МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования «БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет Информационных Технологий

Кафедра Информационных систем и технологий

Специальность 1-40 05 01 «Информационные системы и технологии»

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

**К КУРСОВОМУ ПРОЕКТУ НА ТЕМУ:**

Веб-приложение для управления библиотекой книг

Выполнил студент Халалеенко Андрей Николаевич

(Ф.И.О.)

Руководитель проекта

(учен. степень, звание, должность, подпись, Ф.И.О.)

Заведующий кафедрой

(учен. степень, звание, должность, подпись, Ф.И.О.)

Консультанты ассистент Кантарович В.С.

(учен. степень, звание, должность, подпись, Ф.И.О.)

Нормоконтролер ассистент Кантарович В.С.

(учен. степень, звание, должность, подпись, Ф.И.О.)

Курсовой проект защищен с оценкой

Минск 2023

**Содержание**

**Введение**

Современное информационное общество требует доступа к высококачественной литературе, и управление большими библиотеками книг может представлять сложность без эффективных инструментов и ресурсов. В свете этой проблемы становится актуальным и востребованным разработка функционального веб-приложения для управления библиотекой книг.

Целью данного курсового проекта является создание такого веб-приложения, которое позволит пользователям эффективно управлять библиотекой книг. Главная цель заключается в разработке полнофункционального приложения, обеспечивающего возможность регистрации и авторизации пользователей, поддержку ролей администратора и пользователя, добавление новых книг и информации о них, чтение и скачивание книг, редактирование информации, поиск книг по различным критериям, просмотр информации о книгах, оставление отзывов и оценок, а также создание личных списков прочитанных книг, списка «хочу прочитать» и списка «избранное», а также «прочитанное» и «читаю сейчас». Для достижения этих целей были выполнены следующие задачи:

* постановка задачи;
* проектирование веб-приложения;
* разработка веб-приложения;
* тестирование веб-приложения;
* составление руководства пользователя.

Для разработки веб-приложения были выбраны современные технологии и инструменты, которые позволили скорость разработки и удобство использования приложения.

В данном проекте на стороне сервера были использованы два популярных инструмента: Express и Socket.IO. Для хранения данных была использована база данных MySQL, которая является одной из наиболее распространенных реляционных баз данных. Для работы с базой данных MySQL в проекте был использован ORM-фреймворк Prisma. Таким образом, Express обеспечивает обработку HTTP-запросов и маршрутизацию на стороне сервера, Socket.IO обеспечивает двунаправленную связь в режиме реального времени между клиентом и сервером, MySQL служит в качестве хранилища данных, а Prisma упрощает взаимодействие с базой данных и выполнение запросов. Эти инструменты совместно позволяют разрабатывать функциональное и эффективное веб-приложение для управления библиотекой книг.

На стороне клиента для разработки пользовательского интерфейса был использован React Bootstrap. Bootstrap — это открытый и бесплатный HTML, CSS и JS фреймворк, который используется веб-разработчиками для быстрой вёрстки адаптивных дизайнов сайтов и веб-приложений.

1. Постановка задачи веб-приложения

Изучая, аналогичные веб-приложения, были найдены схожие по теме и функционалу приложения. Большая часть веб-приложений похожи друг на друга, и функционалом почти не отличаются. Так же были найдены статьи, которые помогли реализовать некоторые функции. Результат анализа представлен ниже.

## **Обзор аналогов веб приложения**

Goodreads.com – это популярная онлайн-платформа для любителей чтения книг. Она предоставляет возможность пользователям искать, открывать и оценивать книги. Goodreads.com также позволяет пользователям составлять личные списки прочитанных книг, добавлять отзывы и рекомендации, а также общаться с другими читателями через сообщества и группы обсуждения. Сайт предлагает разнообразные функции для удобного взаимодействия с книжным контентом и обмена литературными впечатлениями.

Интерфейс этого сайта можно увидеть на рисунке 1.1.

