМ} = 0,03 * 100100 = 3003$ (руб.).

$З_{э} = 0,518 * 1702 * 0,95 * 4,01 = 3358,6$ (руб.).

Расходы на содержание и эксплуатацию оборудования

$$З_{РЭ} = З_{РЕМ} + З_{э} \quad (3.1.10)$$

$З_{РЭ} = 3003 + 3358,6 = 6361,6$ (руб.).

Суммарные годовые затраты.

$$\Sigma Z_i = Z_M + AO_{ОБОР} + AO_{ПП} + З_{РЭ} \quad (3.1.11)$$

$\Sigma Z_i = 2002 + 20020 + 64,95 + 6361,6 = 28448,55$ (руб.)

Себестоимость машинного часа, из формулы (3.1.1), составляет:

$$C_{м.ч.} = \frac{\sum_{i=1}^n Z_i}{F_{П} K_{Г}} = 28448,55 / (1702 * 0,95) = 17,59 \text{ (руб.)}.$$

3.2 Расчет цены предложения

Фонд оплаты труда за время работы над программным продуктом – ФОТ:

$$ФОТ = O * T_o * (1 + K_d) * (1 + K_p), \quad (3.2.1)$$

где O – оклад сотрудника, работающего над продуктом;

T_o – общее время работы над программным продуктом;

$K_d = 0,15$ – коэффициент дополнительной заработной платы;

$K_p = 0,15$ – районный коэффициент.

$$\Phi OT = 40000 * 1 * (1 + 0,15) * (1 + 0,15) = 52900 \text{ (руб.)}$$

Начисления на ФОТ:

$$H_{з/п} = \Phi OT * 0,302 \quad (3.2.2)$$

$$H_{з/п} = 52900 * 0,302 = 15975,8 \text{ (руб.)}$$

Затраты, связанные с эксплуатацией и обслуживанием ВиОТ – $Z_{овт}$:

$$Z_{овт} = T_M * ЧР_M * K_{и} * N * C_{м.ч.}, \quad (3.2.3)$$

где T_M = машинное время работы над программным продуктом;

$ЧР_M$ = число рабочих часов в месяце;

N = количество условных комплектов.

$K_{и}$ = коэффициент использования оборудования;

$$Z_{овт} = 1 * 160 * 0,9 * 1 * 17,59 = 2532,96 \text{ (руб.)}$$

Затраты на специальные программные продукты – $Z_{спп}$. Если специальные программные продукты не использовались, то $Z_{спп} = 0$ (руб.).

Специальные программные продукты не использовались.

$$Z_{спп} = 0 \text{ (руб.)}$$

Затраты на хозяйственные операции и нужды ($Z_{хн}$):

$$Z_{хн} = 0 \text{ (руб.)}$$

Накладные расходы:

$$P_H = (0,3...0,6) * \Phi OT \quad (3.2.4)$$

$$P_H = 0,45 * 52900 = 23805 \text{ (руб.)}$$

Полные затраты на разработку программного продукта:

$$Z_{пол} = \Phi OT + H_{з/п} + Z_{овт} + Z_{спп} + Z_{хн} + P_H \quad (3.2.5)$$

$$Z_{пол} = 52900 + 15975,8 + 2532,96 + 0 + 0 + 23805 = 95213,76 \text{ (руб.)}$$

Расчет установочной прибыли:

$$ПР_y = Z_{пол} * P_y * 0,01, \quad (3.2.6)$$

где $P_y = 20\%$ - установочная рентабельность.

$$ПР_y = 95213,76 * 0,2 * 0,01 = 190,43 \text{ (руб.)}$$

Расчет величины налога на добавленную стоимость (НДС):

$$НДС = (З_{ПОЛ} + ПР_{У}) * 0,2 \quad (3.2.7)$$

$$НДС = (95213,76 + 190,43) * 0,2 = 19080,84 \text{ (руб.)}.$$

Цена предложения разработанного программного продукта:

$$Ц_{ПР} = З_{ПОЛ} + ПР_{У} + НДС \quad (3.2.8)$$

$$Ц_{ПР} = 95213,76 + 190,43 + 19080,84 = 114485,03 \text{ (руб.)}.$$

					40.А-2075-2023 09.02.07 ДП-ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		46

ЗАКЛЮЧЕНИЕ

Дипломный проект на тему «Разработка веб-приложения для ведения читательского дневника» был выполнен в соответствии с поставленным заданием. В ходе выполнения дипломного проекта было разработано веб-приложение «Lib».

При выполнении дипломного проекта были решены следующие задачи:

- изучена предметная область;
- спроектирована база данных;
- разработана структура и дизайн веб-приложения;
- реализованы функции для работы пользователей: читатель, администратор;

В результате проделанной работы было разработано веб-приложение для ведения читательского дневника. Оно автоматизирует доступ к базе данных, оптимизирует ведение читательского дневника, упрощает поиск интересующих книг по их рецензиям.

Написанное веб-приложение «Lib» протестировано на данных контрольного примера. Исходный код представлен в приложении Б. Результат работы веб-приложения представлен в приложении В в виде списков: популярных книг, книг с самыми высокими оценками, книг пользователя.

					40.А-2075-2023 09.02.07 ДП-ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		47

Приложение А

Контрольный пример

Таблица А.1 – Список авторов

ID автора	Имя	Фото	Биография
1	Александр Пушкин	https://upload.wikimedia.org/wikipedia/commons/thumb/5/56/Kiprensky_Pushkin.jpg/512px-Kiprensky_Pushkin.jpg	Пушкин Александр Сергеевич - русский поэт, драматург и прозаик, заложивший основы русского реалистического направления, литературный кри...
2	Михаил Лермонтов	https://upload.wikimedia.org/wikipedia/commons/thumb/2/24/Mikhail_lermontov.jpg/512px-Mikhail_lermontov.jpg	Лермонтов Михаил Юрьевич - русский поэт, прозаик, драматург, художник. Поручик лейб-гвардии Гусарского полка. Творчество Лермонтова, в кото...
3	Николай Гоголь	https://upload.wikimedia.org/wikipedia/commons/b/b2/N.Gogol_by_F.Moller_%281840%2C_Tretyakov_gallery%29.jpg	Никола́й Васи́льевич Го́голь (при рождении Яно́вский, с 1821 года — Го́голь-Яно́вский; 20 марта [1 апреля] 1809, Сорочинцы, Миргородский уезд, Пол...
4	Лев Толстой	https://upload.wikimedia.org/wikipedia/commons/thumb/d/d3/Tolstoy_Leo_port.jpg/512px-Tolstoy_Leo_port.jpg	Толстой Лев Николаевич - один из наиболее известных русских писателей и мыслителей, один из величайших в мире писателей-романистов.

Таблица А.2 – Автор - книга

ID связи	ID автора	ID книги
1	2	3
1	1	1

Продолжение приложения А

Продолжение таблицы А.2

1	2	3
2	1	2
3	1	3
4	1	4
5	1	5
6	2	6
7	2	7
8	4	8
9	4	9
10	3	10
11	3	11
12	3	12
13	6	13
14	5	14

Таблица А.3 – Список ролей пользователей

ID роли	Название
1	reader
2	admin

Продолжение приложения А

Таблица А.4 – Список книг

ID связи	Название	Описание	Фото	Средний рейтинг	Дата написания
1	2	3	4	5	6
1	Евгений Онегин	роман в стихах русского писателя и поэта Александра Сергеевича Пушкина, начат 9 мая 1823 года и закончен 5 октября 1831 года, одно из самых значи...	https://upload.wikimedia.org/wikipedia/c ommons/e/ed/Eugene_Onegin_book_edit ion.jpg	4.777777 777	1833 год
2	Капитанская дочка	исторический роман (или повесть) Александра Пушкина, действие которого происходит во время восстания Емельяна Пугачёва. Впервые опублико...	https://upload.wikimedia.org/wikipedia/c ommons/thumb/9/9c/Captain%27s_Daug hter_1837.jpg/256px- Captain%27s_Daughter_1837.jpg	4.75	1836 год
3	Руслан и Людмила	первая законченная поэма Александра Сергеевича Пушкина; волшебная сказка, вдохновлённая древнерусскими былинами.	https://upload.wikimedia.org/wikipedia/c ommons/b/b7/%D0%91%D0%BE%D0% B9_%D0%A7%D0%B5%D1%80%D0% BD%D0%BE%D0%BC%D0%BE%D1 %80%D0%B0_%D1%81_%D0%A0%D 1%83%D1%81%D0%BB%D0%B0%D0 %BD%D0%BE%D0%BC.jpg	4.666666 666	1820 год

Продолжение приложения А

Продолжение таблицы А.4

4	Дубровский	наиболее известный разбойничий роман на русском языке, необработанное для печати (и неоконченное) произведение А. С. Пушкина. Повествует о ...	https://upload.wikimedia.org/wikipedia/commons/a/a6/Pushkin_Dubrovsky_1919.jpg	4.333333 333	1841 год
5	Сказка о рыбаке и рыбке	сказка А. С. Пушкина. Написана 2 (14) октября 1833 года. Впервые напечатана в 1835 году в журнале «Библиотека для чтения» [1]. В рукописи есть пометка...	https://upload.wikimedia.org/wikipedia/commons/3/3b/%D0%A1%D0%BA%D0%B0%D0%B7%D0%BA%D0%B0_%D0%BE_%D1%80%D1%8B%D0%B1%D0%B0%D0%BA%D0%B5_%D0%B8_%D1%80%D1%8B%D0%B1%D0%BA%D0%B5_1913.jpg	5	1835 год
6	Бородино	баллада поэта Михаила Юрьевича Лермонтова. Было написано в 1837 году. Опубликовано в журнале «Современник» в том же 1837 году. Посвящено Бород...	https://upload.wikimedia.org/wikipedia/commons/3/3d/Lermontov-Borodino-Sovremennik1837.jpg	3.666666 666	1837 год
7	Герой нашего времени	первый в русской прозе социально-психологический роман, написанный Михаилом Юрьевичем Лермонтовым в 1838—1840 годах. Классика русской литер...	https://upload.wikimedia.org/wikipedia/commons/4/41/Geroy_nashego_vremeni.png	4.75	1840 год

Продолжение приложения А

Продолжение таблицы А.4

8	Анна Каренина	роман Льва Толстого о трагической любви замужней дамы Анны Карениной и блестящего офицера Алексея Вронского на фоне счастливой семейной ...	https://upload.wikimedia.org/wikipedia/commons/thumb/c/c7/AnnaKareninaTitle.jpg/512px-AnnaKareninaTitle.jpg	4	1875— 1877 года
9	Война и мир	роман-эпопея Льва Николаевича Толстого, описывающий русское общество в эпоху войн против Наполеона в 1805—1812 годах. Эпилог романа доводит п...	https://upload.wikimedia.org/wikipedia/commons/2/2a/T25-011.jpg	5	1865— 1869 года
10	Шинель	одна из петербургских повестей Николая Гоголя. Увидела свет в 3-м томе собрания сочинений Гоголя, отпечатанного на исходе 1842 года и поступи...	https://upload.wikimedia.org/wikipedia/commons/b/bf/Gogol_Palto.jpg	3	1843 год
11	Мёртвые души	произведение Николая Васильевича Гоголя, жанр которого сам автор обозначил как поэма. Писать книгу Гоголь начал в 1835 году как трёхтомник. ...	https://upload.wikimedia.org/wikipedia/commons/b/bc/Dead_Souls_%28novel%29_Nikolai_Gogol_1842_title_page.jpg	4	1842 год

Продолжение приложения А

Продолжение таблицы А.4

12	Тарас Бульба	События произведения происходят в среде запорожских казаков в первой половине XVII века[1]. В основу повести Н. В. Гоголя легла история казацк...	https://upload.wikimedia.org/wikipedia/commons/0/0c/%D0%A2%D0%B0%D1%80%D0%B0%D1%81_%D0%91%D1%83%D0%BB%D1%8C%D0%B1%D0%B0.jpg	4.5	1835 год
13	451 градус по Фаренгейту	научно-фантастический роман-антиутопия Рэя Брэдбери, изданный в 1953 году. Роман описывает американское общество близкого будущего, в котор...	https://upload.wikimedia.org/wikipedia/ru/d/d3/451_%D0%B3%D1%80%D0%B0%D0%B4%D1%83%D1%81_%D0%BF%D0%BE_%D0%A4%D0%B0%D1%80%D0%B5%D0%BD%D0%B3%D0%B5%D0%B9%D1%82%D1%83.jpg	4.333333 333	1953 год
14	Гарри Поттер и философски й камень	первый роман в серии книг про юного волшебника Гарри Поттера, написанный Дж. К. Роулинг. В нём рассказывается, как Гарри узнает, что он волшеб...	https://upload.wikimedia.org/wikipedia/en/6/6b/Harry_Potter_and_the_Philosopher%27s_Stone_Book_Cover.jpg	3.666666 666	26 июня 1997 года

Таблица А.5 – Список избранных книг

ID связи	ID пользователя	ID книги	Дата добавления	ID отметки
1	2	3	4	5
2	1	13	2023-04-02 10:10:10	2

Продолжение приложения А

Продолжение таблицы А.5

1	2	3	4	5
5	2	14	2023-05-03 10:10:10	2
6	2	13	2023-05-22 10:26:13	1
8	1	2	2023-06-02 13:24:04	1
9	1	14	2023-06-03 14:08:16	3
10	3	1	2023-06-07 17:57:58	3
11	3	2	2023-06-07 17:58:32	3
12	3	3	2023-06-07 18:00:57	3
13	3	4	2023-06-07 18:01:19	1
14	3	5	2023-06-07 18:01:30	2
15	3	6	2023-06-07 18:01:36	2
16	3	7	2023-06-07 18:02:27	3
17	3	8	2023-06-07 18:04:45	3
18	3	13	2023-06-07 18:08:13	3
19	3	14	2023-06-07 18:08:23	4
20	4	1	2023-06-07 18:08:59	3
21	4	2	2023-06-07 18:09:43	3
22	4	3	2023-06-07 18:10:04	3
23	4	4	2023-06-07 18:10:35	3

Продолжение приложения А

Продолжение таблицы А.5

1	2	3	4	5
24	5	4	2023-06-07 18:12:52	3
25	5	1	2023-06-07 18:13:26	3
26	5	5	2023-06-07 18:13:43	3
27	5	6	2023-06-07 18:14:44	3
28	5	7	2023-06-07 18:15:56	3
29	5	9	2023-06-07 18:16:38	3
30	5	8	2023-06-07 18:16:57	4
31	5	10	2023-06-07 18:17:07	3
32	5	11	2023-06-07 18:17:23	3
33	5	12	2023-06-07 18:17:45	3
34	5	13	2023-06-07 18:18:03	3
35	5	14	2023-06-07 18:18:30	3
36	5	2	2023-06-07 18:19:05	1
37	6	2	2023-06-07 18:20:39	3
38	6	1	2023-06-07 18:22:16	1
39	6	3	2023-06-07 18:22:25	1
40	6	4	2023-06-07 18:22:33	3
41	6	5	2023-06-07 18:23:18	1

Продолжение приложения А

Продолжение таблицы А.5

1	2	3	4	5
42	6	6	2023-06-07 18:23:26	3
43	6	7	2023-06-07 18:23:43	3
44	6	8	2023-06-07 18:24:08	2
45	6	9	2023-06-07 18:24:15	3
46	6	10	2023-06-07 18:24:37	2
47	6	11	2023-06-07 18:24:42	3
48	6	12	2023-06-07 18:25:24	1
49	6	13	2023-06-07 18:25:30	3
50	6	14	2023-06-07 18:26:00	3
51	7	1	2023-06-07 18:26:47	3
52	7	2	2023-06-07 18:26:52	1
53	7	3	2023-06-07 18:27:02	3
54	7	4	2023-06-07 18:27:51	4
55	7	5	2023-06-07 18:28:01	3
56	7	6	2023-06-07 18:28:08	3
57	7	7	2023-06-07 18:28:35	3
58	7	8	2023-06-07 18:29:08	2
59	7	9	2023-06-07 18:29:13	2

Продолжение приложения А

Продолжение таблицы А.5

1	2	3	4	5
60	8	9	2023-06-07 18:30:59	3
61	8	2	2023-06-07 18:31:37	3
62	2	2	2023-06-07 21:31:45	3
63	2	1	2023-06-02 00:15:56	3
64	1	1	2023-06-02 00:24:17	3

Таблица А.6 – Список жанров

ID жанра	Название
1	2
1	Детектив
2	Повесть
3	Роман
4	Сказка
5	Фэнтези
6	Исторический
7	Поэма
8	Баллада
9	Эпопея

Продолжение приложения А

Продолжение таблицы А.6

1	2
10	Драма
11	Научная фантастика
12	Антиутопия

Таблица А.7 – Жанр - книга

ID связи	ID жанра	ID книги
1	2	3
1	3	1
4	7	3
5	3	4
6	4	5
7	8	6
8	3	7
9	3	8
10	3	9
11	9	9
12	2	10
13	10	10

Продолжение приложения А

Продолжение таблицы А.7

1	2	3
14	3	11
15	7	11
16	2	12
17	11	13
18	12	13
19	3	13
20	3	14
21	5	14
62	1	2
63	2	2
64	3	2
65	6	2

Таблица А.8 – Список пользователей

ID пользователя	Логин	Пароль	Имя	Дата регистрации	ID роли
1	2	3	4	5	6
1	neverket@gmail.com	123456	Arslan	2023-01-01 00:00:00	2

Продолжение приложения А

Продолжение таблицы А.8

1	2	3	4	5	6
2	neverket2@gmail.com	123456	Aminov	2023-01-02 00:00:00	1
3	0@gmail.com	123456	Jane Martinez	2023-01-03 00:00:00	1
4	1@gmail.com	123456	Eugene Gross	2023-01-03 00:00:00	1
5	2@gmail.com	123456	Leon Hernandez	2023-01-03 00:00:00	1
6	3@gmail.com	123456	Linda Delgado	2023-01-03 00:00:00	1
7	4@gmail.com	123456	David Myers	2023-01-03 00:00:00	1
8	5@gmail.com	123456	Virgil Lopez	2023-01-04 00:00:00	1
9	6@gmail.com	123456	Mark Diaz	2023-01-04 00:00:00	1
10	7@gmail.com	123456	Brian Hernandez	2023-01-04 00:00:00	1
11	8@gmail.com	123456	Charles Adams	2023-01-05 00:00:00	1
12	9@gmail.com	123456	Lois Johnson	2023-01-05 00:00:00	1
13	10@gmail.com	123456	James Baker	2023-01-05 00:00:00	1

Таблица А.9 – Список меток

ID роли	Название
1	2
1	Прочитать позже
2	Читаю

Продолжение приложения А

Продолжение таблицы А.9

1	2
3	Прочитано
4	Заброшено

Таблица А.10 – Список рецензий

ID рецензии	ID пользователя	ID книги	Дата создания	Оценка	Контент
1	2	3	4	5	6
1	3	1	2023-03-11 22:11:00	5	Онегин, добрый мой приятель... Писать рецензию на Евгения Онегина, после сотен тысяч страниц пушкиноведов, то еще занятие, но - сама, как гово...
2	4	1	2023-02-08 12:22:00	4	эта классика обошла меня стороной в школе, а на сочинении, скорее всего же оно было, не могло не быть, спасло то, что в музыкальной школе мы п...
3	5	1	2023-04-30 21:43:00	5	Прошла любовь - завяли помидоры (а она не прошла, но всё равно завяли). Впервые (если не забыла) читаю роман в стихотворном формате. В начале н...
4	7	1	2023-03-17 09:51:00	5	Подвиг русского поэта, роман в стихах: о жизни, о России, о любви и о себе. Евгений Онегин – юноша 18 лет из Санкт-Петербурга, “забав и роскоши...

