

МИНИСТЕРСТВО ЦИФРОВОГО РАЗВИТИЯ, СВЯЗИ И МАССОВЫХ  
КОММУНИКАЦИЙ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение  
высшего образования  
«Поволжский государственный университет телекоммуникаций и информатики»  
КОЛЛЕДЖ СВЯЗИ

Допустить к защите  
Зав. отделением \_\_\_\_\_ / Ситникова Л.Г. /  
Подпись      Расшифровка подписи  
« \_\_\_\_\_ » июня 2024 г.

Форма обучения \_\_\_\_\_ Очная

Специальность \_\_\_\_\_ 09.02.07 «Информационные системы и программирование»

Тема: \_\_\_\_\_ Интерактивный опыт в фитнесе: проектирование и создание  
\_\_\_\_\_ веб-сайта для фитнес-клуба

ДИПЛОМНАЯ РАБОТА

*КС ПГУТИ 09.02.07. 001*

Руководитель	<u>Преподаватель</u> Должность	_____	/ <u>Андреевская Н.В.</u> / Подпись      Расшифровка подписи	_____ <u>.2024</u> Дата
Н.контролер	<u>Преподаватель</u> Должность	_____	/ <u>Горобей В.В.</u> / Подпись      Расшифровка подписи	_____ <u>.2024</u> Дата
Рецензент	<u>Преподаватель</u> Должность	_____	/ <u>Шомас Е.А.</u> / Подпись      Расшифровка подписи	_____ <u>.2024</u> Дата
Разработал студент	<u>4 ИСПп-5</u> Группа	_____	/ <u>Угатьев Е.В.</u> / Подпись      Расшифровка подписи	_____ <u>.2024</u> Дата

Защищен(а) с оценкой \_\_\_\_\_

Самара  
2024 г

## Содержание

Введение.....	3
1. Теоретические основы построения веб-приложений.....	10
1.1 Понятие веб-приложения. Виды веб-приложений .....	10
1.2 Структура веб-приложения .....	12
1.3 Анализ потребностей библиотеки и ее пользователей .....	14
2. Анализ средств создания веб-приложения .....	16
2.1 Языки разметки и стили .....	16
2.2 Языки программирования .....	18
2.3 Фреймворки и библиотеки .....	20
2.4 Системы управления контентом (CMS).....	22
2.5 Графические редакторы и дизайн-инструменты.....	23
2.6 Базы данных.....	25
2.7 Хостинг-провайдеры и доменные регистраторы .....	26
3 Разработка веб-приложения .....	28
3.1 Описание предметной области и требований к веб-приложению .....	28
3.2 Выбор технологического стека для разработки сайта.....	30
3.3 Создание структуры сайта .....	32
3.4 Создание базы данных и реализация функционала хранения и управления данными.....	34
3.5 Разработка пользовательского интерфейса и его компонентов.....	43
3.6 Оптимизация и производительность.....	47
Заключение.....	50
Список использованных источников.....	52

**МИНИСТЕРСТВО ЦИФРОВОГО РАЗВИТИЯ, СВЯЗИ И МАССОВЫХ  
КОММУНИКАЦИЙ РОССИЙСКОЙ ФЕДЕРАЦИИ**

Федеральное государственное бюджетное образовательное учреждение  
высшего образования  
«Поволжский государственный университет телекоммуникаций и информатики»  
**КОЛЛЕДЖ СВЯЗИ**

Утверждаю  
Зав. отделением \_\_\_\_\_ / Ситникова Л.Г. /  
Подпись Расшифровка подписи  
«    » \_\_\_\_\_ 2024 г.

**ЗАДАНИЕ**  
**по подготовке выпускной квалификационной работы**

Студента(-ки) *Угатьева Евгения Владимировича*

1 Тема ВКР *SideeVerse: проектирование и создание веб-приложения для библиотеки*

Утверждена приказом по КС ПГУТИ от *19.01.2024 № 07-2*

2 Срок сдачи студентом законченной ВКР *12.06.24*

3 Исходные данные и постановка задачи

*1) Справочный центр*

4 Перечень подлежащих разработке в ВКР вопросов или краткое содержание ВКР. Сроки исполнения *12.06.2024*

*1) Изучение теоретических аспектов разработки веб-приложений*

*2) Анализ средств создания веб-приложений*

*3) Разработка требований к программному продукту*

*4) Разработка ПП*

*5) Разработка перспективных направлений создания проекта*

5 Перечень графического материала. Сроки исполнения *12.06.2024 г.*

*1) Презентация в программе Microsoft Office PowerPoint*

6 Сформировать общие и профессиональные компетенции:

*ОК 1. Понимать сущность и социальную значимость своей будущей профессии, проявлять к ней устойчивый интерес*

*ОК 2. Организовывать собственную деятельность, определять методы и способы выполнения профессиональных задач, оценивать их эффективность и качество*

*ОК 4. Осуществлять поиск и использование информации, необходимой для эффективного выполнения профессиональных задач, профессионального и личностного развития*

*ОК 5. Использовать информационно-коммуникационные технологии в профессиональной деятельности*

*ОК 8. Самостоятельно определять задачи профессионального и личностного развития, заниматься самообразованием, осознанно планировать повышение квалификации*

*ПК 2.1. Участвовать в разработке технического задания.*

*ПК 2.2. Программировать в соответствии с требованиями технического задания.*

*ПК 2.3. Применять методики тестирования разрабатываемых приложений.*

*ПК 2.6. Использовать критерии оценки качества и надежности функционирования информационной системы.*

7 Задание рассмотрено на заседании П(Ц)К «Информационных систем и технологий»

Протокол № \_\_\_\_ от « \_\_\_\_ » \_\_\_\_\_ 2024 г.