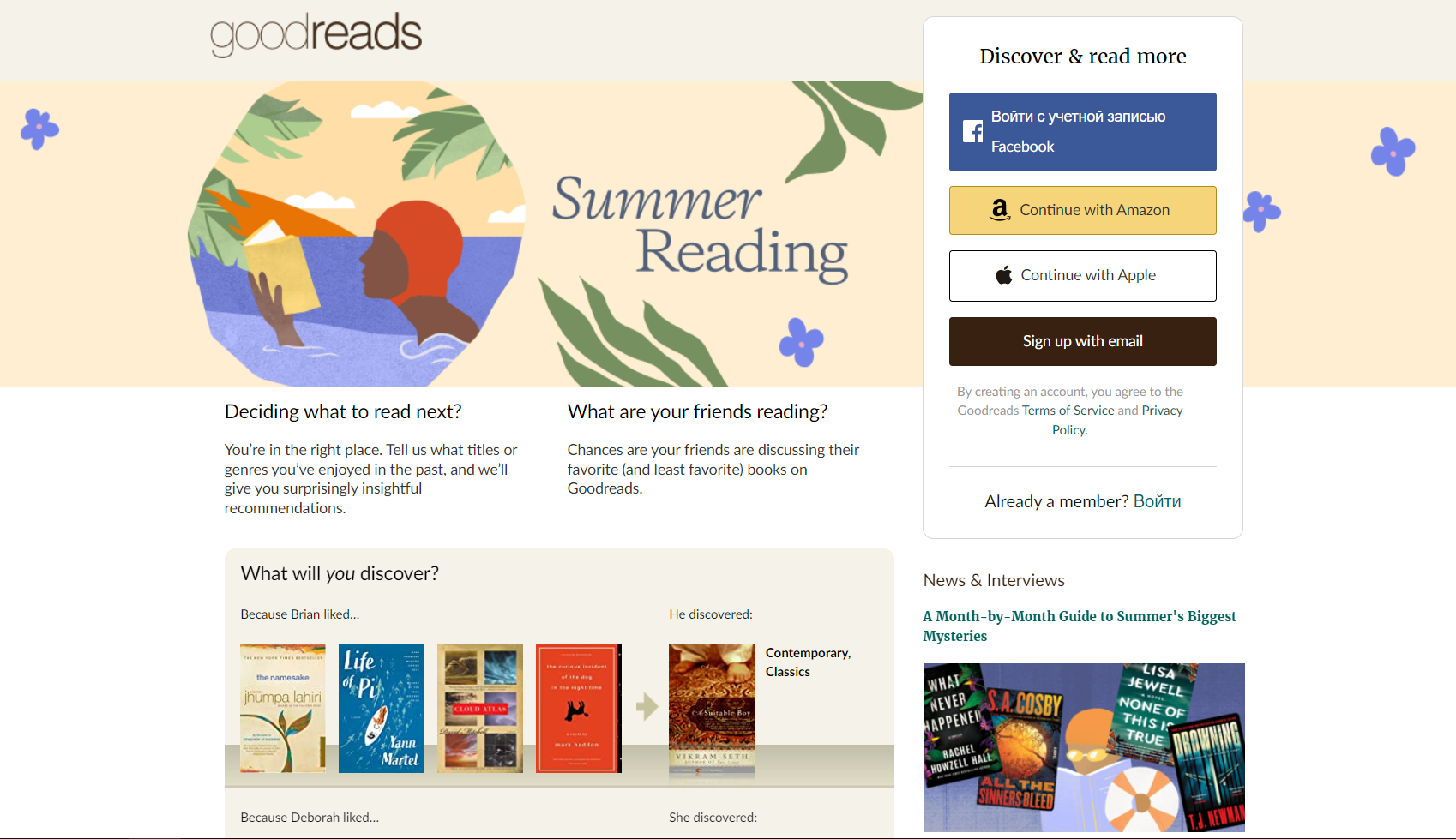


Рисунок 1.1 – Домашняя страница Goodreads

Преимущества данного сайта представлены ниже:

* большой каталог книг;
* сообщество чтения: Goodreads предоставляет возможность вступить в сообщество чтения, где пользователи могут обмениваться рекомендациями, отзывами и обсуждать книги с другими читателями;
* личные списки и отслеживание прочитанного. Пользователи могут создавать личные списки прочитанных книг, отмечать книги, которые они хотят прочитать, и отслеживать свой прогресс чтения. Это помогает пользователям организовать свою литературную активность и следить за своими литературными интересами.

Project Gutenberg - это онлайн-библиотека, которая предоставляет бесплатный доступ к большой коллекции электронных книг. Сайт B характеризуется огромным выбором литературных произведений разных жанров, включая классику, научно-популярные и художественные книги. Пользователи могут бесплатно скачивать книги в различных форматах, таких как EPUB, MOBI, PDF и др. Project Gutenberg сосредоточен на предоставлении доступа к общественному достоянию литературы и является важным ресурсом для чтения книг в электронном формате.

Интерфейс этого сервиса можно увидеть на рисунке 1.2.



Рисунок 1.2 – Домашняя страница Project Gutenberg

В целом, Project Gutenberg предоставляет бесплатный и удобный доступ к большой коллекции книг в общественном достоянии, что делает его ценным ресурсом для чтения и распространения литературного наследия. Рассмотрим некоторые приемущества.

Project Gutenberg имеет несколько преимуществ:

* бесплатный доступ;
* общественное достояние;
* разнообразие книг. Коллекция Project Gutenberg сосредоточена на произведениях, находящихся в общественном достоянии. Это включает классические литературные работы, которые больше не защищены авторским правом. Пользователи могут получить доступ к произведениям, которые иначе могли бы быть недоступны или требовали бы платы;
* различные форматы. Книги в Project Gutenberg доступны в различных форматах, включая EPUB, MOBI, PDF и другие. Это позволяет пользователям выбирать формат, который наиболее удобен для чтения на их устройствах, будь то электронные книги, смартфоны или планшеты.