Продолжение приложения А

Продолжение таблицы А.10

1	2	3	4	5	6
5	8	1	2023-03-23 23:40:00	5	На данный роман написано много разборов, много рецензий. И о его значении в мире литературы, и о проблемах, раскрытых тут, и о методах написа...
6	10	1	2022-12-01 18:28:00	5	Очередное знакомство с Онегиным Очень приятным оказалось очередное знакомство с Пушкиным и главной звездой - Онегиным. Многое со времен ш...
7	11	1	2022-12-22 16:35:00	4	Зная, как Александр Сергеевич относился к своим друзьям, так и хочется предположить, что Евгений Онегин и правда его приятель. Тем более, им...
8	12	1	2022-12-30 14:09:00	5	Нет, такие книги нужно читать и перечитывать, растаскивать на цитаты, запоминать наиболее важные строчки. Помнится, в школе девочки учили п...
9	1	14	2023-06-02 14:59:16	3	До чего же приятно вернуться к любимой истории спустя 20 лет! Удивительно, но при повторном прочтении книга оставила ровно те же эмоции, что ...
10	1	1	2023-06-03 14:04:02	5	Очень приятным оказалось очередное знакомство с Пушкиным и главной звездой - Онегиным. Многое со времен школы забыто было, на что-то и вним...

Продолжение приложения А

Продолжение таблицы А.10

1	2	3	4	5	6
11	3	2	2023-06-07 17:59:40	5	Когда честь дороже жизни. Произведение представляет собой семейную историю, написанную Петром Андреевичем Гринёвым. Картины детства в ро...
12	3	3	2023-06-07 18:01:01	5	Не ожидала что перечитывать классику в разном возрасте оказывается увлекательно). История о похищенной Людмиле и верном возлюбленном Рус...
13	3	7	2023-06-07 18:02:31	4	Конечно, в школе «Героя нашего времени» я читала. Эта книга подвергалась пристальному вниманию и анализу на уроках литературы, по ней писа...
14	3	8	2023-06-07 18:07:25	4	Разобранная на лоскутики, изученная по экранизациям, проштудированная по программным отрывкам великая книга в моей голове никогда не скл...
15	3	13	2023-06-07 18:08:16	4	Эта книга завирусилась довольно давно, но к сожалению, мои руки дошли прочитать её только сейчас. Задумка автора очень интересная и хорошо...
16	4	2	2023-06-07 18:09:47	5	Пушкин лёгким росчерком пера рисует самого себя в образе приезжего поэта-импровизатора и добавляет в текст повести собственные стихотво...

Продолжение приложения А

Продолжение таблицы А.10

1	2	3	4	5	6
17	4	3	2023-06-07 18:10:23	4	Это было не первое мое знакомство с книгой: кажется, в школьные годы мне пришлось читать эту поэму раза три или четыре (в начальной школе, в ...
18	4	4	2023-06-07 18:11:55	4	Читаю и перечитываю с дочкой её список на лето. И вот "Дубровский". Пушкин для меня не является любимым автором ни как поэт, ни как прозаик. В...
19	5	4	2023-06-07 18:13:08	4	Все-таки классику, которую проходят в школе, определенно нужно перечитывать в сознательном возрасте. Помню, в школе нам дали задание нап...
20	5	6	2023-06-07 18:14:47	5	Дочери задали сочинение по стихотворению "Бородино", пришлось вспоминать. Получилось примерно вот что. Даже интересно, какую оценку нам по...
21	5	5	2023-06-07 18:14:56	5	Вечная классика, которая уходит корнями в народный фольклор и откликается в самых разных национальных сказках. Что всегда удивляет меня в...
22	5	7	2023-06-07 18:16:10	5	На школьных уроках "Героя нашего времени" часто величают первым в истории русской литературы социально-психологическим романом. Что ж, ...

Продолжение приложения А

Продолжение таблицы А.10

1	2	3	4	5	6
23	5	9	2023-06-07 18:16:35	5	роман-эпопея «война и мир» стоит внимания каждого. его объем теряется на фоне сюжета, глубины героев и, конечно, на фоне тех проблем, которы...
24	5	10	2023-06-07 18:17:17	3	Николай Васильевич Гоголь «Шинель» Я решила перечитать «Шинель», потому что она упоминалась в недавно прочитанных «Бедных людях». Захо...
25	5	11	2023-06-07 18:17:39	3	Честно, когда училась в школе, совершенно не хотелось читать ничего из программы. Наконец, я пришла к тому, чтобы по желанию заняться изуче...
26	5	12	2023-06-07 18:17:56	5	Роман, написанный украинским писателем Николаем Гоголем и опубликованный в 1835 году, стал одним из наиболее известных произведений автора...
27	5	13	2023-06-07 18:18:23	4	Такие книги, конечно, нужно читать в юности. Или просто 20-30 лет назад. Искушенный мозг не может адекватно воспринимать написанное. В далеко...
28	5	14	2023-06-07 18:18:50	4	Книгу прочитала на одном дыхании. Для себя вынесла несколько идей: 1. Стремись к лучшему; 2. Ты всё равно станешь тем, кем должен быть, нес...

Продолжение приложения А

Продолжение таблицы А.10

1	2	3	4	5	6
29	6	2	2023-06-07 18:21:25	5	Классика - есть классика. Хотя многие и сравнивают Пиковую даму с Преступлением и наказанием в сокращении, для меня это что-то особенное. ...
30	6	4	2023-06-07 18:22:50	5	Моя первая мысль после прочтения романа: как будто прочтала о русском Робин Гуде. О благородном разбойнике, который мечтал отомстить чел...
31	6	6	2023-06-07 18:23:33	5	Книга "Бородино" Михаила Лермонтова относится к патриотическим стихотворениям. Помнится мне как всех школьников приходилось заставлять ...
32	6	7	2023-06-07 18:24:00	5	Прочитав эту книгу после школы, с новыми мозгами - это нечто. Воспринимается иначе, подчеркиваешь в каждой главе что то близкое для себя и ...
33	6	9	2023-06-07 18:24:31	5	война и мир это великая книга. В ней Толстой смог объединить все виды русской прозы, Как и роман, трагедия, эпос. Хорошее описание личностей...
34	6	11	2023-06-07 18:25:11	5	Классика о людских пороках, которую я люблю анализировать больше, чем читать. В школе я так и не прочла это произведение полностью. Решение...

Продолжение приложения А

Продолжение таблицы А.10

1	2	3	4	5	6
35	6	13	2023-06-07 18:25:52	5	В романе описывается мир будущего, в котором все печатные издания сжигаются, а хранение книг преследуется законом. Присутствует интеракт...
36	6	14	2023-06-07 18:26:17	4	Мое мнение: Я не думаю, что есть смысл описывать сюжет Поттера, большинство из нас знакомы с ним очень и очень давно. Фильм хочу заметить сн...
37	7	3	2023-06-07 18:27:34	5	Вечная классика, которая уходит корнями в народный фольклор и откликается в самых разных национальных сказках. Что всегда удивляет меня в...
38	7	6	2023-06-07 18:28:25	1	мне понравился этот стих тем что автор хорошо описывает то что было.
39	7	7	2023-06-07 18:28:55	5	Роман состоит из нескольких частей, хронологический порядок которых нарушен. Такое расположение служит особым художественным задачам: с...
40	8	12	2023-06-07 18:30:43	4	Сам текст просто восхитительный. Думаю не найдется ни одного человека, который при чтении не захотел бы оказаться в прекрасной деревеньке...

Продолжение приложения А

Продолжение таблицы А.10

1	2	3	4	5	6
41	8	9	2023-06-07 18:31:20	5	Итак, я добралась до романа "Война и мир". роман у меня в двух книгах (по два тома в каждой) , поэтому отзыва будет два . В общем, у нас тут ист...
42	8	2	2023-06-07 18:31:54	4	Если в "Барышне-крестьянке" вражда семейств и соседей заканчивается благополучно, то здесь из малейшей и глупейшей искры разгорается настоящий пожар вражды - и перерастает в разбой, сумятицу и волнения. И главное, попытки примирения были, но ни к чему хорошему не привели, лишь к ещё большему горю и даже губительному исходу. В итоге молодой Дубровский окончательно становится разбойником - но вскоре получает шанс на исправление. Он влюбляется не в кого-нибудь, а в дочь врага, и меняет личину, чтобы втереться в доверие.

Таблица А.11 – Список заметок

ID заметки	ID пользователя	Дата создания	Название	Контент
1	2	2023-06-01 20:42:54	Без названия	
2	2	2023-06-01 20:40:21	Заметка 1	
3	2	2023-06-01 20:43:36	Тест	Текст заметки

Таблица А.12 – Список лайков

ID лайка	ID рецензии	ID пользователя	Дата добавления
1	1	1	2023-05-03 18:38:05
2	2	1	2023-05-03 15:04:33
4	2	2	2023-05-04 01:33:24
5	1	3	2023-05-04 01:42:40
6	3	2	2023-05-22 10:25:15
7	3	1	2023-06-01 20:57:35
8	10	1	2023-06-03 14:04:24
9	42	2	2023-06-07 21:32:51

Приложение Б

Исходный код

Program.cs

```
using DinkToPdf.Contracts;
using DinkToPdf;

var builder = WebApplication.CreateBuilder(args);

// Add services to the container.
builder.Services.AddControllersWithViews();
builder.Services.AddSession();

builder.Services.AddSingleton(typeof(IConverter),
    new SynchronizedConverter(new PdfTools()));

var app = builder.Build();

// Configure the HTTP request pipeline.
if (!app.Environment.IsDevelopment()) {
    app.UseExceptionHandler("/Home/Error");
    // The default HSTS value is 30 days. You may want to change this for production scenarios, see
    https://aka.ms/aspnetcore-hsts.
    app.UseHsts();
}

app.UseHttpsRedirection();
app.UseStaticFiles();

app.UseRouting();

app.UseSession();
app.UseAuthentication();
app.UseAuthorization();

app.MapControllerRoute(
```

Продолжение приложения Б

```
name: "default",
pattern: "{controller=Home}/{action=Index}/{id?}");

app.Run();

    Controllers/AuthController.cs

using Lib.Models;
using Microsoft.AspNetCore.Mvc;

namespace Lib.Controllers {
    public class AuthController : Controller {

        [HttpGet("login")]
        public ActionResult Login(string? message) {
            if (message != null) {
                ViewBag.message = message;
            }
            return View();
        }

        // POST:
        [HttpPost("login")]
        [ValidateAntiForgeryToken]
        public ActionResult Login(string login, string password) {
            User user = LibDbContext.Instance.Users.FirstOrDefault(u => u.Login == login &&
u.Password == password);
            if (user != null) {
                HttpContext.Session.SetInt32("userId", user.Id);
                HttpContext.Session.SetString("userName", user.Name);
                if (UserController.IsCurrentUserAdmin(user)) {
                    HttpContext.Session.SetString("userIsAdmin", "true");
                }
                Console.WriteLine(HttpContext.Session.GetInt32("userId"));
                return RedirectToAction("Index", "Home", new { welcome = true });
            }
        }
    }
}
```

Продолжение приложения Б

```
        return RedirectToAction("Login", new { message = "Неверные данные для входа"
    });
}

// POST:
[HttpPost("logout")]
[ValidateAntiForgeryToken]
public ActionResult Logout() {
    int? userId = HttpContext.Session.GetInt32("userId");
    if (userId.HasValue) {
        HttpContext.Session.Remove("userId");
        HttpContext.Session.Remove("userName");
        HttpContext.Session.Remove("userIsAdmin");
    }
    return RedirectToAction("Login", new { message = "Произведен выход из
аккаунта" });
}

[HttpGet("register")]
public ActionResult Register(string? message) {
    if (message != null) {
        ViewBag.message = message;
    }
    return View();
}

// POST:
[HttpPost("register")]
[ValidateAntiForgeryToken]
public async Task<ActionResult> Register(
    string login,
    string password1,
    string password2,
    string name) {
    if (!password1.Equals(password2)) {
```

Продолжение приложения Б

```
        return RedirectToAction("Register", new { message = "Пароли не
совпадают" });
    }
    if (LibDbContext.Instance.Users.Where(u => u.Login == login).Any()) {
        return RedirectToAction("Register", new { message = "Пользователь с
таким логином уже есть" });
    }
    User user = new User {
        Id = LibDbContext.Instance.Users.OrderBy(u => u.Id).Last().Id + 1,
        Login = login,
        Password = password1,
        Name = name,
        DateOfRegistration = DateTime.Now,
        RoleId = 1
    };
    LibDbContext.Instance.Users.Add(user);
    await LibDbContext.Instance.SaveChangesAsync();

    HttpContext.Session.SetInt32("userId", user.Id);
    HttpContext.Session.SetString("userName", user.Name);
    if (UserController.IsCurrentUserAdmin(user)) {
        HttpContext.Session.SetString("userIsAdmin", "true");
    }
    Console.WriteLine(HttpContext.Session.GetInt32("userId"));
    return RedirectToAction("Index", "Home", new { welcome = true });
}
}
}
```

Controllers/AuthorController.cs

```
using Lib.Models;
using Microsoft.AspNetCore.Mvc;
using Microsoft.EntityFrameworkCore;

namespace Lib.Controllers {
```


Продолжение приложения Б

```
[Route("author")]
public class AuthorController : Controller {

    private const int COUNT_OF_SAVED_AUTHORS = 10;

    // GET:
    [HttpGet("")]
    [HttpGet("all")]
    public ActionResult All() {
        List<Author> authors = LibDbContext.Instance.Authors.OrderBy(a =>
a.Name).ToList();
        ViewBag.authors = authors;
        User user = UserController.getCurrentUser(HttpContext);
        if (user != null) {
            ViewBag.user = user;
            ViewBag.IsAdmin = UserController.isCurrentUserAdmin(user);
        }
        return View();
    }

    // GET:
    [HttpGet("{id:int}")]
    public IActionResult One(int id) {
        Author author = LibDbContext.Instance.Authors.FirstOrDefault(a => a.Id == id);
        if (author == null) {
            return RedirectToAction("All", "Author");
        }
        List<Book> books = new List<Book>();
        List<AuthorBook> authorBooks = LibDbContext.Instance.AuthorBooks.Where(ab
=> ab.AuthorId == id).ToList();
        foreach (AuthorBook authorBook in authorBooks) {
            books.Add(LibDbContext.Instance.Books
                .Include(b => b.AuthorBooks)
                .ThenInclude(ab => ab.Author)
                .Include(b => b.FeaturedBooks)
```

Продолжение приложения Б

```
        .FirstOrDefault(b => b.Id == authorBook.BookId));
    }
    ViewBag.author = author;
    ViewBag.newBooks = books
        .OrderByDescending(b => b.Id)
        .Take(4).ToList();
    ViewBag.popularBooks = books
        .OrderByDescending(b => b.FeaturedBooks.Count)
        .Take(4).ToList();
    SaveAuthorToSession(author);
    User user = UserController.getCurrentUser(HttpContext);
    if (user != null) {
        ViewBag.user = user;
        ViewBag.IsAdmin = UserController.isCurrentUserAdmin(user);
    }
    return View();
}

// GET:
[HttpGet("create")]
public IActionResult Create(string? message) {
    if (message != null) {
        ViewBag.message = message;
    }
    User user = UserController.getCurrentUser(HttpContext);
    if (user != null && UserController.isCurrentUserAdmin(user)) {
        return View("~/Views/Author/Update.cshtml");
    }
    return RedirectToAction("Login", "Auth");
}

// POST:
[HttpPost("delete")]
public async Task<ActionResult> Delete(int id) {
    User user = UserController.getCurrentUser(HttpContext);
```

Продолжение приложения Б

```
if (user != null && UserController.isCurrentUserAdmin(user)) {
    Author author = LibDbContext.Instance.Authors
        .Include(a => a.AuthorBooks)
        .FirstOrDefault(a => a.Id == id);
    if (author != null) {
        foreach (var ab in author.AuthorBooks) {
            LibDbContext.Instance.AuthorBooks.Remove(ab);
        }
        LibDbContext.Instance.Authors.Remove(author);
        await LibDbContext.Instance.SaveChangesAsync();
    }
    return RedirectToAction("All", "Author");
}
return RedirectToAction("Login", "Auth");
}

// GET:
[HttpGet("{id:int}/edit")]
public IActionResult Edit(int id, string? message) {
    if (message != null) {
        ViewBag.message = message;
    }
    User user = UserController.getCurrentUser(HttpContext);
    if (user != null && UserController.isCurrentUserAdmin(user)) {
        Author author = LibDbContext.Instance.Authors.FirstOrDefault(a => a.Id
== id);

        Console.WriteLine("author get edit = " + id.ToString());
        ViewBag.author = author;
        return View("~/Views/Author/Update.cshtml");
    }
    return RedirectToAction("Login", "Auth");
}

// POST:
[HttpPost("update")]
```

Продолжение приложения Б

```
public async Task<ActionResult> Update(int id, string name, string photo, string biography)
{

    User user = UserController.getCurrentUser(HttpContext);
    if (user != null && UserController.isCurrentUserAdmin(user)) {
        var last = LibDbContext.Instance.Authors.OrderBy(n =>
n.Id).LastOrDefault();

        Author author = null;
        if (id >= 1) {
            author = LibDbContext.Instance.Authors
                .Include(a => a.AuthorBooks)
                .ThenInclude(ab => ab.Book)
                .FirstOrDefault(a => a.Id == id);
            author.Name = name;
            author.Photo = photo;
            author.Biography = biography;
            LibDbContext.Instance.Authors.Update(author);
        } else {
            author = new Author() {
                Id = last != null ? (last.Id + 1) : 0,
                Name = name,
                Photo = photo,
                Biography = biography
            };
            LibDbContext.Instance.Authors.Add(author);
        }
        await LibDbContext.Instance.SaveChangesAsync();
        return RedirectToAction("One", new { id });
    }
    return RedirectToAction("Login", "Auth");
}

private void SaveAuthorToSession(Author author) {
    if (author == null) { return; }
    List<int> savedIds = new List<int>();
```

Продолжение приложения Б

```
for (int i = 0; i < COUNT_OF_SAVED_AUTHORS; i++) {

    int? savedBookId = HttpContext.Session.GetInt32($"savedAuthor_{i}_id");
    if (!savedBookId.HasValue) {
        break;
    }
    savedIds.Add(savedBookId.Value);
}
if (savedIds.Contains(author.Id)) {
    savedIds.Remove(author.Id); // удалить в середине
}
savedIds.Insert(0, author.Id); // добавить в начало
if ((savedIds.Count - 1) == COUNT_OF_SAVED_AUTHORS) {
    savedIds.RemoveAt(savedIds.Count - 1);
}
for (int i = 0; i < savedIds.Count; i++) {
    HttpContext.Session.SetInt32($"savedAuthor_{i}_id", savedIds[i]);
}
}
}
```