Председатель П(Ц)К	_____	/Шомас Е.А. /	_____
	Подпись	Расшифровка подписи	Дата

Дата выдачи задания « 15 » апреля 2024 г.

Руководитель	<u>Преподаватель</u>	_____	/Андреевская Н.В./	<u> . .2024</u>
	Должность	Подпись	Расшифровка подписи	Дата

Задание принял к исполнению студент(-ка)	<u>4 ИСПп- 05</u>	_____	/Угатьев Е.В./	<u> . .2024</u>
	Группа	Подпись	Расшифровка подписи	

**МИНИСТЕРСТВО ЦИФРОВОГО РАЗВИТИЯ, СВЯЗИ И МАССОВЫХ  
КОММУНИКАЦИЙ РОССИЙСКОЙ ФЕДЕРАЦИИ**

Федеральное государственное бюджетное образовательное учреждение  
высшего образования  
**«Поволжский государственный университет телекоммуникаций и информатики»**

## КОЛЛЕДЖ СВЯЗИ

## ОТЗЫВ РУКОВОДИТЕЛЯ

ВКР на тему «*Интерактивный опыт в фитнесе: проектирование и создание веб-сайта для фитнес-клуба*»

Студента (-ки) Угатьева Евгения Владимировича

Специальность 09.02.07 «Информационные системы и программирование»

Руководитель Андреевская Наталья Владимировна

This image shows a single page of white paper with horizontal black lines, resembling notebook paper. The lines are evenly spaced and run across the width of the page. There is no handwriting or other markings on the paper.

Blank lined area for text entry.

Руководитель ВКР

\_\_\_\_\_  
Дата

\_\_\_\_\_  
Подпись

Андреевская Н.В.  
ФИО

Рецензент Шомас Елена Александровна

[illegible]

## Рецензент

Подпись

Дата

Шомас Е..А.  
ФИО



## **Введение**

В современном мире информационные технологии и интернет играют важнейшую роль в различных аспектах жизни, включая литературу, образование и доступ к информации. Библиотеки, будучи хранилищами знаний и культуры, также активно интегрируются в цифровое пространство, предоставляя свои услуги через интернет. Создание веб-приложения для библиотеки открывает новые возможности для пользователей, позволяя им получать доступ к разнообразным ресурсам в любое время и из любого места. Это особенно актуально в условиях цифровой трансформации и удалённого обучения, когда пользователи предпочитают искать и получать информацию онлайн.

Цель данной дипломной работы заключается в разработке веб-приложения библиотеки, которое будет соответствовать современным требованиям и удовлетворять потребности пользователей. Основными задачами сайта являются обеспечение удобного доступа к литературе, предоставление функционала для поиска и чтения книг, а также организация обратной связи пользователей с разработчиками. Разработка такого сайта требует глубокого понимания теоретических основ построения веб-ресурсов, анализа существующих инструментов и технологий, а также учёта специфических требований библиотеки и её аудитории.

# **1. Теоретические основы построения веб-приложений**

## **1.1 Понятие веб-приложения. Виды веб-приложений**

Веб-приложение — это программное обеспечение, доступное через веб-браузер, которое позволяет пользователям выполнять различные задачи и взаимодействовать с данными через Интернет. Веб-приложения классифицируются на несколько типов, каждый из которых имеет свои особенности и применение:

**Статические веб-приложения:**

Статические веб-приложения представляют собой веб-сайты, содержащие фиксированный контент, который не изменяется в зависимости от пользовательского взаимодействия. Они обычно используются для представления информации и редко содержат интерактивные элементы. Примеры статических веб-приложений включают сайты-визитки, лендинги или онлайн-брошюры. Эти приложения просты в разработке и обслуживании, но имеют ограниченные возможности по взаимодействию с пользователями.

**Динамические веб-приложения:**

Динамические веб-приложения предоставляют пользователю интерактивные возможности и динамическое обновление контента без перезагрузки страницы. Они широко применяются в различных областях, таких как электронная коммерция, социальные сети, банковские системы и онлайн-сервисы. Основные компоненты динамических веб-приложений включают формы для ввода данных, обработку запросов и отображение динамического контента. Динамические веб-приложения требуют более сложной архитектуры и могут включать серверную логику для обработки запросов.