## **Постановка задач веб-приложения**

На основе проведенного анализа и учета требований, указанных в задании на курсовое проектирование, для разработки функционального веб-приложения по управлению библиотекой книг могут быть поставлены следующие задачи:

* разработка системы регистрации и авторизации: задача состоит в создании функционала, позволяющего пользователям регистрироваться на сайте, авторизовываться и сохранять их учетные данные в безопасном виде;
* создание системы ролей: задача заключается в реализации механизма, позволяющего выделять роли администратора и пользователя. Администратору предоставляются дополнительные привилегии для управления книгами и данными библиотеки;
* разработка функционала добавления и редактирования книг: задача состоит в создании возможности добавления новых книг в библиотеку с указанием соответствующей информации (название, автор, жанр и др.). Также необходимо реализовать функционал редактирования информации о существующих книгах;
* реализация функционала чтения и скачивания книг: задача заключается в создании механизма, позволяющего пользователям читать книги онлайн и скачивать их в удобном формате;
* разработка системы поиска книг: задача состоит в создании возможности поиска книг в библиотеке по различным критериям, таким как название, автор, жанр и т.д. Это позволит пользователям быстро находить интересующие их книги;
* реализация функционала отзывов и оценок: задача заключается в создании механизма, позволяющего пользователям оставлять отзывы на книги и ставить им оценки. Также необходимо предусмотреть возможность просмотра отзывов и оценок других пользователей;
* разработка функционала личных списков: задача состоит в создании функционала, позволяющего пользователям создавать личные списки прочитанных книг, списки «хочу прочитать» и «избранное», а также «читаю сейчас» и «прочитанные». Это позволит пользователям организовать и управлять своими литературными предпочтениями;

## **Вывод по разделу**

В результате анализа аналогичных веб-приложений и постановки задач были определены ключевые требования для разработки системы управления библиотекой книг. Изучение существующих решений позволило выделить функции, повышающие удобство и эффективность работы с приложением: регистрацию и авторизацию, поддержку ролей, добавление и редактирование книг, поиск, а также чтение и скачивание книг. Реализация личных списков, отзывов и оценок дополнительно позволяет пользователям персонализировать взаимодействие с библиотекой. На основе собранных данных определены задачи для создания полнофункционального приложения, отвечающего современным требованиям библиотечных систем.

1. Проектирование веб-приложения

Проект, разработанный в рамках данной работы, представляет собой клиент-серверную архитектуру, включающую два основополагающие части: клиентскую и серверную. Клиентская часть отвечает за взаимодействие с пользователем, предоставляет интерфейс для ввода данных и отображения результатов, в то время как серверный компонент обеспечивает обработку запросов, взаимодействие с базой данных и логику бизнес-процессов. Такое разделение функциональности позволяет эффективно управлять ресурсами и обеспечить надежную работу приложения.

## **Архитектура веб-приложения**

Общая структура проекта клиентская часть и серверная.

Проект состоит из клиентской и серверной частей. Клиент представляет собой веб-интерфейс, построенный с использованием библиотеки React Bootstrap. Он позволяет клиенту отправлять запросы на сервер и отображать полученные ответы. На рисунке 2.1 предложена схема-развертывания приложения.

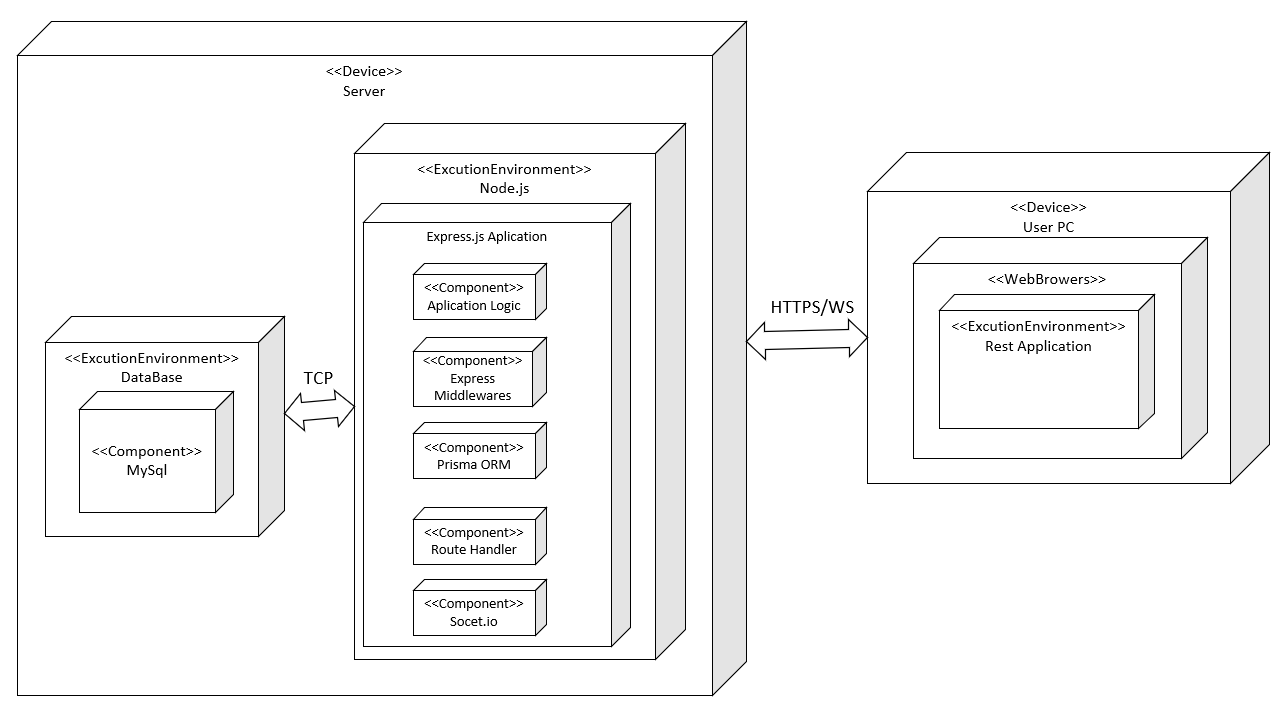


Рисунок 2.1 – Структурная схема приложения

Серверная часть реализована с использованием фреймворка Express. Он обрабатывает запросы, поступающие от клиента, и использует Prisma для взаимодействия с базой данных.