Controllers/BookController.cs

```
using Lib.Models;
using Microsoft.AspNetCore.Mvc;
using Microsoft.EntityFrameworkCore;

namespace Lib.Controllers {
    [Route("book")]
    public class BookController : Controller {

        private const int COUNT_OF_SAVED_BOOKS = 10;

        // GET: BookController
        [HttpGet("")]
```

Продолжение приложения Б

```
[HttpGet("all")]
public ActionResult All() {
    ViewBag.books = LibDbContext.Instance.Books.ToList();

    User user = UserController.getCurrentUser(HttpContext);
    if (user != null) {
        ViewBag.user = user;
        ViewBag.IsAdmin = UserController.isCurrentUserAdmin(user);
    }

    return View();
}

// GET:
[HttpGet("{id:int}")]
public IActionResult One(int id, string? message = null) {
    Console.WriteLine("book get one = " + id.ToString());
    Book book = LibDbContext.Instance.Books
        .Include(b => b.AuthorBooks)
        .ThenInclude(ab => ab.Author)
        .Include(b => b.GenreBooks)
        .ThenInclude(ab => ab.Genre)
        .Include(b => b.FeaturedBooks)
        .ThenInclude(ab => ab.Mark)
        .Include(b => b.FeaturedBooks)
        .ThenInclude(ab => ab.User)
        .Include(b => b.Reviews)
        .ThenInclude(ab => ab.User)
        .Include(b => b.Reviews)
        .ThenInclude(ab => ab.Likes)
        .ThenInclude(ab => ab.User)
        .FirstOrDefault(b => b.Id == id);
    if (book == null) {
        return RedirectToAction("All", "Book");
    }
}
```

Продолжение приложения Б

```
ViewBag.message = message;
ViewBag.book = book;
ViewBag.authors = book.AuthorBooks.Select(ab => ab.Author).ToList();
ViewBag.genres = book.GenreBooks.Select(ab => ab.Genre).ToList();
ViewBag.reviews = book.Reviews.OrderByDescending(b =>
b.DateOfCreation).ToList();
ViewBag.marks = LibDbContext.Instance.Marks.ToList();
SaveBookToSession(book);
User user = UserController.getCurrentUser(HttpContext);
if (user != null) {
    ViewBag.user = user;
    ViewBag.IsAdmin = UserController.isCurrentUserAdmin(user);
}
return View();
}

// GET:
[HttpGet("create")]
public IActionResult Create(string? message) {
    if (message != null) {
        ViewBag.message = message;
    }
    User user = UserController.getCurrentUser(HttpContext);
    if (user != null && UserController.isCurrentUserAdmin(user)) {
        ViewBag.genres = LibDbContext.Instance.Genres.ToList();
        ViewBag.authors = LibDbContext.Instance.Authors.ToList();
    }

    return View("~/Views/Book/Update.cshtml");
}

return RedirectToAction("Login", "Auth");
}

// POST:
[HttpPost("delete")]
public async Task<ActionResult> Delete(int id) {
```

Продолжение приложения Б

```
User user = UserController.getCurrentUser(HttpContext);
if (user != null && UserController.IsCurrentUserAdmin(user)) {
    Book book = LibDbContext.Instance.Books
        .Include(b => b.AuthorBooks)
        .ThenInclude(ab => ab.Author)
        .Include(b => b.GenreBooks)
        .ThenInclude(ab => ab.Genre)
        .Include(b => b.FeaturedBooks)
        .ThenInclude(ab => ab.Mark)
        .Include(b => b.Reviews)
        .ThenInclude(ab => ab.User)
        .Include(b => b.Reviews)
        .ThenInclude(ab => ab.Likes)
        .ThenInclude(ab => ab.User)
        .FirstOrDefault(b => b.Id == id);
    if (book != null) {
        foreach (var n in book.AuthorBooks) {
            LibDbContext.Instance.AuthorBooks.Remove(n);
        }
        foreach (var n in book.GenreBooks) {
            LibDbContext.Instance.GenreBooks.Remove(n);
        }
        foreach (var n in book.FeaturedBooks) {
            LibDbContext.Instance.FeaturedBooks.Remove(n);
        }
        foreach (var r in book.Reviews) {
            foreach (var l in r.Likes) {
                LibDbContext.Instance.Likes.Remove(l);
            }
            LibDbContext.Instance.Reviews.Remove(r);
        }
        LibDbContext.Instance.Books.Remove(book);
        await LibDbContext.Instance.SaveChangesAsync();
    }
    return RedirectToAction("All", "Book");
}
```


Продолжение приложения Б

```
    }
    return RedirectToAction("Login", "Auth");
}

// GET:
[HttpGet("{id:int}/edit")]
public IActionResult Edit(int id, string? message) {
    if (message != null) {
        ViewBag.message = message;
    }
    User user = UserController.GetCurrentUser(HttpContext);
    if (user != null && UserController.IsCurrentUserAdmin(user)) {
        Book book = LibDbContext.Instance.Books
            .Include(n => n.GenreBooks)
            .Include(n => n.AuthorBooks)
            .FirstOrDefault(b => b.Id == id);
        Console.WriteLine("book get edit = " + id.ToString());
        ViewBag.book = book;
        ViewBag.genre_ids = book.GenreBooks.Select(n => n.GenreId).ToList();
        ViewBag.author_ids = book.AuthorBooks.Select(n => n.AuthorId).ToList();
        ViewBag.genres = LibDbContext.Instance.Genres.ToList();
        ViewBag.authors = LibDbContext.Instance.Authors.ToList();
        return View("~/Views/Book/Update.cshtml");
    }
    return RedirectToAction("Login", "Auth");
}

// POST:
[HttpPost("update")]
public async Task<ActionResult> Update(int id, string name, string description, string
photo,
                                string date_of_creation, string[] genre_ids, string[] author_ids) {
    Console.WriteLine(genre_ids.Length + " lol " + string.Join(", ", genre_ids));
    User user = UserController.GetCurrentUser(HttpContext);
    if (user != null && UserController.IsCurrentUserAdmin(user)) {
```

Продолжение приложения Б

```
var last = LibDbContext.Instance.Books.OrderBy(n =>
n.Id).LastOrDefault();

Book book = null;
if (id >= 1) {
    book = LibDbContext.Instance.Books
        .Include(n => n.GenreBooks)
        .ThenInclude(n => n.Genre)
        .Include(n => n.AuthorBooks)
        .ThenInclude(n => n.Author)
        .FirstOrDefault(b => b.Id == id);
    foreach (GenreBook gb in book.GenreBooks) {
        LibDbContext.Instance.GenreBooks.Remove(gb);
    }

    foreach (var genreIdStr in genre_ids) {
        if (int.TryParse(genreIdStr, out int genreId)) {
            GenreBook gb = new GenreBook() {
                BookId = book.Id,
                GenreId = genreId
            };
            LibDbContext.Instance.GenreBooks.Add(gb);
        }
    }

    foreach (AuthorBook ab in book.AuthorBooks) {
        LibDbContext.Instance.AuthorBooks.Remove(ab);
    }

    foreach (var authorIdStr in author_ids) {
        if (int.TryParse(authorIdStr, out int authorId)) {
            AuthorBook ab = new AuthorBook() {
                BookId = book.Id,
                AuthorId = authorId
            };
            LibDbContext.Instance.AuthorBooks.Add(ab);
        }
    }

    book.Name = name;
```

Продолжение приложения Б

```
        book.Description = description;
        book.Photo = photo;
        book.DateOfCreation = date_of_creation;
        LibDbContext.Instance.Books.Update(book);
    } else {
        book = new Book() {
            Id = last != null ? (last.Id + 1) : 0,
            Name = name,
            Description = description,
            Photo = photo,
            DateOfCreation = date_of_creation
        };

        LibDbContext.Instance.Books.Add(book);
        foreach (var genreIdStr in genre_ids) {
            if (int.TryParse(genreIdStr, out int genreId)) {
                GenreBook gb = new GenreBook() {
                    BookId = book.Id,
                    GenreId = genreId
                };
                LibDbContext.Instance.GenreBooks.Add(gb);
            }
        }
        foreach (var authorIdStr in author_ids) {
            if (int.TryParse(authorIdStr, out int authorId)) {
                AuthorBook ab = new AuthorBook() {
                    BookId = book.Id,
                    AuthorId = authorId
                };
                LibDbContext.Instance.AuthorBooks.Add(ab);
            }
        }

        await LibDbContext.Instance.SaveChangesAsync();
        return RedirectToAction("One", new { id });
    }
}
```

Продолжение приложения Б

```
        return RedirectToAction("Login", "Auth");
    }

    private void SaveBookToSession(Book book) {
        List<int> savedIds = new List<int>();
        for (int i = 0; i < COUNT_OF_SAVED_BOOKS; i++) {
            int? savedBookId = HttpContext.Session.GetInt32($"savedBook_{i}_id");
            if (!savedBookId.HasValue) {
                break;
            }
            savedIds.Add(savedBookId.Value);
        }
        if (savedIds.Contains(book.Id)) {
            savedIds.Remove(book.Id); // удалить в середине
        }
        savedIds.Insert(0, book.Id); // добавить в начало
        if ((savedIds.Count - 1) == COUNT_OF_SAVED_BOOKS) {
            savedIds.RemoveAt(savedIds.Count - 1);
        }
        for (int i = 0; i < savedIds.Count; i++) {
            HttpContext.Session.SetInt32($"savedBook_{i}_id", savedIds[i]);
        }
    }
}
```

Controllers/FeaturedController.cs

```
using Lib.Models;
using Microsoft.AspNetCore.Mvc;
using Microsoft.EntityFrameworkCore;

namespace Lib.Controllers {
    [Route("featured")]
    public class FeaturedController : Controller {
```

Продолжение приложения Б

```
private User getCurrentUser() {
    int? userId = HttpContext.Session.GetInt32("userId");
    if (userId.HasValue) {
        User user = LibDbContext.Instance.Users
            .Include(u => u.Role)
            .FirstOrDefault(u => u.Id == userId);
        return user;
    }
    return null;
}

// GET:
[HttpGet("")]
[HttpGet("all")]
public ActionResult All() {
    User user = getCurrentUser();
    if (user != null) {
        List<FeaturedBook> featuredBooks =
LibDbContext.Instance.FeaturedBooks
            .Include(fb => fb.Mark)
            .Include(fb => fb.Book)
            .Where(fb => fb.UserId == user.Id).ToList();
        ViewBag.user = user;
        ViewBag.featuredBooks = featuredBooks;
        return View();
    }
    return RedirectToAction("Login", "Auth");
}

// POST:
[HttpPost("one")]
[ValidateAntiForgeryToken]
public async Task<ActionResult> One(int book_id, int mark_id) {
    int? userId = HttpContext.Session.GetInt32("userId");
    if (userId.HasValue) {
```

Продолжение приложения Б

```
FeaturedBook fb = LibDbContext.Instance.FeaturedBooks
    .FirstOrDefault(fb=> fb.UserId == userId && fb.BookId ==
book_id);

if (fb == null) {
    var last = LibDbContext.Instance.FeaturedBooks.OrderBy(n=>
n.Id).LastOrDefault();

    fb = new FeaturedBook {
        Id = last != null ? (last.Id + 1) : 0,
        UserId = userId.Value,
        BookId = book_id,
        DateOfAdd = DateTime.Now,
        MarkId = mark_id
    };
    LibDbContext.Instance.FeaturedBooks.Add(fb);
} else {
    if (mark_id == 0) {
        LibDbContext.Instance.FeaturedBooks.Remove(fb);
    } else {
        fb.MarkId = mark_id;
        fb.DateOfAdd = DateTime.Now;
        LibDbContext.Instance.FeaturedBooks.Update(fb);
    }
}

await LibDbContext.Instance.SaveChangesAsync();
return Redirect($"{Url.Action("One", "Book", new { id = book_id })}");
}

return RedirectToAction("Login", "Auth");
}
}
}
```

Controllers/GenreController.cs

```
using Lib.Models;
using Microsoft.AspNetCore.Mvc;
using Microsoft.EntityFrameworkCore;
```

Продолжение приложения Б

```
namespace Lib.Controllers {
    [Route("genre")]
    public class GenreController : Controller {

        //[Route("~/genres")]
        [HttpGet("")]
        [HttpGet("all")]
        public IActionResult All() {
            ViewBag.genres = LibDbContext.Instance.Genres.OrderBy(g => g.Name).ToList();
            User user = UserController.getCurrentUser(HttpContext);
            if (user != null) {
                ViewBag.user = user;
                ViewBag.IsAdmin = UserController.isCurrentUserAdmin(user);
            }
            return View();
        }

        [HttpGet("{id:int}")]
        public IActionResult One(int id) {
            Console.WriteLine("genre get one = " + id.ToString());
            List<Genre> genres = LibDbContext.Instance.Genres.OrderBy(g =>
g.Name).ToList();
            ViewBag.genres = genres;
            Genre genre = genres.FirstOrDefault(g => g.Id == id);
            if (genre == null) {
                return RedirectToAction("All", "Genre");
            }
            ViewBag.genre = genre;
            List<Book> books = new List<Book>();
            List<GenreBook> genreBooks = LibDbContext.Instance.GenreBooks
                .Include(gb => gb.Book)
                .ThenInclude(b => b.FeaturedBooks)
                .Where(gb => gb.GenreId == id).ToList();
            foreach (GenreBook genreBook in genreBooks) {
                books.Add(genreBook.Book);
            }
        }
    }
}
```

Продолжение приложения Б

```
}
ViewBag.popularBooks = books.OrderByDescending(b => b.FeaturedBooks.Count).ToList();
ViewBag.newBooks = books.OrderByDescending(b => b.Id).ToList();
    User user = UserController.getCurrentUser(HttpContext);
    if (user != null) {
        ViewBag.user = user;
        ViewBag.IsAdmin = UserController.isCurrentUserAdmin(user);
    }
    return View();
}

// GET:
[HttpGet("create")]
public IActionResult Create(string? message) {
    if (message != null) {
        ViewBag.message = message;
    }
    User user = UserController.getCurrentUser(HttpContext);
    if (user != null && UserController.isCurrentUserAdmin(user)) {
        return View("~/Views/Genre/Update.cshtml");
    }
    return RedirectToAction("Login", "Auth");
}

// POST:
[HttpPost("delete")]
public async Task<ActionResult> Delete(int id) {
    User user = UserController.getCurrentUser(HttpContext);
    if (user != null && UserController.isCurrentUserAdmin(user)) {
        Genre genre = LibDbContext.Instance.Genres
            .Include(g => g.GenreBooks)
            .FirstOrDefault(g => g.Id == id);
        if (genre != null) {
            foreach (var gb in genre.GenreBooks) {
                LibDbContext.Instance.GenreBooks.Remove(gb);
            }
        }
    }
}
```


Продолжение приложения Б

```
    }
    LibDbContext.Instance.Genres.Remove(genre);
    await LibDbContext.Instance.SaveChangesAsync();
}
return RedirectToAction("All", "Genre");
}
return RedirectToAction("Login", "Auth");
}

// GET:
[HttpGet("{id:int}/edit")]
public IActionResult Edit(int id, string? message) {
    if (message != null) {
        ViewBag.message = message;
    }
    User user = UserController.getCurrentUser(HttpContext);
    if (user != null && UserController.isCurrentUserAdmin(user)) {
        Genre genre = LibDbContext.Instance.Genres.FirstOrDefault(g => g.Id == id);
        Console.WriteLine("genre get edit = " + id.ToString());
        ViewBag.genre = genre;
        return View("~/Views/Genre/Update.cshtml");
    }
    return RedirectToAction("Login", "Auth");
}

// POST:
[HttpPost("update")]
public async Task<ActionResult> Update(int id, string name) {
    User user = UserController.getCurrentUser(HttpContext);
    if (user != null && UserController.isCurrentUserAdmin(user)) {
        var last = LibDbContext.Instance.Genres.OrderBy(n => n.Id).LastOrDefault();
        Genre genre = null;
        if (id >= 1) {
            genre = LibDbContext.Instance.Genres
                .Include(g => g.GenreBooks)
```

Продолжение приложения Б

```
        .ThenInclude(gb => gb.Book)
        .FirstOrDefault(g => g.Id == id);
    genre.Name = name;
    LibDbContext.Instance.Genres.Update(genre);
    await LibDbContext.Instance.SaveChangesAsync();
    return RedirectToAction("One", new { id });
} else {
    genre = new Genre() {
        Id = last != null ? (last.Id + 1) : 0,
        Name = name
    };
    LibDbContext.Instance.Genres.Add(genre);
    await LibDbContext.Instance.SaveChangesAsync();
    return RedirectToAction("All", "Genre");
}
}
return RedirectToAction("Login", "Auth");
}
}
}
```

Controllers/HomeController.cs

```
using Lib.Models;
using Microsoft.AspNetCore.Mvc;
using Microsoft.EntityFrameworkCore;
using System.Diagnostics;

namespace Lib.Controllers {
    public class HomeController : Controller {
        private readonly ILogger<HomeController> _logger;

        public HomeController(ILogger<HomeController> logger) {
            _logger = logger;
        }
    }
}
```

Продолжение приложения Б

```
[ResponseCache(Duration = 0, Location = ResponseCacheLocation.None, NoStore = true)]
public IActionResult Error() {
    return View(new ErrorViewModel { RequestId = Activity.Current?.Id ?? HttpContext.TraceIdentifier
});
}

[HttpGet("")]
public IActionResult Index(bool? welcome) {
    ViewBag.popularBooks = LibDbContext.Instance.Books
.Include(b => b.AuthorBooks)
    .ThenInclude(ab => ab.Author)
.Include(b => b.FeaturedBooks)
    .OrderByDescending(b => b.FeaturedBooks.Count)
    .Take(10).ToList();
    ViewBag.newBooks = LibDbContext.Instance.Books
.Include(b => b.AuthorBooks)
    .ThenInclude(ab => ab.Author)
    .OrderByDescending(b => b.Id)
    .Take(10).ToList();
    ViewBag.bestReviews = LibDbContext.Instance.Reviews
.Include(r=>r.Book)
.Include(r=>r.Likes)
.Include(r=>r.User)
    .OrderByDescending(r => r.Likes.Count)
    .Take(10).ToList();
    ViewBag.showWelcome = welcome.HasValue;
    User user = UserController.getCurrentUser(HttpContext);
    if (user != null) {
        ViewBag.user = user;
        ViewBag.IsAdmin = UserController.isCurrentUserAdmin(user);
    }
    return View();
}

[HttpGet("about")]
```

Продолжение приложения Б

```
public IActionResult About() {
    return View();
}

[HttpGet("popular")]
public IActionResult Popular() {
    ViewBag.books = LibDbContext.Instance.Books
        .Include(b => b.FeaturedBooks)
        .OrderByDescending(b => b.FeaturedBooks.Count)
        .Take(10).ToList();

    ViewBag.title = "Популярное";
    ViewBag.ActivePopular = "active";
    ViewBag.ShowPopular = true;

    User user = UserController.getCurrentUser(HttpContext);
    if (user != null) {
        ViewBag.user = user;
        ViewBag.IsAdmin = UserController.isCurrentUserAdmin(user);
    }
    return View("~/Views/Book/All.cshtml");
}

[HttpGet("new")]
public IActionResult New() {
    ViewBag.books = LibDbContext.Instance.Books
        .OrderByDescending(b => b.Id)
        .Take(10).ToList();

    ViewBag.title = "Новинки";
    ViewBag.ActiveNew = "active";
    ViewBag.ShowPopular = false;

    User user = UserController.getCurrentUser(HttpContext);
    if (user != null) {
        ViewBag.user = user;
        ViewBag.IsAdmin = UserController.isCurrentUserAdmin(user);
    }
    return View("~/Views/Book/All.cshtml");
}
```

Продолжение приложения Б

```
}
```

```
[HttpGet("top")]
```

```
public IActionResult Top() {  
    int takeCount = 10;  
    ViewBag.books = LibDbContext.Instance.Books  
        .Where(b => b.AvgRating.HasValue)  
        .OrderByDescending(b => b.AvgRating)  
        .Take(takeCount).ToList();  
    ViewBag.title = $"Топ-{takeCount}";  
    ViewBag.isTop = true;  
    User user = UserController.getCurrentUser(HttpContext);  
    if (user != null) {  
        ViewBag.user = user;  
        ViewBag.IsAdmin = UserController.isCurrentUserAdmin(user);  
    }  
    return View("~/Views/Book/All.cshtml");  
}
```

```
[HttpGet("search")]
```

```
public IActionResult Search(string q, int? genre_id) {  
    if (string.IsNullOrEmpty(q)) {  
        return RedirectToAction("Index", "Home");  
    }  
    q = q.ToLower().Trim();  
    ViewBag.genres = LibDbContext.Instance.Genres.ToList();  
    List<Book> books = LibDbContext.Instance.Books  
        .Include(b => b.GenreBooks)  
        .Where(b => b.Name.Contains(q)).ToList();  
    if (genre_id.HasValue) {  
        if (genre_id.Value != 0) {  
            books = books.Where(b => b.GenreBooks.Select(gb =>  
gb.GenreId).Contains(genre_id.Value)).ToList();  
            ViewBag.genre_id = genre_id;
```

Продолжение приложения Б

```
        }  
    }  
  
    List<Author> authors = LibDbContext.Instance.Authors  
        .Where(a => a.Name.Contains(q)).ToList();  
    ViewBag.q = q;  
    ViewBag.books = books;  
    ViewBag.authors = authors;  
    return View();  
}  
}  
}
```

Controllers/NoteController.cs

```
using Lib.Models;  
using Microsoft.AspNetCore.Mvc;  
using Microsoft.EntityFrameworkCore;  
  
namespace Lib.Controllers {  
    [Route("note")]  
    public class NoteController : Controller {  
  
        private Note getNoteByIdFromCurrentUser(int noteId, int userId) {  
            return LibDbContext.Instance.Notes.FirstOrDefault(n => n.Id == noteId &&  
n.UserId == userId);  
        }  
  
        // GET:  
        [HttpGet("")]  
        [HttpGet("all")]  
        public ActionResult All() {  
            int? userId = HttpContext.Session.GetInt32("userId");  
            if (userId.HasValue) {  
                List<Note> notes = LibDbContext.Instance.Notes  
                    .Include(n => n.User)  
                    .Where(n => n.UserId == userId).ToList();  
            }  
        }  
    }  
}
```

Продолжение приложения Б

```
        notes.Reverse();
        ViewBag.notes = notes;
        return View();
    }
    return RedirectToAction("Login", "Auth");
}

// POST:
[HttpPost("create")]
[ValidateAntiForgeryToken]
public async Task<ActionResult> Create(ICollection collection) {
    int? userId = HttpContext.Session.GetInt32("userId");
    if (userId.HasValue) {
        var last = LibDbContext.Instance.Notes.OrderBy(n =>
n.Id).LastOrDefault();

        Note note = new Note {
            Id = last != null ? (last.Id + 1) : 0,
            UserId = userId.Value,
            DateOfCreation = DateTime.Now,
            Name = "Без названия"
        };
        LibDbContext.Instance.Notes.Add(note);
        await LibDbContext.Instance.SaveChangesAsync();
        return RedirectToAction("One", new { note.Id });
    }
    return RedirectToAction("All");
}

// POST:
[HttpPost]
[ValidateAntiForgeryToken]
public async Task<ActionResult> Delete(int id) {
    int? userId = HttpContext.Session.GetInt32("userId");
    if (userId.HasValue) {
        Note note = getNoteByIdFromCurrentUser(id, userId.Value);
```

Продолжение приложения Б

```
        if (note != null) {
            LibDbContext.Instance.Notes.Remove(note);
            await LibDbContext.Instance.SaveChangesAsync();
        }
    }
    return RedirectToAction("All");
}

// POST:
[HttpPost("{id:int}/edit")]
[ValidateAntiForgeryToken]
public async Task<ActionResult> Edit(int id, string newName, string newContent) {
    int? userId = HttpContext.Session.GetInt32("userId");
    if (userId.HasValue) {
        Note note = getNoteByIdFromCurrentUser(id, userId.Value);
        if (note != null) {
            if (newName == null) {
                newName = "Без названия";
            }
            note.Name = newName;
            note.Content = newContent;
            LibDbContext.Instance.Notes.Update(note);
            await LibDbContext.Instance.SaveChangesAsync();
        }
    }
    return Redirect($"{Url.Action("One", "Note", new { id = id, message =
"Сохранено" })}");
}

// GET:
[HttpGet("{id:int}")]
public IActionResult One(int id, string? message = null) {
    int? userId = HttpContext.Session.GetInt32("userId");
    if (userId.HasValue) {
        Note note = getNoteByIdFromCurrentUser(id, userId.Value);
```


Продолжение приложения Б

```
        if (note != null) {
            ViewBag.note = note;
            ViewBag.message = message;
            return View();
        }
    }
    return RedirectToAction("All");
}
}
```

Controllers/ReviewController.cs

```
using Lib.Models;
using Microsoft.AspNetCore.Mvc;
using System.Collections.Generic;

namespace Lib.Controllers {
    [Route("review")]
    public class ReviewController : Controller {

        // POST:
        [HttpPost("update")]
        [ValidateAntiForgeryToken]
        public async Task<ActionResult> Update(int book_id, int rating, string content) {
            if (string.IsNullOrEmpty(content)) {
                return Redirect($"{Url.Action("One", "Book", new { id = book_id, message
= "Текст рецензии не может быть пустым" })}");
            }
            int? userId = HttpContext.Session.GetInt32("userId");
            if (userId.HasValue) {
                Review review = LibDbContext.Instance.Reviews
                    .FirstOrDefault(r => r.UserId == userId && r.BookId == book_id);
                if (review == null) {
                    var last = LibDbContext.Instance.Reviews.OrderBy(n =>
n.Id).LastOrDefault();
```

Продолжение приложения Б

```
        review = new Review {
            Id = last != null ? (last.Id + 1) : 0,
            UserId = userId.Value,
            BookId = book_id,
            DateOfCreation = DateTime.Now,
            Rating = rating,
            Content = content
        };
        LibDbContext.Instance.Reviews.Add(review);
    } else {
        review.Rating = rating;
        review.Content = content;
        LibDbContext.Instance.Reviews.Update(review);
    }
    await LibDbContext.Instance.SaveChangesAsync();
    return Redirect($"{Url.Action("One", "Book", new { id = book_id })}");
}
return RedirectToAction("All");
}

// POST:
[HttpPost("delete")]
[ValidateAntiForgeryToken]
public async Task<ActionResult> Delete(int book_id) {
    int? userId = HttpContext.Session.GetInt32("userId");
    if (userId.HasValue) {
        Review review = LibDbContext.Instance.Reviews
            .FirstOrDefault(r => r.UserId == userId && r.BookId == book_id);
        foreach (var like in review.Likes) {
            LibDbContext.Instance.Likes.Remove(like);
        }
        LibDbContext.Instance.Reviews.Remove(review);
        await LibDbContext.Instance.SaveChangesAsync();
        return Redirect($"{Url.Action("One", "Book", new { id = book_id })}");
    }
}
```

Продолжение приложения Б

```
        return RedirectToAction("All");
    }

    // POST:
    [HttpPost("delete_by_admin")]
    [ValidateAntiForgeryToken]
    public async Task<ActionResult> DeleteByAdmin(int book_id, int user_id) {
        User user = UserController.GetCurrentUser(HttpContext);
        if (user != null && UserController.IsCurrentUserAdmin(user)) {
            Review review = LibDbContext.Instance.Reviews
                .FirstOrDefault(r => r.UserId == user_id && r.BookId == book_id);
            foreach (var like in review.Likes) {
                LibDbContext.Instance.Likes.Remove(like);
            }
            LibDbContext.Instance.Reviews.Remove(review);
            await LibDbContext.Instance.SaveChangesAsync();
            return Redirect($"{Url.Action("One", "Book", new { id = book_id })}");
        }
        return RedirectToAction("All");
    }

    // POST:
    [HttpPost("like")]
    [ValidateAntiForgeryToken]
    public async Task<ActionResult> Like(int review_id, int book_id) {
        int? userId = HttpContext.Session.GetInt32("userId");
        if (userId.HasValue) {
            var last = LibDbContext.Instance.Likes.OrderBy(n => n.Id).LastOrDefault();
            Like like = new Like {
                Id = last != null ? (last.Id + 1) : 0,
                ReviewId = review_id,
                UserId = userId.Value,
                Date = DateTime.Now
            };
            LibDbContext.Instance.Likes.Add(like);
        }
    }
}
```

Продолжение приложения Б

```
        await LibDbContext.Instance.SaveChangesAsync();
        return Redirect($"{Url.Action("One", "Book", new { id = book_id
    }}}#{review_id}");
    }
    return RedirectToAction("Login", "Auth");
}

// POST:
[HttpPost("unlike")]
[ValidateAntiForgeryToken]
public async Task<ActionResult> Unlike(int review_id, int book_id) {
    int? userId = HttpContext.Session.GetInt32("userId");
    if (userId.HasValue) {
        Like like = LibDbContext.Instance.Likes
            .FirstOrDefault(l => l.ReviewId == review_id && l.UserId ==
userId);
        if (like != null) {
            LibDbContext.Instance.Likes.Remove(like);
            await LibDbContext.Instance.SaveChangesAsync();
            return Redirect($"{Url.Action("One", "Book", new { id = book_id
    }}}#{review_id}");
        }
    }
    return RedirectToAction("Login", "Auth");
}
}
```

Controllers/UserController.cs

```
using DinkToPdf;
using Lib.Models;
using Microsoft.AspNetCore.Http;
using Microsoft.AspNetCore.Mvc;
using Microsoft.EntityFrameworkCore;
```

Продолжение приложения Б

```
namespace Lib.Controllers {
    [Route("user")]
    public class UserController : Controller {

        public static User getCurrentUser(HttpContext context) {
            int? userId = context.Session.GetInt32("userId");
            if (userId.HasValue) {
                User user = LibDbContext.Instance.Users
                    .Include(u => u.Likes)
                    .Include(u => u.Role)
                    .Include(u => u.Reviews)
                    .Include(u => u.Notes)
                    .Include(u => u.FeaturedBooks)
                    .FirstOrDefault(u => u.Id == userId);
                return user;
            }
            return null;
        }

        public static bool isCurrentUserAdmin(User user) {
            return user.RoleId == 2 ? true : false;
        }

        // GET:
        [HttpGet("{id:int}")]
        public IActionResult One(int id) {
            User user = getCurrentUser(HttpContext);
            if (user?.Id == id) {
                return RedirectToAction("Profile");
            }
            user = LibDbContext.Instance.Users
                .Include(u => u.Role)
                .Include(u => u.Reviews)
                .FirstOrDefault(u => u.Id == id);
            ViewBag.user = user;
        }
    }
}
```

Продолжение приложения Б

```
List<FeaturedBook> featuredBooks = LibDbContext.Instance.FeaturedBooks
    .Include(fb => fb.Mark)
    .Include(fb => fb.Book)
    .Where(fb => fb.UserId == user.Id).ToList();
ViewBag.featuredBooks = featuredBooks;
return View();
}

// GET:
[HttpGet("~/profile")]
public IActionResult Profile() {
    User user = getCurrentUser(HttpContext);
    if (user != null) {
        ViewBag.user = user;
        ViewBag.IsProfile = true;
        List<FeaturedBook> featuredBooks =
LibDbContext.Instance.FeaturedBooks
            .Include(fb => fb.Mark)
            .Include(fb => fb.Book)
            .Where(fb => fb.UserId == user.Id).ToList();
        ViewBag.featuredBooks = featuredBooks;
        return View("~/Views/User/One.cshtml");
    }
    return RedirectToAction("Login", "Auth");
}

// GET:
[HttpGet("~/profile/edit")]
public IActionResult Update(string? message) {
    if (message != null) {
        ViewBag.message = message;
    }
    User user = getCurrentUser(HttpContext);
    if (user != null) {
        ViewBag.user = user;
```

Продолжение приложения Б

```
        return View();
    }
    return RedirectToAction("Login", "Auth");
}

// POST:
[HttpPost("~/profile/edit")]
[ValidateAntiForgeryToken]
public async Task<ActionResult> Update(
    string login,
    string oldPassword,
    string newPassword1,
    string newPassword2,
    string name) {
    User user = getCurrentUser(HttpContext);
    if (user != null) {
        if (!newPassword1.Equals(newPassword2)) {
            return RedirectToAction("Update", new { message = "Пароли не
совпадают" });
        }
        if (LibDbContext.Instance.Users.Where(u => u.Login == login && u.Id !=
user.Id).Any()) {
            return RedirectToAction("Update", new { message =
"Пользователь с таким логином уже есть" });
        }
        if (LibDbContext.Instance.Users.Where(u => u.Id == user.Id &&
u.Password != oldPassword).Any()) {
            return RedirectToAction("Update", new { message = "Старый
пароль неверный" });
        }
        user.Login = login;
        user.Password = newPassword1;
        user.Name = name;
        LibDbContext.Instance.Users.Update(user);
        await LibDbContext.Instance.SaveChangesAsync();
    }
}
```

Продолжение приложения Б

```
        return RedirectToAction("Profile");
    }
    return RedirectToAction("Login", "Auth");
}
}
```

Views/Auth/Login.cshtml

```
@{
    ViewData["Title"] = "Авторизация";
}
<div class="col-xs-12 col-sm-10 col-md-8 col-lg-6 mt-3 mx-auto">
    <h2>@ViewData["Title"]</h2>
    @{
        if (@ViewBag.message != null) {
            <div class="alert alert-danger" role="alert">
                @ViewBag.message
            </div>
        }
    }
    <div class="">
        <form asp-controller="Auth" asp-action="Login" method="post">
            <div class="form_group mt-2 mb-2">
                @*<label for="login">Логин:</label>*@
                <input class="form-control" id="login" name="login" type="email" required
placeholder="Логин" />
            </div>

            <div class="form_group mt-2 mb-2">
                @*<label for="password">Пароль:</label>*@
                <input class="form-control" id="password" name="password"
type="password" required placeholder="Пароль" />
            </div>

            <input class="btn btn-success mt-2 mb-2" type="submit" value="Войти" />
        </form>
    </div>
</div>
```


Продолжение приложения Б

```
</form>
</div>
</div>

Views/Auth/Register.cshtml
@{
    ViewData["Title"] = "Регистрация";
}
<div class="col-xs-12 col-sm-10 col-md-8 col-lg-6 mt-3 mx-auto">
    <h2>@ViewData["Title"]</h2>
    @{
        if (@ViewBag.message != null) {
            <div class="alert alert-danger" role="alert">
                @ViewBag.message
            </div>
        }
    }
    <div class="">
        <form asp-controller="Auth" asp-action="Register" method="post">
            <div class="form_group mt-2 mb-2">
                @*<label for="login">Логин:</label>*@
                <input class="form-control" id="login" name="login" type="email" required
placeholder="Логин" />
            </div>

            <div class="form_group mt-2 mb-2">
                @*<label for="password1">Пароль:</label>*@
                <input class="form-control" id="password1" name="password1"
type="password" required placeholder="Пароль" />
            </div>

            <div class="form_group mt-2 mb-2">
                @*<label for="password2">Подтверить пароль:</label>*@
                <input class="form-control" id="password2" name="password2"
type="password" required placeholder="Подтверить пароль" />
            </div>
        </form>
    </div>
</div>
```

Продолжение приложения Б

```
</div>

<div class="form_group mt-2 mb-2">
    @*<label for="name">Имя:</label>*@
    <input class="form-control" id="name" name="name" type="text" required
placeholder="Имя" />
</div>

<input class="btn btn-success mt-2 mb-2" type="submit"
value="Зарегистрироваться" />

</form>
</div>
</div>
```

Views/Author/All.cshtml

```
@{
    ViewData["Title"] = "Авторы";
}
<div>
    @{
        <nav aria-label="breadcrumb">
            <ol class="breadcrumb">
                <li class="breadcrumb-item"><a asp-controller="Author" asp-action="All">Все авторы</a></li>
                <li class="breadcrumb-item"></li>
            </ol>
        </nav>

        User user = @ViewBag.user;
        if (@ViewBag.IsAdmin != null && @ViewBag.IsAdmin == true) {
            <a class="btn btn-primary mt-2 mb-2" asp-area="" asp-controller="Author" asp-
action="Create">Добавить</a>
        }
    }
</div>
```

Продолжение приложения Б

```
<div class="row">
  @{
    List<Author> authors = ViewBag.authors;
    if (authors != null) {
      List<char> firstLetters = authors.Select(a => a.Name.ToUpper()[0]).Distinct().ToList();

      <div class="col-xs-2 col-sm-2 col-md-2 ">
        <div id="list-letters" class="list-group">
          @{
            for (int i = 0; i < firstLetters.Count; i++) {
              <a class="list-group-item list-group-item-action" href="#list-letters-item-
@i">@firstLetters[i]</a>
            }
          }
        </div>
      </div>

      <div class="col-xs-10 col-sm-10 col-md-10">
        <div data-bs-spy="scroll" data-bs-target="#list-letters" data-bs-smooth-scroll="true"
class="scrollspy-example align-content-end" tabindex="0">
          @{
            for (int i = 0; i < firstLetters.Count; i++) {
              <div class="mb-3">
                <h4 id="list-letters-item-@i">@firstLetters[i]</h4>
                <ul class="list-unstyled">
                  @{
                    List<Author> authorsByLetter = authors.Where(a => a.Name.ToUpper()[0] ==
firstLetters[i]).ToList();
                    if (authorsByLetter.Count != 0) {
                      foreach (var author in authorsByLetter) {
                        <li class="d-inline " style="width: auto;">
                          <partial name="Partial/_AuthorItem" model="author"></partial>
                        </li>
                      }
                    }
                  }
                }
              </div>
            }
          }
        </div>
      </div>
    }
  }
</div>
```

Продолжение приложения Б

```
        }
      </ul>
    </div>
  }
}
</div>
</div>
}
}
</div>
```

```
<div>
  <partial name="Partial/_SavedToSessionItem"></partial>
</div>
```