Взаимодействие между клиентом и сервером осуществляется по протоколу HTTPS. Клиент отправляет запросы на определенные маршруты сервера, а сервер обрабатывает эти запросы и возвращает соответствующие ответы.

React Bootstrap обеспечивает создание пользовательского интерфейса с использованием готовых компонентов, стилей и сеток, что упрощает разработку и создание современного дизайна веб-приложения.

Серверная часть использует различные промежуточные компоненты, такие как CORS для обработки запросов с разных источников, ErrorHandlerMiddleware для обработки ошибок. Кроме того, сервер использует пакет express-fileupload для обработки загрузки файлов, что позволяет клиентской части скачивать, читать книги, а другие фалы, по типу обложки книг и аватар пользователя.

Связь между клиентом и сервером в режиме реального времени реализована с использованием библиотеки Socket.IO. Она позволяет установить двустороннее соединение между клиентом и сервером, обеспечивая обновление данных на клиентской стороне без необходимости повторных запросов.

В результате, данная структура программного продукта обеспечивает эффективное и отзывчивое веб-приложение для управления отзывами и обработки данных из базы данных.

## **Описание работы веб-приложения**

При запуске программы, сервер настраивается для прослушивания входящих запросов на определенном порту. Клиентское приложение загружается в веб-браузере пользователя, и пользователь может взаимодействовать с интерфейсом, предоставленным React Bootstrap.

Когда пользователь выполняет определенные действия, например, отправляет запрос на добавление закладки книги, клиентское приложение формирует соответствующий HTTPS-запрос и отправляет его на сервер.

Сервер, используя фреймворк Express. Express представляет собой популярный веб-фреймворк, написанный на JavaScript и работающий внутри среды исполнения node.js. Этот модуль освещает некоторые ключевые преимущества этого фреймворка, установку среды разработки и выполнение основных задач веб-разработки и развёртывания. принимает запрос и маршрутизирует его к соответствующему обработчику маршрута. Обработчик маршрута выполняет необходимые операции, например, создает новую задачу в базе данных с использованием Prisma или рабаты с файлами. При успешном выполнении операции, сервер формирует ответ и отправляет его обратно клиентскому приложению.

Клиентское приложение получает ответ от сервера и обновляет свой интерфейс, отображая актуальные данные или информацию об успешном выполнении операции. В случае возникновения ошибок, сервер использует middleware для обработки и отправки соответствующего сообщения об ошибке обратно клиентскому приложению.

Кроме того, взаимодействие между клиентом и сервером в режиме реального времени осуществляется с использованием Socket.IO. Например, при появлении новой задачи, сервер отправляет уведомление о ней клиентскому приложению через WebSocket-соединение, и клиентское приложение мгновенно отображает это уведомление пользователю.

Данная диаграмма показывает, что клиентское приложение взаимодействует с сервером через протокол HTTPS, а также осуществляет обмен данными с БД. Такая структура программного продукта позволяет пользователям взаимодействовать с сервером, отправлять запросы и получать ответы, а также обновлять данные в режиме реального времени с помощью сокетов. Это создает эффективное и отзывчивое веб-приложение для управления задачами и обработки данных из базы данных.

## **Описание диаграммы использования**

Диаграмма использования разработана для визуального представления функционала веб-приложения для управления библиотекой книг. На диаграмме представлены основные роли пользователей системы и соответствующие им функции, что позволяет наглядно продемонстрировать, какие действия доступны каждой роли и как эти действия взаимосвязаны.

Диаграмма использования включает следующие основные роли:

* гость – роль, представляющая пользователей, не прошедших авторизацию. Гости имеют доступ к базовым функциям, таким как регистрация и авторизация в приложении;
* пользователь – авторизованный участник системы, который наследует функционал роли гостя и обладает расширенными возможностями, такими как чтение и скачивание книг, управление личными списками, оставление отзывов и оценок, а также использование закладок;
* администратор – привилегированная роль, наследующая возможности пользователя, но также обладающая функциями для управления книгами, жанрами, тегами и авторами;
* автор – роль, похожая на администратора, наследующая функции гостя и позволяющая управлять книгами, но не имеющая доступа к управлению жанрами и тегами.

Основной функционал приложения можно наглядно увидеть на рисунке 2.2.