Views/Author/One.cshtml

```
@{
    Author author = ViewBag.author;
    ViewData["Title"] = @author.Name;
}
<nav aria-label="breadcrumb">
    <ol class="breadcrumb">
        <li class="breadcrumb-item"><a asp-controller="Author" asp-action="All">Все
авторы</a></li>
        <li class="breadcrumb-item">@author.Name</li>
    </ol>
</nav>

<div class="row mb-5">
    <div class="col-xs-12 col-sm-4 col-md-3">
        
    </div>
    <div class="col-xs-12 col-sm-8 col-md-9">
        <h2>@author.Name</h2>
```

Продолжение приложения Б

```
<p>@author.Biography</p>
@{
    User user = @ViewBag.user;
    if (@ViewBag.IsAdmin != null && @ViewBag.IsAdmin == true) {
        <a class="btn btn-primary mt-2 mb-2" asp-area="" asp-controller="Author"
asp-action="Edit" asp-route-id="@author.Id">Редактировать</a>
    }
}
</div>
</div>

<nav class="mt-3">
    <div class="nav nav-tabs" id="nav-tab" role="tablist">
        <button class="nav-link active" id="nav-popular-tab" data-bs-toggle="tab" data-bs-
target="#nav-popular"
            type="button" role="tab" aria-controls="nav-home" aria-selected="true">
            Популярное
        </button>
        <button class="nav-link" id="nav-new-tab" data-bs-toggle="tab" data-bs-target="#nav-new"
            type="button" role="tab" aria-controls="nav-profile" aria-selected="false">
            Новинки
        </button>
    </div>
</nav>

<div class="tab-content mt-3" id="nav-tabContent">
    <div class="tab-pane fade show active" id="nav-popular" role="tabpanel" aria-labelledby="nav-
popular-tab" tabindex="0">
        <div class="mt-2">
            <ul class="list-unstyled row">
                @{
                    foreach (var book in ViewBag.popularBooks) {
                        <li class="d-inline " style="width: auto;">
```

Продолжение приложения Б

```

                                <partial name="_BookItem"
model="book"></partial>

                                </li>
                                }
                                if (ViewBag.popularBooks.Count == 0) {
                                    <div class="p-1">
                                        Пусто
                                    </div>
                                }
                            }
                        </ul>
                    </div>
                </div>
            <div class="tab-pane fade" id="nav-new" role="tabpanel" aria-labelledby="nav-new-tab"
tabindex="0">
                <div class="mt-2">
                    <ul class="list-unstyled row">
                        @{
                            foreach (var book in ViewBag.newBooks) {
                                <li class="d-inline " style="width: auto;">
                                    <partial name="_BookItem"
model="book"></partial>
                                </li>
                            }
                            if (ViewBag.newBooks.Count == 0) {
                                <div class="p-1">
                                    Пусто
                                </div>
                            }
                        }
                    </ul>
                </div>
            </div>
        </div>
    </div>

```

Views/Author/Update.cshtml

Продолжение приложения Б

```
@{
    if (@ViewBag.message != null) {
        <div class="alert alert-danger" role="alert">
            @ViewBag.message
        </div>
    }
    Author author = ViewBag.author;
    if (author != null) {
        ViewData["Title"] = author.Name + " - Редактирование";
        <form asp-controller="Author" asp-action="Update" method="post">
            <input class="form-control" type="text" name="id" value="@author.Id" hidden />
            <div class="form_group mt-2 mb-2">
                <input class="form-control" type="text" name="name" placeholder="Имя"
value="@author.Name" required />
            </div>
            <div class="form_group mt-2 mb-2">
                <input class="form-control" type="text" name="photo"
placeholder="Ссылка на фото" value="@author.Photo" required />
            </div>
            <div class="form_group mt-2 mb-2">
                <textarea class="form-control" name="biography" rows="5" placeholder="Биография"
required>@author.Biography</textarea>
            </div>
            <input class="btn btn-success mt-2 mb-2" type="submit" value="Сохранить" />
        </form>
        <form asp-controller="Author" asp-action="Delete" asp-route-id="@author.Id" method="post">
            <input class="btn btn-danger mt-2" type="submit" value="Удалить">
        </form>
    } else {
        ViewData["Title"] = "Добавление автора";
        <form asp-controller="Author" asp-action="Update" method="post">
            <input class="form-control" type="text" name="id" value="-1" hidden />
            <div class="form_group mt-2 mb-2">
                <input class="form-control" type="text" name="name" placeholder="Имя" required />
            </div>
```

Продолжение приложения Б

```
<div class="form_group mt-2 mb-2">
    <input class="form-control" type="text" name="photo" placeholder="Ссылка на фото" required
/>

</div>

<div class="form_group mt-2 mb-2">
    <textarea class="form-control" name="biography" rows="5" placeholder="Биография"
required></textarea>
</div>

<input class="btn btn-success mt-2 mb-2" type="submit" value="Сохранить" />
</form>
}
}
```

Views/Book/All.cshtml

```
@{
    ViewData["Title"] = @ViewBag.title != null ? @ViewBag.title : "Все книги";
}

<nav aria-label="breadcrumb">
    <ol class="breadcrumb">
        <li class="breadcrumb-item"><a asp-controller="Book" asp-action="All">Все
книги</a></li>
        <li class="breadcrumb-item active" aria-current="page">
            @{
                if (@ViewBag.title != "Все книги") {
                    @ViewBag.title
                }
            }
        </li>
    </ol>
</nav>

<div class="row">
    @{
        string classForBooks = "";
        if (ViewBag.isTop == null && ViewBag.isTop != true) {
            <div class="col-xs-12 col-sm-3 col-md-2">
```


Продолжение приложения Б

```
<div class="border rounded p-3 ">
    <label class="mb-2">Жанры</label>
    @{
        List<Genre> genres =
LibDbContext.Instance.Genres.OrderBy(g => g.Name).ToList();
        foreach (var genre in genres) {
            <h5>
                <a style="text-decoration: none;" asp-
controller="Genre" asp-action="One" asp-route-id="@genre.Id">@genre.Name</a>
            </h5>
        }
    }
</div>

</div>
classForBooks = "col-xs-12 col-sm-9 col-md-10";
}
}
<div class="@classForBooks">
    <div class="">
        <h2>@ViewBag.title</h2>
    </div>
    @{
        if (ViewBag.isTop == null && ViewBag.isTop != true) {
            User user = @ViewBag.user;
            if (@ViewBag.IsAdmin != null && @ViewBag.IsAdmin == true) {
                <h5>
                    <a class="btn btn-primary mt-2 mb-2" asp-area="" asp-
controller="Book" asp-action="Create">Добавить книгу</a>
                </h5>
            }
        }
    }
    <div class="">
        <ul class="list-unstyled row mt-3">
            @{
```

Продолжение приложения Б

```
        foreach (var book in ViewBag.books) {
            <li class="d-inline " style="width: auto;">
                <partial name="Partial/_BookItem"
model="book"></partial>

            </li>
        }
        if (ViewBag.books.Count == 0) {
            <div class="p-1">
                Пусто
            </div>
        }
    }
</ul>
</div>
</div>
</div>

<div class="mt-3">
    <partial name="Partial/_SavedToSessionItem"></partial>
</div>
```

Views/Book/One.cshtml

```
@{
    Book book = @ViewBag.book;
    ViewData["Title"] = @book.Name;
}
<div class="row mb-5">
    @{
        if (@ViewBag.message != null) {
            <div class="alert alert-danger" role="alert">
                @ViewBag.message
            </div>
        }
    }
    <div class="col-xs-12 col-sm-4 col-md-3">
```

Продолжение приложения Б

```

</div>
<div class="col-xs-12 col-sm-8 col-md-9">
    <div class="mb-3">
        <div class="d-lg-none pt-3"></div> @* отступ для мобилок *@
        <h2>@book.Name</h2>
        @{
            if (@ViewBag.IsAdmin != null && @ViewBag.IsAdmin == true) {
                <a class="btn btn-primary mt-2 mb-3" asp-area="" asp-
controller="Book" asp-action="Edit" asp-route-id="@book.Id">Редактировать</a>
            }
        }
        <h5>
            @{
                for (int i = 0; i < ViewBag.authors.Count; i++) {
                    <a asp-controller="Author" asp-action="One" asp-route-
id="@ViewBag.authors[i].Id" class="" style="text-decoration: none;">
                        @ViewBag.authors[i].Name
                    </a>
                    if (i != ViewBag.authors.Count - 1) {
                        <span class=" ">, </span>
                    }
                }
            }
        </h5>
        <h5>
            <span>Жанр: </span>
            @{
                for (int i = 0; i < ViewBag.genres.Count; i++) {
                    <a asp-controller="Genre" asp-action="One" asp-route-
id="@ViewBag.genres[i].Id" class="" style="text-decoration: none;">
                        <span>@ViewBag.genres[i].Name</span>
                    </a>
                    if (i != ViewBag.genres.Count - 1) {
                        <span class=" ">, </span>
                    }
                }
            }
        </h5>
    </div>
</div>
```

Продолжение приложения Б

```

    }
    }
    }
</h5>
</div>
<div class="my-3">
    <button href="#" class="btn btn-outline-warning" type="button" data-bs-
toggle="collapse"
                                data-bs-target="#collapseRating" aria-expanded="false" aria-
controls="collapseRating">
        <h5 class="mb-0">
            
            @{
                if (book.AvgRating.HasValue) {
                    <span class="d-
inline">@book.AvgRating.Value.ToString("f")</span>
                } else {
                    <span class="d-inline">нет оценок</span>
                }
            }
        </h5>
    </button>
    <div class="collapse mt-1" id="collapseRating">
        <div class="card card-body col-xs-12 col-sm-12 col-md-12 col-lg-6">
            @{
                List<Review> reviews = ViewBag.reviews;
                List<int> ratings = reviews.Select(r => r.Rating).ToList();
                for (int i = 5; i > 0; i--) {
                    List<int> iRatings = ratings.Where(r => r ==
i).ToList();

                    double percent = 0;
                    if (ratings.Count != 0) {
                        percent = ((double)iRatings.Count /
ratings.Count) * 100;
                    }
                }
            }
        </div>
    </div>
</div>

```

Продолжение приложения Б

```

percent = Math.Round(percent, 2);
<div class="row my-1">
    <div class="col-2 d-flex">
        <div class="m-auto">
            
            <span class="d-
inline">@i</span>
        </div>
    </div>
    <div class="col-2">
        <div class="m-auto">
            <span class="d-inline text-
nowrap">@percent%</span>
        </div>
    </div>
    <div class="col-5">
        <div class="">
            <div class="progress">
                @ {
                    string
percentStr = percent.ToString().Replace(',', '.');
                }
            <div
class="progress-bar" role="progressbar" aria-label="Basic example"
style="width: @percentStr%" aria-valuenow="25" aria-valuemin="0" aria-valuemax="100"></div>
        </div>
    </div>
    <div class="col-3">
        <div class="m-auto">
            <span class="d-
inline">@iRatings.Count</span>
        </div>

```

Продолжение приложения Б

```

        </div>
    </div>
}
}
</div>
</div>
</div>
<div class="my-3">
    @ {
        if (ViewBag.user != null) {
            User user = ViewBag.User;
            FeaturedBook featuredBook =
book.FeaturedBooks.FirstOrDefault(fb => fb.UserId == user.Id);
            Review review = book.Reviews.FirstOrDefault(r => r.UserId ==
user.Id);

            if (featuredBook != null) {
                <button type="button" class="btn btn-success d-inline" data-
bs-toggle="modal" data-bs-target="#modalForFeatureBook">
                    
                    <span>Добавлено</span>
                </button>
            } else {
                <button type="button" class="btn btn-primary d-inline"
data-bs-toggle="modal" data-bs-target="#modalForFeatureBook">
                    
                    <span>Добавить</span>
                </button>
            }
        }
    <!-- Modal -->
    <div class="modal fade" id="modalForFeatureBook" tabindex="-1"
role="dialog"
        aria-labelledby="modalForFeatureBookTitle" aria-
hidden="true">

```

Продолжение приложения Б

```
<div class="modal-dialog modal-dialog-centered"
role="document">

    <div class="modal-content">
        @{
            <form class="form-inline" asp-
controller="Featured" asp-action="One" method="post">

                <div class="modal-header">
                    <h5 class="modal-
title">Избранное</h5>

                </div>
                <div class="modal-body">
                    <input
type="hidden" name="book_id" value="@book.Id" />

                    <div class="form-
group">

                        <label
for="selectMark">Мерка</label>

                        <select
id="selectMark" class="form-select mt-2" name="mark_id">

                            @ {

                                <option value="0">-</option>

                                foreach (var mark in ViewBag.marks) {

                                    if (featuredBook != null && featuredBook.MarkId == mark.Id) {

                                        <option value="@mark.Id" selected>@mark.Name</option>

                                    } else {

                                        <option value="@mark.Id">@mark.Name</option>

                                    }

                                }

                            }

                        }

                    }

                }

            }

        }

    }

</div>

</div>
```

Продолжение приложения Б

```

    }

    }
    </select>
  </div>
</div>
<div class="modal-footer">
  <button
type="button" class="btn btn-secondary" data-bs-dismiss="modal">Отмена</button>
  <input class="btn
btn-primary" type="submit" value="Сохранить" />
  </div>
</form>
}
</div>
</div>
</div>
if (review != null) {
  <button type="button" class="btn btn-success d-inline" data-
bs-toggle="modal" data-bs-target="#modalForReviewForBook">
    
    <span>Редактировать рецензию</span>
  </button>
} else {
  <button type="button" class="btn btn-primary d-inline"
data-bs-toggle="modal" data-bs-target="#modalForReviewForBook">
    
    <span>Написать рецензию</span>
  </button>
}
<!-- Modal -->
<div class="modal fade" id="modalForReviewForBook"
tabindex="-1" role="dialog"

```


Продолжение приложения Б

```
aria-labelledby="modalForReviewForBookTitle" aria-
hidden="true">
<div class="modal-dialog modal-dialog-centered"
role="document">
    <div class="modal-content">
        @{
            <div class="modal-header">
                <h5 class="modal-
title">Рецензия</h5>
                @{
                    if (review != null) {
                        <form
class="form-inline" asp-controller="Review" asp-action="Delete" method="post">
                            <input type="hidden" name="book_id" value="@book.Id" />
                            <input class="btn btn-danger" type="submit" value="Удалить" />
                        </form>
                    }
                }
            </div>
            <form class="form-inline" asp-
controller="Review" asp-action="Update" method="post">
                <div class="modal-body">
                    <input
type="hidden" name="book_id" value="@book.Id" />
                    <div class="form-
group">
                        <label
for="selectRating">Оценка</label>
                        <select
id="selectRating" class="form-select mt-2" name="rating">
                            @{
                                for (int i = 1; i < 6; i++) {
```

Продолжение приложения Б

```
        if (review != null && review.Rating == i) {

            <option value="@i" selected>@i</option>

        } else {

            <option value="@i">@i</option>

        }

    }

}

</select>
</div>
<div class="form-

group mt-2">

    <label

for="textareaContent">Текст</label>

    <textarea

class="form-control mt-2" id="textareaContent"

    rows="5" name="content" required> @{

        if (review != null) {

            @review.Content

        }

    }</textarea>

    </div>
</div>
<div class="modal-footer">
```

Продолжение приложения Б

```

type="button" class="btn btn-secondary" data-bs-dismiss="modal">Отмена</button>
<input class="btn
btn-primary" type="submit" value="Сохранить" />

</div>
</form>
}
</div>
</div>
</div>
} else { // если юзер не залогинен, то кнопки перенаправляют на
авторизацию
<a asp-controller="Auth" asp-action="Login" type="button"
class="btn btn-primary d-inline">

<span>Добавить</span>
</a>
<a asp-controller="Auth" asp-action="Login" type="button"
class="btn btn-primary d-inline">

<span>Написать рецензию</span>
</a>
}
}
</div>
<div class="mt-3 mb-2">
@{
    var featuredBooks = @book.FeaturedBooks.ToList();
    var read = featuredBooks.Where(fb => fb.MarkId == 3);
    var readLater = featuredBooks.Where(fb => fb.MarkId == 1);
}
<h5>
<a class="d-inline" data-bs-toggle="modal"
data-bs-target="#modalForUsersWhoRead">
    Прочитали - @read.Count()
```

Продолжение приложения Б

```

</a>
</h5>
<!-- Modal -->
<div class="modal fade" id="modalForUsersWhoRead" tabindex="-1" role="dialog"
    aria-labelledby="modalForReviewForBookTitle" aria-hidden="true">
    <div class="modal-dialog modal-dialog-centered" role="document">
        <div class="modal-content">
            <div class="modal-header">
                <h5 class="modal-title">Прочитали</h5>
            </div>
            <div class="modal-body">
                <div class="form-group">
                    @{
                        foreach (var user in read.Select(fb
=> fb.User)) {
                            <h4>
                                <a class="" asp-
controller="User" asp-action="One"
                                asp-route-
id="@user.Id">@user.Name</a>
                            </h4>
                        }
                    }
                </div>
            </div>
            <div class="modal-footer">
                <button type="button" class="btn btn-secondary"
data-bs-dismiss="modal">Ок</button>
            </div>
        </div>
    </div>
</div>
</h5>
<a class=" d-inline " data-bs-toggle="modal"
    data-bs-target="#modalForUsersWhoReadLater">

```

Продолжение приложения Б

```
Планируют - @readLater.Count()
</a>
</h5>
<!-- Modal -->
<div class="modal fade" id="modalForUsersWhoReadLater" tabindex="-1"
role="dialog"

aria-labelledby="modalForReviewForBookTitle" aria-hidden="true">
<div class="modal-dialog modal-dialog-centered" role="document">
  <div class="modal-content">
    <div class="modal-header">
      <h5 class="modal-title">Планируют</h5>
    </div>
    <div class="modal-body">
      <div class="form-group">
        @ {
          foreach (var user in
readLater.Select(fb => fb.User)) {
            <h4>
              <a class="" asp-
controller="User" asp-action="One"
asp-route-
id="@user.Id">@user.Name</a>
            </h4>
          }
        }
      </div>
    </div>
    <div class="modal-footer">
      <button type="button" class="btn btn-secondary"
data-bs-dismiss="modal">Ок</button>
    </div>
  </div>
</div>
</div>
</div>
</div>
```

Продолжение приложения Б

```
</div>
</div>
<div class="mt-3">
  <nav>
    <div class="nav nav-tabs" id="nav-tab" role="tablist">
      <button class="nav-link active" id="nav-description-tab" data-bs-toggle="tab" data-
bs-target="#nav-description"
      type="button" role="tab" aria-controls="nav-home" aria-
selected="true">
        О книге
      </button>
      <button class="nav-link" id="nav-reviews-tab" data-bs-toggle="tab" data-bs-
target="#nav-reviews"
      type="button" role="tab" aria-controls="nav-profile" aria-
selected="false">
        Рецензии<span class="mx-1 text-
secondary">@ViewBag.reviews.Count</span>
      </button>
    </div>
  </nav>
  <div class="tab-content mt-3" id="nav-tabContent">
    <div class="tab-pane fade show active" id="nav-description" role="tabpanel" aria-
labelledby="nav-description-tab" tabindex="0">
      <p>
        <h5>Описание</h5>
        @book.Description
      </p>
      <p>
        <h5>Дата написания</h5>
        @book.DateOfCreation
      </p>
    </div>
    <div class="tab-pane fade" id="nav-reviews" role="tabpanel" aria-labelledby="nav-reviews-
tab" tabindex="0">
      @{
```

Продолжение приложения Б

```
if (ViewBag.reviews.Count == 0) {
    <div class="p-1">
        Пусто
    </div>
} else {
    for (int i = 0; i < reviews.Count; i++) {
        Review review = @reviews[i];
        <div id="@review.Id" class="border rounded p-3 mb-3">
            <h5>
                <a asp-controller="User" asp-action="One"
asp-route-id="@review.User.Id" class="" style="text-decoration: none;">
                    @review.User.Name
                </a>
            </h5>
            <div class="text-secondary">
                @review.StrDateOfCreation
            </div>
            <h5 class="py-1">
                
                <span class="align-baseline d-
inline">@review.Rating</span>
            </h5>
            <div class="pb-1">
                @review.Content
            </div>
            <div class="">
                @ {
                    List<Like> likes =
review.Likes.ToList();
                <button type="button" class="btn
btn-outline-secondary border-0" data-bs-toggle="modal"
                data-bs-
target="#modalForUsersLikesFromReview_@review.Id">
```

Продолжение приложения Б

```

inline">@likes.Count </span>
<span class="d-
</button>
<!-- Modal -->
<div class="modal fade"
id="modalForUsersLikesFromReview_@review.Id" tabindex="-1" role="dialog"
aria-
labelledby="modalForReviewForBookTitle" aria-hidden="true">
<div class="modal-dialog
modal-dialog-centered" role="document">
<div class="modal-
content">
<div
class="modal-header">
<h5
class="modal-title">Лайки</h5>
</div>
<div
class="modal-body">
<div
class="form-group">
@ {
if (likes.Count == 0) {
<h4>Пусто</h4>
}
foreach (var like in likes) {
<h4>
<a class="" asp-controller="User" asp-action="One"

```


Продолжение приложения Б

```
asp-route-
id="@like.User.Id">@like.User.Name</a>

</h4>

    }

}

</div>

</div>
<div
class="modal-footer">

    <button type="button" class="btn btn-secondary" data-bs-dismiss="modal">Ок</button>

</div>
</div>
</div>
</div>
bool IsChecked = false;
if (ViewBag.user != null) {
    User user = ViewBag.User;

    for (int j = 0; j <
likes.Count; j++) {

        if (likes.Select(l =>
l.UserId).Contains(user.Id)) {

            IsChecked =
true;

        }

    }

}
if (IsChecked) {
    <form asp-
controller="Review" asp-action="Unlike"
```

Продолжение приложения Б

```

asp-route-review_id="@review.Id" asp-route-
book_id="@book.Id" method="post" class="mx-1 d-inline">
<button
type="submit" class="btn btn-outline-danger border-0">

</button>
</form>
} else {
<form asp-
controller="Review" asp-action="Like"
asp-route-review_id="@review.Id" asp-route-
book_id="@book.Id" method="post" class="mx-1 d-inline">
<button
type="submit" class="btn btn-outline-su border-0">

</button>
</form>
}
if (@ViewBag.IsAdmin != null &&
@ViewBag.IsAdmin == true) {
User user = ViewBag.User;
<form class="form-inline
mx-2 d-inline" asp-controller="Review" asp-action="DeleteByAdmin" method="post">
<input
type="hidden" name="book_id" value="@book.Id" />
<input
type="hidden" name="user_id" value="@review.UserId" />
<input class="btn
btn-danger" type="submit" value="Снять с публикации" />
</form>
}
}
</div>

```

Продолжение приложения Б

```

        </div>
    }
}
}
</div>
</div>
</div>

```

Views/Book/Update.cshtml

```

@{
    if (@ViewBag.message != null) {
        <div class="alert alert-danger" role="alert">
            @ViewBag.message
        </div>
    }
    Book book = ViewBag.book;
    if (book != null) {
        ViewData["Title"] = book.Name + " - Редактирование";
        <form asp-controller="Book" asp-action="Update" method="post">
            <input class="form-control" type="text" name="id" value="@book.Id" hidden />
            <div class="form_group mt-2 mb-2">
                <input class="form-control" type="text" name="name" placeholder="Имя" value="@book.Name"
required />
            </div>
            <div class="form_group mt-2 mb-2">
                <textarea class="form-control" name="description" rows="5"
placeholder="Описание" required>@book.Description</textarea>
            </div>
            <div class="form_group mt-2 mb-2">
                <input class="form-control" type="text" name="photo" placeholder="Ссылка на фото"
value="@book.Photo" required />
            </div>
            <div class="form_group mt-2 mb-2">
                <input class="form-control" type="text" name="date_of_creation" placeholder="Дата создания"
value="@book.DateOfCreation" required />

```

Продолжение приложения Б

```
</div><div class="form_group mt-2 mb-2">
  <label for="selectGenre">Жанры книги</label>
  <select id="selectGenre" class="form-select mt-2" name="genre_ids" multiple>
    @{
      List<Int32> g_ids = ViewBag.genre_ids;
      foreach (var genre in ViewBag.genres) {
        if (g_ids.Contains(genre.Id)) {
          <option value="@genre.Id" selected>@genre.Name</option>
        } else {
          <option value="@genre.Id">@genre.Name</option>
        }
      }
    }
  </select>
</div>
<div class="form_group mt-2 mb-2">
  <label for="selectAuthor">Авторы книги</label>
  <select id="selectAuthor" class="form-select mt-2" name="author_ids" multiple>
    @{
      List<Int32> author_ids = ViewBag.author_ids;
      foreach (var author in ViewBag.authors) {
        if (author_ids.Contains(author.Id)) {
          <option value="@author.Id" selected>@author.Name</option>
        } else {
          <option value="@author.Id">@author.Name</option>
        }
      }
    }
  </select>
</div>
  <input class="btn btn-success mt-2 mb-2" type="submit" value="Сохранить" />
</form>
<form asp-controller="Book" asp-action="Delete" asp-route-id="@book.Id" method="post">
  <input class="btn btn-danger mt-2" type="submit" value="Удалить">
</form>
```

Продолжение приложения Б

```
} else {
    ViewData["Title"] = "Добавление книги";
    <form asp-controller="Book" asp-action="Update" method="post">
        <input class="form-control" type="text" name="id" value="-1" hidden />
        <div class="form_group mt-2 mb-2">
            <input class="form-control" type="text" name="name" placeholder="Имя" required />
        </div>
        <div class="form_group mt-2 mb-2">
            <textarea class="form-control" name="description" rows="5"
placeholder="Описание" required></textarea>
        </div>
        <div class="form_group mt-2 mb-2">
            <input class="form-control" type="text" name="photo" placeholder="Ссылка на фото" required
/>
        </div>
        <div class="form_group mt-2 mb-2">
            <input class="form-control" type="text" name="date_of_creation" placeholder="Дата создания"
required />
        </div>
        <div class="form_group mt-2 mb-2">
            <label for="selectGenre">Жанры книги</label>
            <select id="selectGenre" class="form-select mt-2" name="genre_ids" multiple>
                @{
                    foreach (var genre in ViewBag.genres) {
                        <option value="@genre.Id">@genre.Name</option>
                    }
                }
            </select>
        </div>
        <div class="form_group mt-2 mb-2">
            <label for="selectAuthor">Авторы книги</label>
            <select id="selectAuthor" class="form-select mt-2" name="author_ids" multiple>
                @{
                    foreach (var author in ViewBag.authors) {
                        <option value="@author.Id">@author.Name</option>
                    }
                }
            </select>
        </div>
    </form>
}
```

Продолжение приложения Б

```
        }
    }
</select>
</div>
<input class="btn btn-success mt-2 mb-2" type="submit" value="Сохранить" />
</form>
}
}
```

Views/Featured/All.cshtml

```
@{
    ViewData["Title"] = "Избранное";
    User user = ViewBag.user;
}
<h2>Избранное</h2>
<div class="">
    <nav class="mt-3">
        <div class="nav nav-tabs" id="nav-tab" role="tablist">
            @{
                List<FeaturedBook> featuredBooks = ViewBag.featuredBooks;
                var readLater = featuredBooks.Where(fb => fb.MarkId == 1).Select(fb =>
fb.Book).ToList();

                var readNow = featuredBooks.Where(fb => fb.MarkId == 2).Select(fb =>
fb.Book).ToList();

                var read = featuredBooks.Where(fb => fb.MarkId == 3).Select(fb =>
fb.Book).ToList();

                var abandonedReading = featuredBooks.Where(fb => fb.MarkId ==
4).Select(fb => fb.Book).ToList();
            }
            <button class="nav-link active" id="nav-read-later-tab" data-bs-toggle="tab" data-
bs-target="#nav-read-later"

                type="button" role="tab" aria-controls="nav-home" aria-
selected="true">

                Прочитать позже<span class="mx-1 text-
secondary">@readLater.Count()</span>
```

Продолжение приложения Б

```
</button>
<button class="nav-link" id="nav-read-now-tab" data-bs-toggle="tab" data-bs-
target="#nav-read-now"
        type="button" role="tab" aria-controls="nav-profile" aria-
selected="false">
        Читаю<span class="mx-1 text-secondary">@readNow.Count()</span>
</button>
<button class="nav-link" id="nav-read-tab" data-bs-toggle="tab" data-bs-
target="#nav-read"
        type="button" role="tab" aria-controls="nav-profile" aria-
selected="false">
        Прочитано<span class="mx-1 text-secondary">@read.Count()</span>
</button>
<button class="nav-link" id="nav-abandoned-reading-tab" data-bs-toggle="tab"
data-bs-target="#nav-abandoned-reading"
        type="button" role="tab" aria-controls="nav-profile" aria-
selected="false">
        Зброшено<span class="mx-1 text-
secondary">@abandonedReading.Count()</span>
</button>
</div>
</nav>
<div class="tab-content mt-3" id="nav-tabContent">
    <div class="tab-pane fade show active" id="nav-read-later" role="tabpanel" aria-
labelledby="nav-read-later-tab" tabindex="0">
        <div class="mt-2">
            <ul class="list-unstyled row">
                @ {
                    foreach (var book in readLater) {
                        <li class="d-inline " style="width: auto;">
                            <partial name="Partial/_BookItem"
model="book"></partial>
                        </li>
                    }
                    if (readLater.Count() == 0) {
```

Продолжение приложения Б

```
<div class="p-1">
    Пусто
</div>
}
}
</ul>
</div>
</div>
<div class="tab-pane fade" id="nav-read-now" role="tabpanel" aria-labelledby="nav-read-
now-tab" tabindex="0">
    <div class="mt-2">
        <ul class="list-unstyled row">
            @{
                foreach (var book in readNow) {
                    <li class="d-inline " style="width: auto;">
                        <partial name="Partial/_BookItem"
model="book"></partial>
                    </li>
                }
                if (readNow.Count() == 0) {
                    <div class="p-1">
                        Пусто
                    </div>
                }
            }
        </ul>
    </div>
</div>
<div class="tab-pane fade" id="nav-read" role="tabpanel" aria-labelledby="nav-read-tab"
tabindex="0">
    <div class="mt-2">
        <ul class="list-unstyled row">
            @{
                foreach (var book in read) {
                    <li class="d-inline " style="width: auto;">
```


Продолжение приложения Б

```

model="book"></partial>
<partial name="Partial/_BookItem"
    </li>
    }
    if (read.Count() == 0) {
        <div class="p-1">
            Пусто
        </div>
    }
}
</ul>
</div>
</div>
<div class="tab-pane fade" id="nav-abandoned-reading" role="tabpanel" aria-
labelledby="nav-abandoned-reading-tab" tabindex="0">
    <div class="mt-2">
        <ul class="list-unstyled">
            @ {
                foreach (var book in abandonedReading) {
                    <li class="d-inline " style="width: auto;">
                        <partial name="Partial/_BookItem"
model="book"></partial>
                        </li>
                    }
                    if (abandonedReading.Count() == 0) {
                        <div class="p-1">
                            Пусто
                        </div>
                    }
                }
            }
        </ul>
    </div>
</div>
</div>
</div>
</div>

```

Продолжение приложения Б

Views/Genre/All.cshtml

```
@{
    ViewData["Title"] = "Жанры";
}
<nav aria-label="breadcrumb">
    <ol class="breadcrumb">
        <li class="breadcrumb-item"><a asp-controller="Genre" asp-action="All">Все жанры</a></li>
        <li class="breadcrumb-item"></li>
    </ol>
</nav>
<div class="row mb-3">
    <div class="col-xs-12 col-sm-3 col-md-2">
        <div class="border rounded p-3">
            <label class="mb-2">Жанры</label>
            @{
                User user = @ViewBag.user;
                if (@ViewBag.IsAdmin != null && @ViewBag.IsAdmin == true) {
                    <h5>
                        <a class="btn btn-primary mt-2 mb-2" asp-area="" asp-controller="Genre" asp-
action="Create">Добавить</a>
                    </h5>
                }
                foreach (var g in ViewBag.genres) {
                    <h5>
                        <a style="text-decoration: none;" asp-controller="Genre" asp-action="One" asp-route-
id="@g.Id">@g.Name</a>
                    </h5>
                }
            }
        </div>
    </div>
    <div class="col-xs-12 col-sm-9 col-md-10">
        <h2 class="">Выберите жанр</h2>
    </div>
</div>
```

Продолжение приложения Б

[illegible]

Продолжение приложения Б

```
    }
  </div>
</div>
<div class="col-xs-12 col-sm-9 col-md-10">
  <div class="">
    <h2>@genre.Name</h2>
  </div>
  @ {
    if (@ViewBag.IsAdmin != null && @ViewBag.IsAdmin == true) {
      <a class="btn btn-primary mt-2 mb-2" asp-area="" asp-controller="Genre"
asp-action="Edit" asp-route-id="@genre.Id">Редактировать</a>
    }
  }
  <nav class="mt-3">
    <div class="nav nav-tabs" id="nav-tab" role="tablist">
      <button class="nav-link active" id="nav-popular-tab" data-bs-toggle="tab"
data-bs-target="#nav-popular"
      type="button" role="tab" aria-controls="nav-home" aria-
selected="true">
        Популярное
      </button>
      <button class="nav-link" id="nav-new-tab" data-bs-toggle="tab" data-bs-
target="#nav-new"
      type="button" role="tab" aria-controls="nav-profile" aria-
selected="false">
        Новинки
      </button>
    </div>
  </nav>
  <div class="tab-content mt-3" id="nav-tabContent">
    <div class="tab-pane fade show active" id="nav-popular" role="tabpanel" aria-
labelledby="nav-popular-tab" tabindex="0">
      <div class="mt-2">
        <ul class="list-unstyled row">
          @ {
```

Продолжение приложения Б

```

                                foreach (var book in ViewBag.popularBooks) {
                                    <li class="d-inline " style="width: auto;">
                                        <partial name="Partial/_BookItem"
model="book"></partial>
                                    </li>
                                }
                                if (ViewBag.popularBooks.Count == 0) {
                                    <div class="p-1">
                                        Пусто
                                    </div>
                                }
                            }
                        </ul>
                    </div>
                </div>
                <div class="tab-pane fade" id="nav-new" role="tabpanel" aria-labelledby="nav-
new-tab" tabindex="0">
                    <div class="mt-2">
                        <ul class="list-unstyled">
                            @{
                                foreach (var book in ViewBag.newBooks) {
                                    <li class="d-inline " style="width: auto;">
                                        <partial name="Partial/_BookItem"
model="book"></partial>
                                    </li>
                                }
                                if (ViewBag.newBooks.Count == 0) {
                                    <div class="p-1">
                                        Пусто
                                    </div>
                                }
                            }
                        </ul>
                    </div>
                </div>
            </div>

```

Продолжение приложения Б

```
        </div>
    </div>
</div>
<div class="mt-3">
    <partial name="Partial/_SavedToSessionItem"></partial>
</div>
```

Views/Genre/Update.cshtml

```
@{
    if (@ViewBag.message != null) {
        <div class="alert alert-danger" role="alert">
            @ViewBag.message
        </div>
    }
    Genre genre = ViewBag.genre;
    if (genre != null) {
        ViewData["Title"] = genre.Name + " - Редактирование";
        <form asp-controller="Genre" asp-action="Update" method="post">
            <input class="form-control" type="text" name="id" value="@genre.Id" hidden />
            <div class="form_group mt-2 mb-2">
                <input class="form-control" type="text" name="name" placeholder="Название"
value="@genre.Name" required />
            </div>
            <input class="btn btn-success mt-2 mb-2" type="submit" value="Сохранить" />
        </form>
        <form asp-controller="Genre" asp-action="Delete" asp-route-id="@genre.Id" method="post">
            <input class="btn btn-danger mt-2" type="submit" value="Удалить">
        </form>
    } else {
        ViewData["Title"] = "Добавление жанра";
        <form asp-controller="Genre" asp-action="Update" method="post">
            <input class="form-control" type="text" name="id" value="-1" hidden />
            <div class="form_group mt-2 mb-2">
                <input class="form-control" type="text" name="name" placeholder="Название" required />
            </div>
```

Продолжение приложения Б

```
        <input class="btn btn-success mt-2 mb-2" type="submit" value="Сохранить" />
    </form>
}
}
```

Views/Home/About.cshtml

```
@{
    ViewData["Title"] = "О сайте";
}
<div class="text-center">
    <h1 class="">About Lib</h1>
</div>
<div class="col-xs-12 col-sm-10 col-md-8 col-lg-6 mt-3 mx-auto">
    <h3>
        Книжный рекомендательный сервис для ведения читательского дневника, с подборками книг и
        рецензиями на них
    </h3>
</div>
<div class="text-center mt-3">
    <p class="">Разработчик - Аминов Арслан 19П-1</p>
</div>
```

Views/Home/Index.cshtml

```
@{
    ViewData["Title"] = "Главная";
}
@{
    if (ViewBag.showWelcome && @ViewBag.user != null) {
        <div class="alert alert-success" role="alert">
            Добро пожаловать, @ViewBag.user.Name!
        </div>
    }
}
<h2>Популярное</h2>
<div class="overflow-x-scroll mt-3 mb-5">
    <ul class="list-unstyled row flex-nowrap mb-0">
```

Продолжение приложения Б

```
@{
    foreach (var book in ViewBag.popularBooks) {
        <li class="d-inline " style="width: auto;">
            <partial name="Partial/_BookItem" model="book"></partial>
        </li>
    }
}
</ul>
</div>
<h2>Новинки</h2>
<div class="overflow-x-scroll mt-3 mb-5">
    <ul class="list-unstyled row flex-nowrap mb-0">
        @{
            foreach (var book in ViewBag.newBooks) {
                <li class="d-inline " style="width: auto;">
                    <partial name="Partial/_BookItem" model="book"></partial>
                </li>
            }
        }
    </ul>
</div>
<h2>Лучшие рецензии</h2>
<div class="overflow-x-scroll mt-3 mb-5">
    <ul class="list-unstyled row flex-nowrap mb-0">
        @{
            foreach (var review in ViewBag.bestReviews) {
                <li class="d-inline " style="width: auto;">
                    <partial name="Partial/_ReviewItem" model="review"></partial>
                </li>
            }
        }
    </ul>
</div>
<div>
    <partial name="Partial/_SavedToSessionItem"></partial>
</div>
```