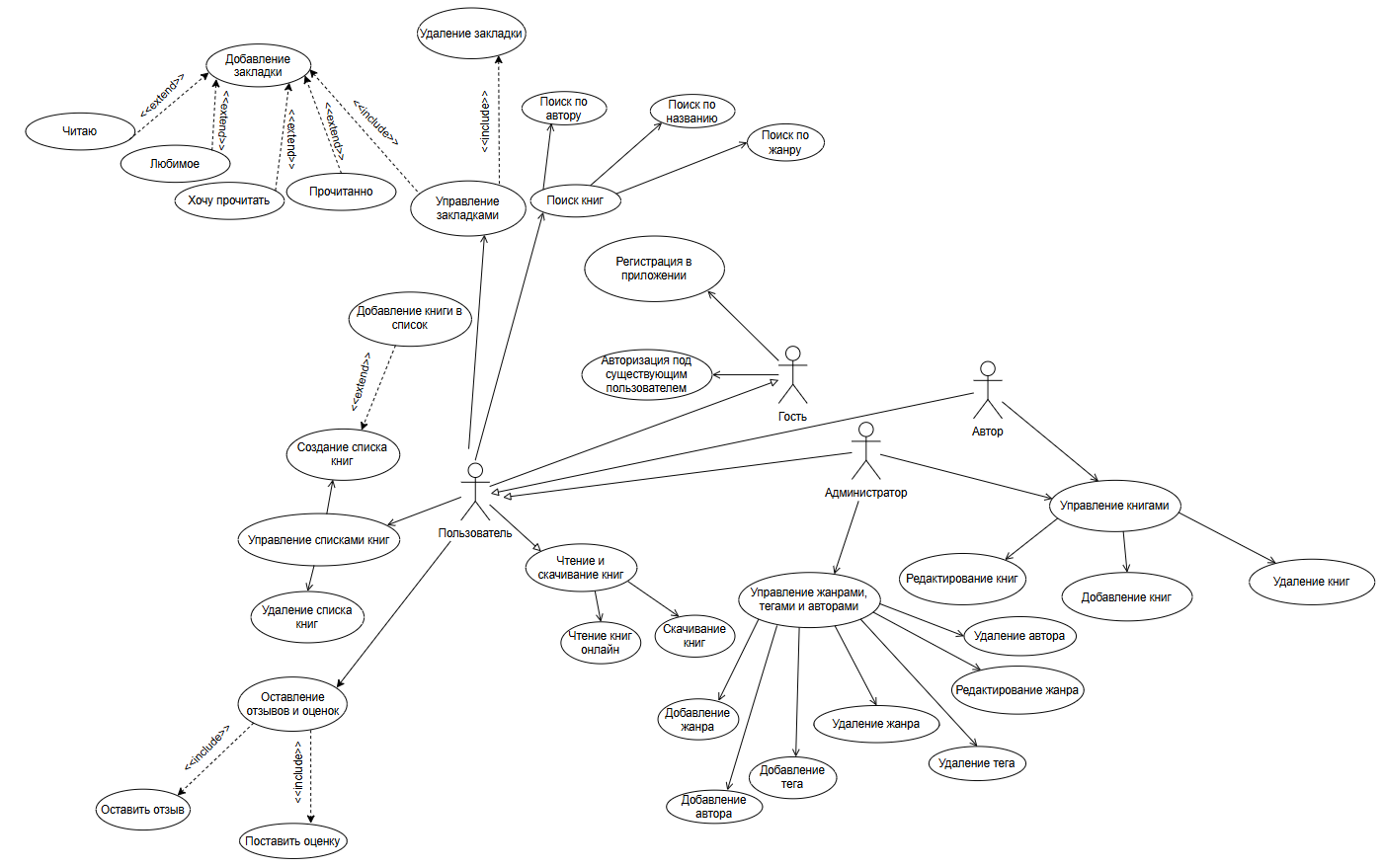


Рисунок 2.2 – Use case-диаграмма

Диаграмма подробно описывает следующие функции и взаимосвязи между ролями и действиями:

* регистрация и авторизация: гость имеет доступ к функциям регистрация в приложении и авторизация под существующим пользователем. Это позволяет новым пользователям зарегистрироваться в системе и существующим пользователям – авторизоваться;
* управление книгами: администратор и автор имеют доступ к блоку управление книгами, в который входят подфункции добавление книги, редактирование книги и удаление книги. Это позволяет администратору и автору добавлять, редактировать и удалять книги в библиотеке;
* чтение и скачивание книг: пользователь может использовать функции чтение книг онлайн и скачивание книг. Это дает возможность авторизованным пользователям читать книги прямо в приложении или загружать их для офлайн-просмотра;
* поиск книг: пользователь имеет доступ к блоку поиск книг, который включает подфункции поиск по названию, поиск по автору и поиск по жанру. Эти функции позволяют быстро находить книги в библиотеке, используя различные критерии;
* управление жанрами, тегами и авторами: администратор имеет доступ к блоку управление жанрами, тегами и авторами, включающему функции добавление жанра, удаление жанра, добавление тега, удаление тега, добавление автора и удаление автора. Это предоставляет администратору возможность настраивать категории и атрибуты для книг, облегчая пользователям поиск и сортировку;
* оставление отзывов и оценок: пользователь может использовать функции оставить отзыв и поставить оценку для книг, доступных через блок оставление отзывов и оценок. Эти действия включены с помощью <<include>>, поскольку для оставления полного отзыва и оценки оба действия должны быть выполнены;
* управление списками книг: пользователь имеет доступ к блоку управление списками книг, который включает подфункции для создания различных списков: прочитанные, хочу прочитать, избранное и читаю сейчас. Каждый из этих списков расширен функцией добавление книги в список с помощью <<extend>>, указывая, что книги могут быть добавлены в каждый список по выбору пользователя.