Продолжение приложения Б

</div>

Views/Home/Search.cshtml

@{

ViewData["Title"] = "Поиск - " + @ViewBag.q;

}

<div>

<form class="form-inline" asp-controller="Home" asp-action="Search" method="get">

<h2>Поиск</h2>

<input class="form-control mt-2" type="search" name="q" placeholder="Введите запрос..."
value="@ViewBag.q" required />

<div class="form-group mt-3">

<label for="selectGenre">Жанр книги</label>

<select id="selectGenre" class="form-select mt-2" name="genre_id">

<option value="0">Все</option>

@{

foreach (var genre in ViewBag.genres) {

if (genre.Id == ViewBag.genre_id) {

<option value="@genre.Id"

selected>@genre.Name</option>

} else {

<option

value="@genre.Id">@genre.Name</option>

}

}

}

</select>

</div>

<input class="btn btn-success mt-3" type="submit" value="Поиск" />

</form>

</div>

@{

List<Book> books = ViewBag.books;

List<Author> authors = ViewBag.authors;

}

Продолжение приложения Б

```
<h2 class="mt-5">Результаты</h2>
<nav class="mt-3">
  <div class="nav nav-tabs" id="nav-tab" role="tablist">
    <button class="nav-link active" id="nav-books-tab" data-bs-toggle="tab" data-bs-
target="#nav-books"
      type="button" role="tab" aria-controls="nav-home" aria-selected="true">
      Книги<span class="mx-1 text-secondary">@books.Count()</span>
    </button>
    <button class="nav-link" id="nav-authors-tab" data-bs-toggle="tab" data-bs-target="#nav-
authors"
      type="button" role="tab" aria-controls="nav-profile" aria-selected="false">
      Авторы<span class="mx-1 text-secondary">@authors.Count()</span>
    </button>
  </div>
</nav>
<div class="tab-content mt-3" id="nav-tabContent">
  <div class="tab-pane fade show active" id="nav-books" role="tabpanel" aria-labelledby="nav-
books-tab" tabindex="0">
    <div class="mt-2">
      <ul class="list-unstyled row">
        @ {
          foreach (var book in books) {
            <li class="d-inline " style="width: auto;">
              <partial name="Partial/_BookItem"
model="book"></partial>
            </li>
          }
          if (books.Count == 0) {
            <div class="p-1">
              Пусто
            </div>
          }
        }
      </ul>
    </div>
  </div>
</div>
```

Продолжение приложения Б

```
</div>
<div class="tab-pane fade" id="nav-authors" role="tabpanel" aria-labelledby="nav-authors-tab"
tabindex="0">
    <div class="mt-2">
        <ul class="list-unstyled row">
            @{
                foreach (var author in authors) {
                    <li class="d-inline " style="width: auto;">
                        <partial name="Partial/_AuthorItem"
model="author"></partial>
                    </li>
                }
                if (authors.Count == 0) {
                    <div class="p-1">
                        Пусто
                    </div>
                }
            }
        </ul>
    </div>
</div>
</div>
<div>
    <partial name="Partial/_SavedToSessionItem"></partial>
</div>
```

Views/Note/All.cshtml

```
@{
    ViewData["Title"] = "Заметки";
}
<h2>Заметки</h2>
<div class="">
    <form asp-controller="Note" asp-action="Create" method="post">
        <input class="btn btn-primary" type="submit" value="Создать">
    </form>
```

Продолжение приложения Б

```
<ul class="mt-2 list-unstyled row">
    @{
        foreach (var note in ViewBag.notes) {
            <li class="d-inline" style="width: auto;">
                <div class="border rounded mb-3 p-3" style="width: 512px;">
                    <a class="" asp-controller="Note" asp-action="One" asp-
route-id="@note.Id" style="text-decoration: none;">
                        <h4 class="text-truncate">
                            @note.Name
                        </h4>
                        <p class="text-secondary mb-
0">@note.StrDateOfCreation</p>
                    </a>
                </div>
            </li>
        }
    }
</ul>
</div>
```

Views/Note/One.cshtml

```
@{
    ViewData["Title"] = @ViewBag.note.Name;
}
<div>
    @{
        if (@ViewBag.message != null) {
            <div class="alert alert-success" role="alert">
                @ViewBag.message
            </div>
        }
    }
    <a class="btn btn-secondary" asp-area="" asp-controller="Note" asp-action="All">Назад</a>
    <form asp-controller="Note" asp-action="Edit" asp-route-id="@ViewBag.note.Id" method="post">
```

Продолжение приложения Б

```
<input class="form-control mt-2" name="newName" value="@ViewBag.note.Name"
placeholder="Заголовок">
```

```
<textarea class="form-control mt-2" name="newContent" rows="5"
placeholder="Текст">@ViewBag.note.Content</textarea>
```

```
<input class="btn btn-success mt-2" type="submit" value="Сохранить">
</form>
<form asp-controller="Note" asp-action="Delete" asp-route-id="@ViewBag.note.Id"
method="post">
    <input class="btn btn-danger mt-2" type="submit" value="Удалить">
</form>
</div>
```

Views/User/One.cshtml

```
@{
    User user = @ViewBag.user;
    ViewData["Title"] = user.Name;
}
@{
    <div class="border rounded mb-3 p-3">
        <h2>@user.Name</h2>
        @{
            if (user.RoleId == 2) {
                <h3 class="text-warning">[ @user.Role.Name]</h3>
            }
            if (ViewBag.IsProfile != null && ViewBag.IsProfile) {
                <div class="mt-3">
                    <a class="btn btn-primary mb-2" asp-area="" asp-controller="User"
asp-action="Update">Редактировать</a>
                    <form asp-controller="Auth" asp-action="Logout" method="post">
                        <input class="btn btn-danger" type="submit"
value="Выйти">
                    </form>
                </div>
            }
        }
    </div>
}
```

Продолжение приложения Б

```
    }
  }
</div>
}
<div class="mt-3">
  <nav class="mt-3">
    <div class="nav nav-tabs" id="nav-tab" role="tablist">
      @ {
        List<FeaturedBook> featuredBooks = ViewBag.featuredBooks;
        var readLater = featuredBooks.Where(fb => fb.MarkId == 1).Select(fb =>
fb.Book).ToList();

        var readNow = featuredBooks.Where(fb => fb.MarkId == 2).Select(fb =>
fb.Book).ToList();

        var read = featuredBooks.Where(fb => fb.MarkId == 3).Select(fb =>
fb.Book).ToList();

        var abandonedReading = featuredBooks.Where(fb => fb.MarkId ==
4).Select(fb => fb.Book).ToList();

        var myReviews = user.Reviews.ToList();
      }
      <button class="nav-link active" id="nav-read-later-tab" data-bs-toggle="tab" data-
bs-target="#nav-read-later"

        type="button" role="tab" aria-controls="nav-home" aria-
selected="true">
        Прочитать позже<span class="mx-1 text-
secondary">@readLater.Count()</span>
      </button>
      <button class="nav-link" id="nav-read-now-tab" data-bs-toggle="tab" data-bs-
target="#nav-read-now"

        type="button" role="tab" aria-controls="nav-profile" aria-
selected="false">
        Читаю<span class="mx-1 text-secondary">@readNow.Count()</span>
      </button>
      <button class="nav-link" id="nav-read-tab" data-bs-toggle="tab" data-bs-
target="#nav-read"
```

Продолжение приложения Б

```
type="button" role="tab" aria-controls="nav-profile" aria-
selected="false">
        Прочитано<span class="mx-1 text-secondary">@read.Count()</span>
    </button>
    <button class="nav-link" id="nav-abandoned-reading-tab" data-bs-toggle="tab"
data-bs-target="#nav-abandoned-reading"
        type="button" role="tab" aria-controls="nav-profile" aria-
selected="false">
        Зброшено<span class="mx-1 text-
secondary">@abandonedReading.Count()</span>
    </button>
    <button class="nav-link" id="nav-my-reviews-tab" data-bs-toggle="tab" data-bs-
target="#nav-my-reviews"
        type="button" role="tab" aria-controls="nav-profile" aria-
selected="false">
        Рецензии<span class="mx-1 text-
secondary">@myReviews.Count()</span>
    </button>
</div>
</nav>
<div class="tab-content mt-3" id="nav-tabContent">
    <div class="tab-pane fade show active" id="nav-read-later" role="tabpanel" aria-
labelledby="nav-read-later-tab" tabindex="0">
        <div class="mt-2">
            <ul class="list-unstyled row">
                @{
                    foreach (var book in readLater) {
                        <li class="d-inline " style="width: auto;">
                            <partial name="Partial/_BookItem"
model="book"></partial>
                        </li>
                    }
                    if (readLater.Count() == 0) {
                        <div class="p-1">
                            Пусто
```

Продолжение приложения Б

```

        </div>
    }
}
</ul>
</div>
</div>
<div class="tab-pane fade" id="nav-read-now" role="tabpanel" aria-labelledby="nav-read-
now-tab" tabindex="0">
    <div class="mt-2">
        <ul class="list-unstyled row">
            @ {
                foreach (var book in readNow) {
                    <li class="d-inline " style="width: auto;">
                        <partial name="Partial/_BookItem"
model="book"></partial>
                    </li>
                }
                if (readNow.Count() == 0) {
                    <div class="p-1">
                        Пусто
                    </div>
                }
            }
        </ul>
    </div>
</div>
<div class="tab-pane fade" id="nav-read" role="tabpanel" aria-labelledby="nav-read-tab"
tabindex="0">
    <div class="mt-2">
        <ul class="list-unstyled row">
            @ {
                foreach (var book in read) {
                    <li class="d-inline " style="width: auto;">
                        <partial name="Partial/_BookItem"
model="book"></partial>

```


Продолжение приложения Б

```

        </li>
    }
    if (read.Count() == 0) {
        <div class="p-1">
            Пусто
        </div>
    }
}
</ul>
</div>
</div>
<div class="tab-pane fade" id="nav-abandoned-reading" role="tabpanel" aria-
labelledby="nav-abandoned-reading-tab" tabindex="0">
    <div class="mt-2">
        <ul class="list-unstyled row">
            @ {
                foreach (var book in abandonedReading) {
                    <li class="d-inline " style="width: auto;">
                        <partial name="Partial/_BookItem"
model="book"></partial>
                    </li>
                }
                if (abandonedReading.Count() == 0) {
                    <div class="p-1">
                        Пусто
                    </div>
                }
            }
        </ul>
    </div>
</div>
<div class="tab-pane fade" id="nav-my-reviews" role="tabpanel" aria-labelledby="nav-my-
reviews-tab" tabindex="0">
    <div class="mt-2">
        <ul class="list-unstyled row">
```

Продолжение приложения Б

```
@{
    foreach (var review in myReviews) {
        <li class="d-inline " style="width: auto;">
            <partial name="Partial/_ReviewItem"
model="review"></partial>

        </li>
    }
    if (myReviews.Count() == 0) {
        <div class="p-1">
            Пусто
        </div>
    }
}
</ul>
</div>
</div>
</div>
</div>
```

Views/User/Update.cshtml

```
@{
    ViewData["Title"] = "Редактирование профиля";
}
<h1>@ViewData["Title"]</h1>
@{
    if (@ViewBag.message != null) {
        <div class="alert alert-danger" role="alert">
            @ViewBag.message
        </div>
    }
}
<form asp-controller="User" asp-action="Update" method="post">
    <div class="form_group mt-2 mb-2">
        <label for="userId">ID:</label>
        <input class="form-control" id="userId" readonly value="@ViewBag.user.Id" />
    </div>
</form>
```

Продолжение приложения Б

```
</div>
<div class="form_group mt-2 mb-2">
    <input class="form-control" id="login" name="login" type="email" required
placeholder="Логин" value="@ViewBag.user.Login" />
</div>
<div class="form_group mt-2 mb-2">
    <input class="form-control" id="oldPassword" name="oldPassword" type="password"
required placeholder="Старый пароль" />
</div>
<div class="form_group mt-2 mb-2">
    <input class="form-control" id="newPassword1" name="newPassword1" type="password"
required placeholder="Новый пароль" />
</div>
<div class="form_group mt-2 mb-2">
    <input class="form-control" id="newPassword2" name="newPassword2" type="password"
required placeholder="Подтвердить новый пароль" />
</div>
<div class="form_group mt-2 mb-2">
    <input class="form-control" id="name" name="name" type="text" required
placeholder="Имя" value="@ViewBag.user.Name"/>
</div>
<input class="btn btn-primary mt-2 mb-2" type="submit" value="Сохранить" />
</form>
```

Views/Shared/_Layout.cshtml

```
<!DOCTYPE html>
@{
    var theme = "light";
    //theme = "dark";
}
<html lang="en" data-bs-theme="@theme">
<head>
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>@ ViewData["Title"]</title>
```

Продолжение приложения Б

```
<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha3/dist/css/bootstrap.min.css"
<link rel="stylesheet" href="~/css/site.css" asp-append-version="true" />
<link rel="stylesheet" href="~/Lib.styles.css" asp-append-version="true" />
<style>
    .lib-logo {
        height: 40px;
    }
    @ {
        var filter = "";
        if (@theme == "dark") {
            filter = "invert(80%)";
        }
    }
    .svg {
        filter: @filter;
    }
</style>
</head>
<body>
    <header>
        <nav class="navbar navbar-expand-lg navbar-light border-bottom box-shadow mb-3">
            <div class="container">
                <a class="" asp-area="" asp-controller="Home" asp-action="Index">
                    
                </a>
                <button class="navbar-toggler" type="button" data-bs-toggle="collapse"
data-bs-target=".navbar-collapse" aria-controls="navbarSupportedContent"
                    aria-expanded="false" aria-label="Toggle navigation">
                    <span class="navbar-toggler-icon"></span>
                </button>

                <div class="d-none d-lg-block mx-1"></div>
                <div class="navbar-collapse collapse d-lg-inline-flex justify-content-
between">
```

Продолжение приложения Б

```
<div class="d-lg-none py-1"></div>
<form class="form-inline my-2 my-lg-0" asp-controller="Home"
asp-action="Search" method="get">
    <input class="form-control" type="search" name="q"
placeholder="Поиск" />
</form>
<div class="d-none d-lg-block mx-1"></div>
<ul class="navbar-nav flex-grow-1">
    <li class="nav-item">
        <a class="nav-link " asp-area="" asp-
controller="Home" asp-action="Popular">Популярное</a>
    </li>
    <li class="nav-item">
        <a class="nav-link " asp-area="" asp-
controller="Home" asp-action="New">Новинки</a>
    </li>
    <li class="nav-item">
        <a class="nav-link " asp-area="" asp-
controller="Home" asp-action="Top">Топ-10</a>
    </li>
    <li class="nav-item">
        <a class="nav-link " asp-area="" asp-
controller="Author" asp-action="All">Авторы</a>
    </li>
    <li class="nav-item">
        <a class="nav-link " asp-area="" asp-
controller="Genre" asp-action="All">Жанры</a>
    </li>
    @ {
        int? userId = Context.Session.GetInt32("userId");
        if (userId.HasValue) {
            <li class="nav-item">
                <a class="nav-link" asp-area="" asp-
controller="Note" asp-action="All">Заметки</a>
            </li>
```

Продолжение приложения Б

```

        <li class="nav-item">
            <a class="nav-link" asp-area="" asp-
controller="Featured" asp-action="All">Избранное</a>
        </li>
    }
}
</ul>
<ul class="navbar-nav">
    @{
        <div class="d-lg-none"><hr class="my-1" /></div>
        if (userId.HasValue) {
            <li class="nav-item">
                <a class="nav-link" asp-area="" asp-
controller="User" asp-action="Profile">
                    @{
                        string? userName =
Context.Session.GetString("userName");
                        string? userIsAdmin
= Context.Session.GetString("userIsAdmin");
                        if (userName.Length
> 0) {
                            <div>
                                @userName
                                @ {
                                    if (userIsAdmin == "true") {
                                        <span class="text-warning"> [admin]</span>
                                    }
                                }
                            </div>
                        } else {

```

Продолжение приложения Б

```
<span>Профиль</span>
    }
  }
</a>
</li>
} else {
  <li class="nav-item">
    <a class="nav-link" asp-area="" asp-
controller="Auth" asp-action="Login">Войти</a>
  </li>
  <div class="d-lg-none py-1"></div>
  <li class="nav-item">
    <a class="btn btn-success" asp-
area="" asp-controller="Auth" asp-action="Register">Регистрация</a>
  </li>
  <div class="d-lg-none py-1"></div>
}
}
</ul>
</div>
</div>
</nav>
</header>
<div class="container">
  <main role="main" class="pb-3">
    @RenderBody()
  </main>
</div>
<footer class="footer text-muted" style="bottom: auto">
  <div class="container border-top pt-3 pb-3">
    &copy; 2023 - Lib - <a asp-area="" asp-controller="Home" asp-
action="About">About</a>
  </div>
</footer>
<script src="~/lib/jquery/dist/jquery.min.js"></script>
```

Продолжение приложения Б

```
<script src="~/lib/bootstrap/dist/js/bootstrap.bundle.min.js"></script>
<script src="~/js/site.js" asp-append-version="true"></script>
@await RenderSectionAsync("Scripts", required: false)
</body>
</html>
```

Views/Shared/_AuthorItem.cshtml

```
@model Author
@{
    Author author = Model;
    if (author != null) {
        <div class="border rounded mb-3" style="width: 256px;">
            <a asp-controller="Author" asp-action="One" asp-route-id="@author.Id" class=""
style="">
                
            </a>
            <div class="p-3">
                <a asp-controller="Author" asp-action="One" asp-route-id="@author.Id"
style="text-decoration: none;">
                    <h5 class="text-truncate">@author.Name</h5>
                </a>
            </div>
        </div>
    }
}
```

Views/Shared/_BookItem.cshtml

```
@model Book
@{
    Book book = Model;
    if (book != null) {
        List<Author> authors = null;
        try {
            authors = book.AuthorBooks.Select(ab => ab.Author).ToList();
        } catch {
        }
    }
}
```


Продолжение приложения Б

```

        authors = LibDbContext.Instance.AuthorBooks.Where(ab => ab.BookId ==
book.Id).Select(ab => ab.Author).ToList();
    }
    <div class="border rounded mb-3" style="width: 256px;">
        <a asp-controller="Book" asp-action="One" asp-route-id="@book.Id" class=""
style="">

        </a>
        <div class="p-3">
            <a asp-controller="Book" asp-action="One" asp-route-id="@book.Id"
style="text-decoration: none;">

                <h5 class="text-truncate">@book.Name</h5>

            </a>
            <div class="text-truncate">
                @ {
                    for (int i = 0; i < authors.Count; i++) {
                        <a asp-controller="Author" asp-action="One" asp-
route-id="@authors[i].Id" class="" style="text-decoration: none;">

                            @authors[i].Name

                        </a>
                        if (i != authors.Count - 1) {
                            <span class=" ">, </span>
                        }
                    }
                }
            </div>
            @ {
                <h5 class="py-1">

                    <div class="align-baseline d-inline">

                        @ {
                            if (book.AvgRating.HasValue) {