## **Проектирование базы данных веб-приложения**

После проведения аналогов данного проекта, была спроектирована база данных, для реализации всех поставленных задач. Данная СУБД выбрана в качестве преимущественной из-за своей простоты создания таблиц и широкой поддержки веб-приложений. MySql обладает простым и понятным языком запросов (SQL) и хорошо интегрируется с фреймворком Prisma, который используется в серверной части проекта.

С использованием MySql, серверная часть может эффективно управлять данными, хранить информацию о пользователях, задачах и других сущностях, необходимых для работы приложения. Создание таблиц и определение связей между ними происходит в соответствии с моделью данных, разработанной для проекта.

Схема созданного БД для данного проекта представлена на рисунке 2.3.

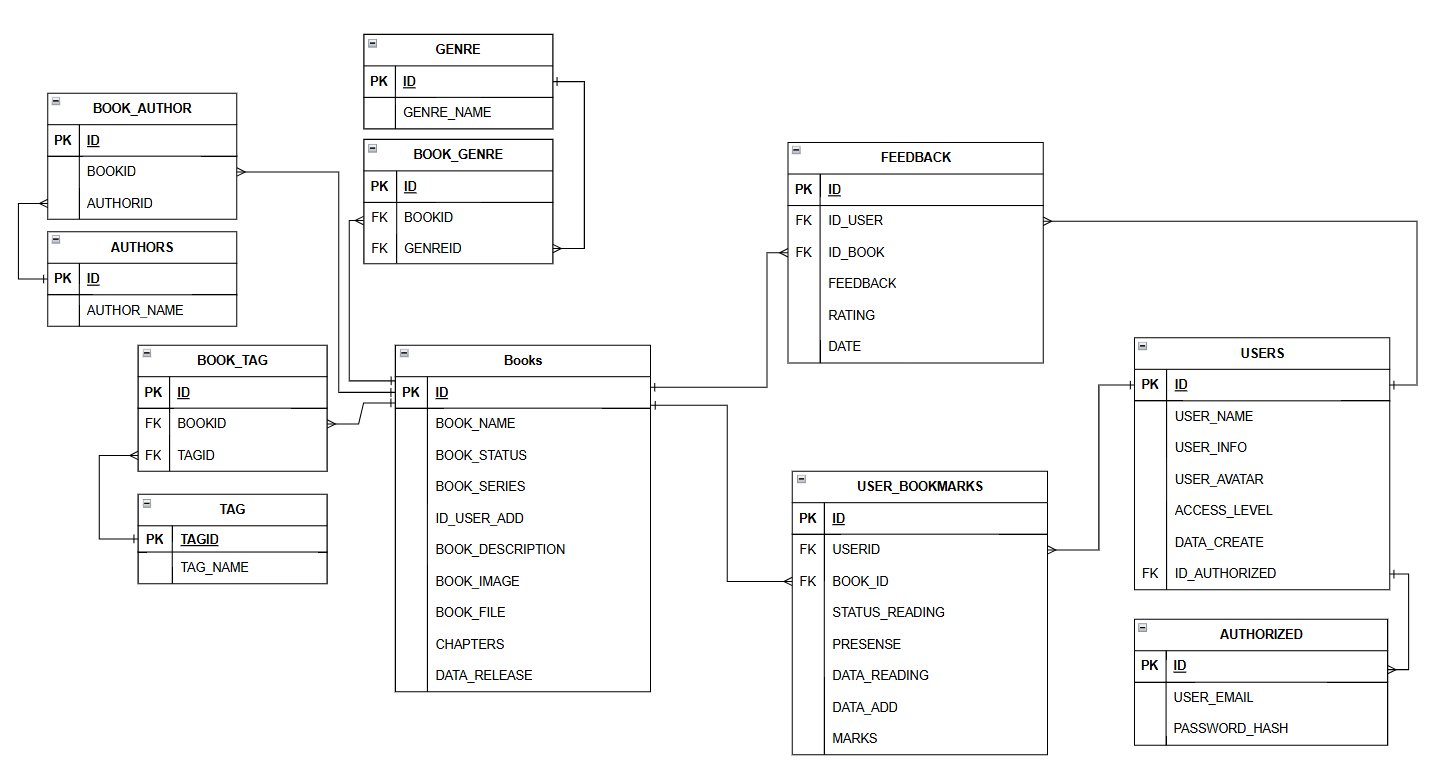


Рисунок 2.3 – Модель базы данных

Основой инфраструктуры базы данных является грамотно спроектированная модель, которая отображает связь пользовательских таблиц. Правильное и корректное взаимодействие их друг с другом как раз и заключается в схеме базы данных со связями, верно отображающими их положение.

Далее будет описана каждая таблица базы данных.

Описание структуры таблицы «users» представлено в таблице 2.1.

Таблица 2.1 – Структура таблицы «users»

|  |  |  |
| --- | --- | --- |
| Название столбца | Тип, ограничение целостности | Описание столбца |
| id | int, primary key | Идентификатор пользователя |
| EMAIL | varchar(50),unique | Электронная почта пользователя |
| pass | varchar(500) | Хеш пароля пользователя |
| ACCESS\_LEVEL | varchar(20) | Роль пользователя |
| USER\_NAME | varchar(20) | Имя пользователя |
| DATA\_CREATE | DateTime | Время создания профиля |
| USER\_INFO | Text | Описание профиля |

Таблица «users» содержит расширенную информацию о пользователях, включая их личные данные и настройки профиля. Здесь хранятся данные об уровне доступа, что позволяет системе разграничивать права пользователей и администраторов. Дополнительные столбцы, такие как аватар и описание профиля, создают основу для индивидуализации профиля, улучшая взаимодействие с пользователем. Связь с таблицей «authorized» позволяет объединить данные профиля с механизмом авторизации, что повышает удобство и безопасность.

Описание структуры таблицы «books» представлено в таблице 2.2.

Таблица 2.2 – Структура таблицы «books»

|  |  |  |
| --- | --- | --- |
| Название столбца | Тип, ограничение целостности | Описание столбца |
| ID | int, primary key | Идентификатор книги |
| BOOK\_NAME | Text | Название книги |
| BOOK\_SERIES | Text | Цикл книги |
| BOOK\_DESCRIPTION | Text | Описание книги |
| CHAPTERS | int | Количество глав |
| DATA\_RELEASE | DateTime | Дата выпуска |
| DATA\_ADD | DateTime | Дата добавления в бд |

Таблица «books» является основным хранилищем данных о каждой книге, объединяя информацию о названии, серии, статусе, описании и медиафайлах. Она также содержит информацию о количестве глав, дате выпуска и других атрибутах, что делает систему универсальным средством для управления литературными произведениями. Эта таблица является центральной частью функционала, отвечающего за добавление, редактирование и удаление книг, а также отображение их детальной информации пользователям.

Структура таблицы «authors» представлена в таблице 2.3.

Таблица 2.3 – Структура таблицы «authors»

|  |  |  |
| --- | --- | --- |
| Название столбца | Тип, ограничение целостности | Описание столбца |
| ID | Int, primary key | Идентификатор автора |
| AUTHOR\_NAME | varchar(25), unique | Имя автора |

Таблица «authors» выполняет важную роль в классификации литературных произведений, представляя уникальный список всех авторов, чьи книги хранятся в библиотеке. Эта таблица позволяет системе связывать каждую книгу с соответствующим автором, а также реализует функционал фильтрации по автору, что помогает пользователям находить книги определенного автора и углубляться в его творчество.

Описание структуры таблицы «genre» представлено в таблице 2.4.

Таблица 2.4 – Структура таблицы «genre»

|  |  |  |
| --- | --- | --- |
| Название столбца | Тип, ограничение целостности | Описание столбца |
| ID | Int, primary key | Идентификатор жанра |
| GENRE\_NAME | varchar(25), unique | Название жанра |

Таблица «genre» используется для организации книг по жанрам, создавая систему классификации для удобного поиска и сортировки. Жанры помогают структурировать библиотеку и предоставляют пользователям возможность быстро ориентироваться в ее содержимом. Эта таблица обеспечивает поддержку фильтров и категорий, что важно для улучшения пользовательского опыта и облегчения навигации.

Описание структуры таблицы «tag» представлено в таблице 2.5.

Таблица 2.5 – Структура таблицы «tag»

|  |  |  |
| --- | --- | --- |
| Название столбца | Тип, ограничение целостности | Описание столбца |
| ID | Int, primary key | Идентификатор тега |
| TAG\_NAME | varchar(25), unique | Название тега |

Таблица «tag» обеспечивает гибкость в организации библиотеки, добавляя к каждой книге ключевые слова или метки, которые описывают ее особенности. Теги предоставляют дополнительный способ для навигации, позволяя пользователям искать книги по специфическим темам, настроениям или характеристикам, не привязанным к жанрам или авторам. Это важный элемент для персонализированного поиска и рекомендации контента.

Описание структуры таблицы представлено в таблице 2.6.

Таблица 2.6 – Структура таблицы «book\_author»

|  |  |  |
| --- | --- | --- |
| Название столбца | Тип, ограничение целостности | Описание столбца |
| ID | int, primary key | Идентификатор записи таблицы |
| BOOKID | int, foreign key | Идентификатор книги |
| AUTHORID | int, foreign key | Идентификатор автора |

Таблица «book\_author» представляет собой связь между книгами и их авторами, позволяя системе поддерживать многие-к-одному и многие-ко-многим отношения. Это важно для работы с книгами, написанными несколькими авторами, а также для отображения полного списка работ конкретного автора.

Описание структуры таблицы представлено в таблице 2.7.

Таблица 2.7 – Структура таблицы «book\_genre»

|  |  |  |
| --- | --- | --- |
| Название столбца | Тип, ограничение целостности | Описание столбца |
| ID | int, primary key | Идентификатор записи таблицы |
| BOOKID | int, foreign key | Идентификатор книги |
| GENREID | int, foreign key | Идентификатор жанра |

Таблица «book\_genre» является связующей таблицей между книгами и их жанрами, создавая структуру для фильтрации книг по определенным категориям. Это важно для организации библиотеки и помогает пользователям быстрее находить книги по интересующим их жанрам.