```

Продолжение приложения Б

```
inline">@book.AvgRating.Value.ToString("f")</span>
                                } else {
                                <span class="d-inline">нет
оценок</span>
                                }
                            }
                        </div>
                    </h5>
                }
            </div>
        </div>
    }
}
```

Views/Shared/_ReviewItem.cshtml

```
@model Review
@{
    Review review = Model;
    if (review != null) {
        <div class="border rounded mb-3 p-2" style="width: 512px;">
            <h5>
                <a asp-controller="User" asp-action="One" asp-route-id="@review.UserId"
class="" style="text-decoration: none;">
                    @review.UserName
                </a>
            </h5>
            <h5 class="text-truncate">
                <span class="text-secondary">На книгу</span>
                <a asp-controller="Book" asp-action="One" asp-route-
id="@review.Book.Id" class="" style="text-decoration: none;">
                    @review.Book.Name
                </a>
            </h5>
            <div class="text-secondary">
```

Продолжение приложения Б

```
@review.StrDateOfCreation
</div>
<h5 class="py-1">
    
    <span class="align-baseline d-inline">@review.Rating</span>
</h5>
<div class="text-truncate pb-1">
    @review.Content
</div>
<div class="text-secondary pb-1">
    Лайков - @review.Likes.Count
</div>
</div>
}
}
```

Views/Shared/_SavedToSessionItem.cshtml

```
@using Microsoft.EntityFrameworkCore;
@{
    List<Book> books = new List<Book>();
    int i = 0;
    int? savedBookId = Context.Session.GetInt32($"savedBook_{i}_id");
    while (savedBookId.HasValue) {
        Book findedBook = LibDbContext.Instance.Books
            .Include(b => b.AuthorBooks)
            .ThenInclude(ab => ab.Author)
            .FirstOrDefault(b => b.Id == savedBookId.Value);
        if (findedBook != null) {
            books.Add(findedBook);
        } else {
            Context.Session.Remove($"savedBook_{i}_id"); // мб ее удалили
        }
        i++;
        savedBookId = Context.Session.GetInt32($"savedBook_{i}_id");
    }
}
```

Продолжение приложения Б

```
}
List<Author> authors = new List<Author>();
int j = 0;
int? savedAuthorId = Context.Session.GetInt32($"savedAuthor_{j}_id");
while (savedAuthorId.HasValue) {
    Author findedAuthor = LibDbContext.Instance.Authors
        .Include(b => b.AuthorBooks)
        .ThenInclude(ab => ab.Author)
        .FirstOrDefault(a => a.Id == savedAuthorId.Value);
    if (findedAuthor != null) {
        authors.Add(findedAuthor);
    } else {
        Context.Session.Remove($"savedAuthor_{j}_id"); // мб ее удалили
    }
    j++;
    savedAuthorId = Context.Session.GetInt32($"savedAuthor_{j}_id");
}
}
<div class="">
    @{
        bool isBooksExist = books.Count != 0 ? true : false;
        bool isAuthorsExist = authors.Count != 0 ? true : false;
        if (isAuthorsExist || isBooksExist) {
            <h2>Недавно смотрели</h2>
            <nav class="mt-3">
                <div class="nav nav-tabs" id="nav-tab" role="tablist">
                    @{
                        if (isBooksExist) {
                            <button class="nav-link active" id="nav-saved-
books-tab" data-bs-toggle="tab" data-bs-target="#nav-saved-books"
                                type="button" role="tab" aria-
controls="nav-home" aria-selected="true">
                                Книги
                            </button>
                        }
                    }
                }
            }
        }
    }
</div>
```

Продолжение приложения Б

```

        if (isAuthorsExist && !isBooksExist) {
            <button class="nav-link active" id="nav-saved-
authors-tab" data-bs-toggle="tab" data-bs-target="#nav-saved-authors"
                                type="button" role="tab" aria-
controls="nav-profile" aria-selected="true">
                Авторы
            </button>
        } else if (isAuthorsExist) {
            <button class="nav-link" id="nav-saved-authors-
tab" data-bs-toggle="tab" data-bs-target="#nav-saved-authors"
                                type="button" role="tab" aria-
controls="nav-profile" aria-selected="false">
                Авторы
            </button>
        }
    }
</div>
</nav>
<div class="tab-content mt-3 mb-5" id="nav-tabContent">
    @ {
        if (isBooksExist) {
            <div class="tab-pane fade show active" id="nav-saved-
books" role="tabpanel" aria-labelledby="nav-books-tab" tabindex="0">
                <div class="overflow-x-scroll">
                    <ul class="list-unstyled row flex-nowrap
mb-0">
                        @ {
                            if (books.Count != 0) {
                                foreach (var book in
books) {
                                    <li class="d-
inline " style="width: auto;">
                                        <partial name="_BookItem" model="book"></partial>
                                    </li>

```

Продолжение приложения Б

```

    }
  }
}

</ul>
</div>
</div>
}
if (isAuthorsExist && !isBooksExist) {
  <div class="tab-pane fade show active" id="nav-saved-
authors" role="tabpanel" aria-labelledby="nav-authors-tab" tabindex="0">
    <div class="overflow-x-scroll">
      <ul class="list-unstyled row flex-nowrap
mb-0">
        @{
          if (authors.Count != 0) {
            foreach (var author
in authors) {
              <li class="d-
inline " style="width: auto;">

                <partial name="_AuthorItem" model="author"></partial>

              </li>
            }
          }
        }
      </ul>
    </div>
  </div>
} else {
  <div class="tab-pane fade" id="nav-saved-authors"
role="tabpanel" aria-labelledby="nav-authors-tab" tabindex="0">
    <div class="overflow-x-scroll">
      <ul class="list-unstyled row flex-nowrap
mb-0">
        @{

```

Продолжение приложения Б

```
in authors) {
    inline " style="width: auto;">
        <partial name="_AuthorItem" model="author"></partial>
    </li>
}
}
}
</ul>
</div>
</div>
}
</div>
}
</div>
```

```
if (authors.Count != 0) {
    foreach (var author
```

Приложение В

Результат работы веб-приложения

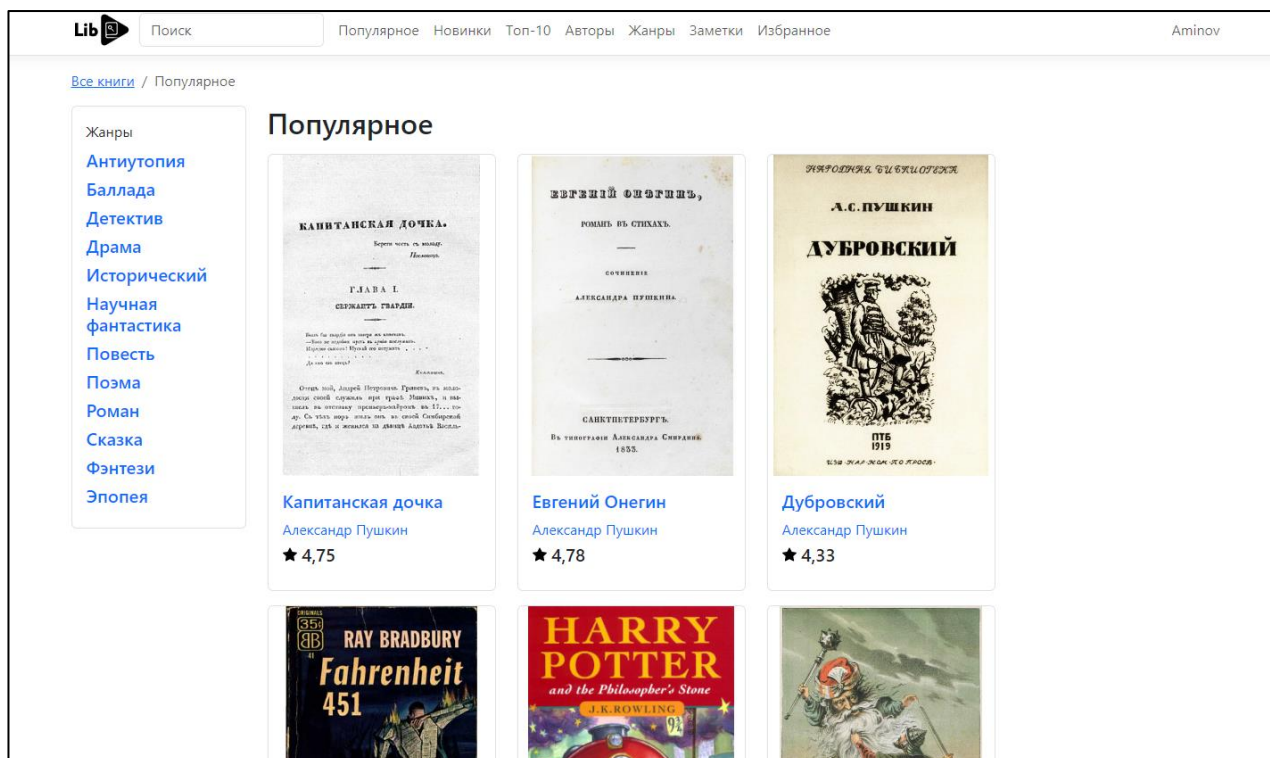


Рисунок В.1 – Список популярных книг

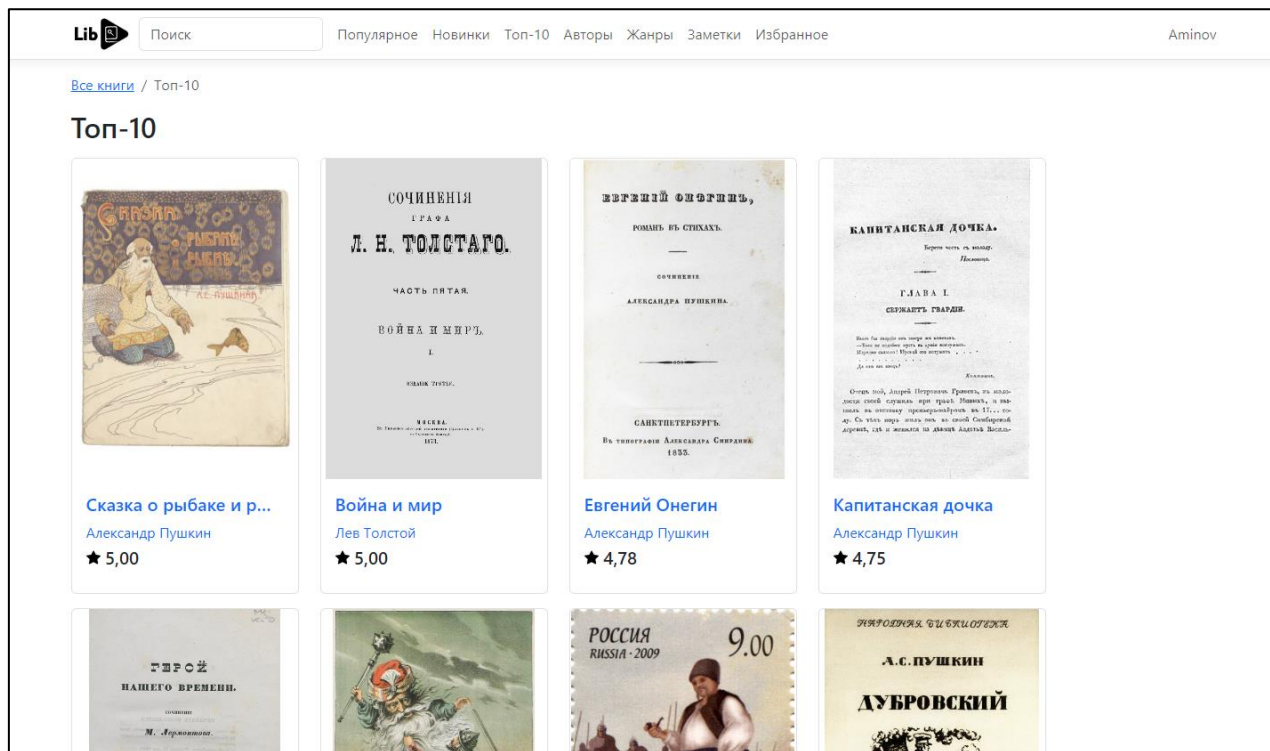


Рисунок В.2 –Список книг с самыми высокими оценками

Lib

Поиск

Популярное

Новинки

Топ-10

Авторы

Жанры

Заметки

Избранное

Aminov

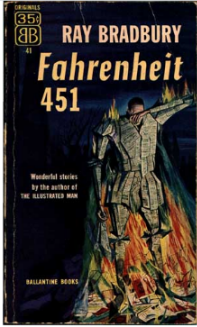
Избранное

Прочитать позже 1

Читаю 1

Прочитано 2

Заброшено 0



451 градус по Фаренг...

Рэй Брэдбери

★ 4,33

Рисунок В.3 – Список книг пользователя

СПИСОК СОКРАЩЕНИЙ

ПК – Персональный компьютер

СУБД – Система управления базами данных

ЭВМ – Электронно-вычислительная машина

ОЗУ – Оперативное запоминающее устройство

					40.А-2075-2023 09.02.07 ДП-ПЗ	Лист
						171
Изм.	Лист	№ докум.	Подпись	Дата		

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

- 1 ГОСТ 19.701-90 ЕСПД. Схемы алгоритмов, программ, данных и систем. Обозначения условные и правила выполнения [Текст]. – Введ. 1992–01–01. – М.: Изд-во стандартов, 1992. – 24 с.
- 2 ГОСТ 2.105 - 95. Общие требования к текстовым документам. Единая система конструктивной документации [Текст]. – Введ. 1996 - 07 - 01. – М.: Стандартиформ, 2007.
- 3 ГОСТ 34.601-90 Информационная технология. Комплекс стандартов на автоматизированные системы. Автоматизированные системы. Стадии создания [Текст]. – Введ. 1990 - 01 - 01. – М.: Стандартиформ, 2007.
- 4 ГОСТ Р 51583-2014. Защита информации. Порядок создания автоматизированных систем в защищенном исполнении. Общие положения. [Текст]. – Введ. 2014- 09 - 01. – М.: Стандартиформ, 2007.
- 5 Албахари, Д., Албахари Б. С# 8.0. Карманный справочник — М.: Диалектика, 2020. — 240 с. (дата обращения: 17.05.2023) — Текст: непосредственный.
- 6 Гордеев, С. И. Организация баз данных в 2 ч. Часть 1 : учебник для среднего профессионального образования / С. И. Гордеев, В. Н. Волошина. — 2-е изд., испр. и доп. — Москва : Издательство Юрайт, 2023. — 310 с. — (Профессиональное образование). — ISBN 978-5-534-11626-7. — Текст: электронный // Образовательная платформа Юрайт [сайт]. — URL: <https://urait.ru/bcode/518510> (дата обращения: 17.05.2023).
- 7 Гордеев, С. И. Организация баз данных в 2 ч. Часть 2 : учебник для вузов / С. И. Гордеев, В. Н. Волошина. — 2-е изд., испр. и доп. — Москва : Издательство Юрайт, 2023. — 513 с. — (Высшее образование). — ISBN 978-5-534-04470-6. — Текст: электронный // Образовательная платформа Юрайт [сайт]. — URL: <https://urait.ru/bcode/515097> (дата обращения: 17.05.2023).
- 8 Зыков, С. В. Программирование. Объектно-ориентированный подход:

учебник и практикум для вузов / С. В. Зыков. — Москва : Издательство Юрайт, 2023. — 155 с. — (Высшее образование). — ISBN 978-5-534-00850-0. — Текст: электронный // Образовательная платформа Юрайт [сайт]. — URL: <https://urait.ru/bcode/512425> (дата обращения: 17.05.2023).

9 Маркин, А. В. Программирование на SQL : учебное пособие для среднего профессионального образования / А. В. Маркин. — Москва : Издательство Юрайт, 2023. — 435 с. — (Профессиональное образование). — ISBN 978-5-534-11093-7. — Текст: электронный // Образовательная платформа Юрайт [сайт]. — URL: <https://urait.ru/bcode/518166> (дата обращения: 17.05.2023).

10 Стасышин, В. М. Базы данных: технологии доступа : учебное пособие для среднего профессионального образования / В. М. Стасышин, Т. Л. Стасышина. — 2-е изд., испр. и доп. — Москва : Издательство Юрайт, 2023. — 164 с. — (Профессиональное образование). — ISBN 978-5-534-09888-4. — Текст: электронный // Образовательная платформа Юрайт [сайт]. — URL: <https://urait.ru/bcode/516927> (дата обращения: 17.05.2023).

11 Стружкин, Н. П. Базы данных: проектирование. Практикум : учебное пособие для среднего профессионального образования / Н. П. Стружкин, В. В. Годин. — Москва : Издательство Юрайт, 2023. — 291 с. — (Профессиональное образование). — ISBN 978-5-534-08140-4. — Текст: электронный // Образовательная платформа Юрайт [сайт]. — URL: <https://urait.ru/bcode/516929> (дата обращения: 17.05.2023).

12 Сырых, Ю. А. Современный веб-дизайн. Настольный и мобильный. — М.: Диалектика, 2019. — 384 с. — URL: <https://monster-book.com/reader/25613> (дата обращения: 17.05.2023) — Текст: электронный.

13 Сысолетин, Е. Г. Разработка интернет-приложений : учебное пособие для среднего профессионального образования / Е. Г. Сысолетин, С. Д. Ростунцев. — Москва : Издательство Юрайт, 2023. — 90 с. — (Профессиональное образование). — ISBN 978-5-534-10015-0. — Текст: электронный // Образовательная платформа Юрайт [сайт]. — URL: <https://urait.ru/bcode/517538>

					40.А-2075-2023 09.02.07 ДП-ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		173

(дата обращения: 17.05.2023).

14 Фримен, А. Entity Framework Core 2 для ASP.NET Core MVC для профессионалов — М.: Диалектика, 2019. — 624 с. (дата обращения: 17.05.2023) — Текст: непосредственный.

15 Эспозито, Д. Разработка веб-приложений с использованием ASP.NET и AJAX / Д. Эспозито. - СПб.: Питер, 2019. - 240 с. – URL: <https://www.chitalkino.ru/esposito-d/razrabotka-veb-prilozheniy-s-ispolzovaniem-asp/> (дата обращения: 17.05.2023) — Текст: электронный.

16 DotNet Microsoft. Язык программирования C#. dotnet.microsoft.com/en-us/languages/csharp/ (дата обращения: 17.05.2023). — Текст: электронный.

17 Geeks for Geeks : Портал компьютерных наук. URL: <https://www.geeksforgeeks.org> (дата обращения: 17.05.2023). – Текст: электронный.

18 Learn Microsoft. Сайт о программирование. learn.microsoft.com/ru-ru/dotnet/csharp/ (дата обращения: 17.05.2023). — Текст: электронный.

19 Metanit : Сайт о программировании. URL: <https://metanit.com/> (дата обращения: 17.05.2023). – Текст: электронный.

20 StackOverFlow : Система вопросов и ответов о программировании. URL: <https://ru.stackoverflow.com/> (дата обращения: 17.05.2023). – Текст: электронный.

21 Q&A Habr : Сервис вопросов и ответов об IT. URL: <https://qna.habr.com/> (дата обращения: 17.05.2023). – Текст : электронный.