Описание структуры таблицы представлено в таблице 2.8.

Таблица 2.8 – Структура таблицы «book\_tag»

|  |  |  |
| --- | --- | --- |
| Название столбца | Тип, ограничение целостности | Описание столбца |
| ID | int, primary key | Идентификатор записи таблицы |
| BOOKID | int, foreign key | Идентификатор книги |
| TAGID | int, foreign key | Идентификатор тега |

Таблица «book\_tag» связывает книги с тегами, что позволяет более подробно описывать каждое произведение. Это особенно полезно для тем, которые могут не попадать в стандартные жанровые категории, такие как темы книг или аудитория. Возможность добавления тегов улучшает фильтрацию и поиск, делая навигацию по библиотеке более интуитивной и удобной.

Описание структуры таблицы представлено в таблице 2.9.

Таблица 2.9 – Структура таблицы «feeadback»

|  |  |  |
| --- | --- | --- |
| Название столбца | Тип, ограничение целостности | Описание столбца |
| ID | Int, primary key | Идентификатор отзыва |
| ID\_USER | Int, foreign key | Идентификатор пользователя |
| ID\_BOOK | Int, foreign key | Идентификатор книги |
| FEEDBACK | text | Текст отзыва |
| RATING | float | Оценка |
| DATE | DateTime | Дата создания |

Таблица «feedback» хранит отзывы и оценки пользователей для каждой книги, поддерживая социальный функционал системы. Отзывы позволяют другим пользователям узнать мнение о книгах и принимать более обоснованные решения, а оценки влияют на общий рейтинг книги, что может быть полезно для рекомендаций. Это также стимулирует пользователей оставлять свои впечатления и участвовать в жизни сообщества.

Описание структуры таблицы представлено в таблице 2.10.

Таблица 2.10 – Структура таблицы «user\_bookmarks»

|  |  |  |
| --- | --- | --- |
| Название столбца | Тип, ограничение целостности | Описание столбца |
| ID | Int, primary key | Идентификатор отзыва |
| ID\_USER | Int, foreign key | Идентификатор пользователя |
| ID\_BOOK | Int, foreign key | Идентификатор книги |
| STATUS\_READING | varchar(25) | Статус чтения книги |
| DATA\_READING | DateTime | Дата последнего чтения |
| DATA\_ADD | DateTime | Дата добавления |
| MARKS | Varchar(1000) | Закладка, в книге |

Таблица «user\_bookmarks» содержит данные о закладках, созданных пользователями, и фиксирует статус прочтения книги. Это позволяет пользователям сохранять прогресс чтения и возвращаться к нужной главе. Закладки улучшают пользовательский опыт, предлагая персонализацию и напоминая пользователям, где они остановились, что делает чтение более удобным и приятным.

Таким образом при проектировании базы данных используется 11 таблиц, присутствует связь один-ко-многим и многие-ко-многим.

## **Вывод по разделу**

В результате данного проекта были разработаны основные компоненты, которые выполняют следующие функции:

Клиентское приложение: Клиентское приложение представляет собой веб-интерфейс на React Bootstrap, который обеспечивает пользовательскую интеракцию. Оно позволяет пользователям отправлять запросы на сервер, визуализировать ответы и взаимодействовать с функциональностью приложения. Клиентское приложение отображает данные, полученные от сервера, и обновляет интерфейс в соответствии с действиями пользователя.

Серверная часть: Серверная часть основана на фреймворке Express и обрабатывает запросы, поступающие от клиентского приложения. Она обеспечивает маршрутизацию запросов и взаимодействие с базой данных. Серверная часть выполняет следующие функции:

* принимает запросы от клиентского приложения через HTTPS протокол;
* маршрутизирует запросы к соответствующим обработчикам маршрутов;
* обрабатывает запросы, взаимодействуя с базой данных для создания, чтения, обновления и удаления данных;
* формирует ответы и отправляет их обратно клиентскому приложению.
* обеспечивает Web-Socket соединение.

База данных: База данных хранит информацию о пользователях, задачах и других сущностях, необходимых для работы приложения. Она обеспечивает сохранение и извлечение данных, которые используются серверной частью приложения. База данных выполняет следующие функции:

* хранит данные о пользователях, задачах и других сущностях;
* обеспечивает доступ к данным для чтения, записи, обновления и удаления;
* поддерживает целостность данных и обеспечивает безопасность доступа к данным.

Концептуально, основными компонентами проекта являются клиентское приложение, серверная часть и база данных. Клиентское приложение предоставляет пользовательский интерфейс и взаимодействует с сервером, отправляя запросы и получая ответы. Серверная часть обрабатывает запросы от клиента, взаимодействует с базой данных и возвращает результаты обратно клиентскому приложению. База данных хранит и обеспечивает доступ к данным, необходимым для функционирования приложения.

Таким образом, проект представляет собой систему, которая позволяет пользователям управлять задачами через веб-интерфейс. Клиентское приложение обеспечивает удобный интерфейс, а серверная часть и база данных обеспечивают хранение данных и выполнение операций над ними.