**Одностраничные приложения (SPA):**

Одностраничные приложения (SPA) — это современный вид веб-приложений, которые загружаются единой страницей в браузере и динамически

обновляют содержимое без перезагрузки страницы. SPA предлагают более плавный пользовательский опыт, так как минимизируют задержки между действиями пользователя. Они часто используются для создания интерактивных веб-приложений, таких как веб-приложения для обработки данных в реальном времени, аналитики данных или управления задачами. Технологии, используемые для разработки SPA, включают JavaScript-фреймворки, такие как Angular, React и Vue.js.

Прогрессивные веб-приложения (PWA):

Прогрессивные веб-приложения (PWA) комбинируют лучшие черты веб-приложений и нативных мобильных приложений. PWA обеспечивают офлайн-работу, мгновенную загрузку, пуш-уведомления и могут быть установлены на устройство как нативное приложение. Они создаются с использованием веб-технологий, таких как HTML, CSS и JavaScript, и обеспечивают высокий уровень производительности и пользовательского опыта.

Мобильные веб-приложения:

Мобильные веб-приложения разрабатываются специально для мобильных устройств, таких как смартфоны и планшеты. Они оптимизированы для маленьких экранов и сенсорного управления. Хотя мобильные веб-приложения могут быть частью любого из вышеперечисленных типов, их основной фокус — адаптация интерфейса и функционала для мобильных пользователей.

Каждый вид веб-приложений имеет свои преимущества и недостатки, и выбор типа приложения зависит от целей и требований конкретного проекта. Для разработки интернет-библиотеки было выбрано создание динамического веб-приложения с элементами одностраничного приложения (SPA) для обеспечения интерактивного и удобного пользовательского опыта.

## 1.2 Структура веб-приложения

Структура веб-приложения определяет организацию его компонентов и взаимодействие между ними. Основная модель для организации веб-приложений — это модель MVC (Model-View-Controller), которая разделяет приложение на три основных компонента:

Модель (Model):

- Отвечает за обработку данных и бизнес-логику приложения.
- Включает в себя структуры данных, классы и функции для работы с данными.
- Может взаимодействовать с базой данных, внешними API и другими ресурсами.

Представление (View):

- Отвечает за отображение данных пользователю.
- Может включать шаблоны для удобного создания динамического контента.

Контроллер (Controller):

- Обрабатывает запросы пользователя и взаимодействует как с моделью, так и с представлением.
- Осуществляет маршрутизацию запросов и управление потоком данных.
- Содержит логику обработки запросов, валидацию данных и прочие операции.

Это классическая архитектура, но в современных веб-приложениях часто используются и другие подходы, такие как:

MVP (Model-View-Presenter):

Отличается от MVC тем, что презентер напрямую взаимодействует с представлением, тогда как в MVC контроллер только обрабатывает запросы и обновляет модель.

MVVM (Model-View-ViewModel):

Отличается от MVC тем, что ViewModel взаимодействует с представлением через привязку данных, обеспечивая автоматическое обновление интерфейса.

Flux/Redux:

Отличается от MVC тем, что в MVC состояние может быть распределено по моделям, а в Flux/Redux состояние централизовано в одном хранилище.

Также различные фреймворки и библиотеки, такие как React, Angular, Vue.js и Django, предлагают свои подходы к организации структуры приложения. Структура веб-приложения может также включать дополнительные компоненты, такие как:

Маршрутизация: определяет, как запросы URL соответствуют различным обработчикам в приложении. В одностраничных приложениях используется клиентская маршрутизация для управления переходами между страницами без перезагрузки.

Аутентификация: проверка подлинности пользователя.

Авторизация: определение прав доступа пользователя к различным частям приложения.

Управление состоянием: управление данными, которые могут изменяться во времени, и их синхронизация между компонентами.

При разработке веб-приложения необходимо учитывать требования проекта, особенности бизнес-логики и потребности пользователей для определения оптимальной структуры приложения. Эффективная структура веб-приложения способствует улучшению производительности, поддерживаемости и масштабируемости приложения, что является ключевым аспектом успешной разработки.

### 1.3 Анализ потребностей библиотеки и ее пользователей

Перед началом разработки веб-приложения библиотеки важно провести детальный анализ потребностей как самой библиотеки, так и её пользователей. Это позволит создать ресурс, который будет функциональным, удобным и востребованным. Рассмотрим основные потребности библиотеки и её аудитории:

Потребности библиотеки:

1. Каталогизация ресурсов: организация и хранение информации о книгах, журналах, электронных изданиях и других материалах. Веб-приложение должно предоставлять возможность быстрого и удобного добавления новых позиций в каталог и редактирования существующих записей.

2. Управление пользователями: регистрация и учет пользователей, контроль доступа к ресурсам и услугам. Веб-приложение должно обеспечивать простую и безопасную регистрацию, авторизацию учетных записей пользователей, разделяя доступ к функциональным элементам для разных ролей пользователей.

3. Обеспечение доступа к цифровым ресурсам: предоставление пользователям доступа к электронным книгам, статьям и другим материалам. Включает организацию онлайн-чтения и скачивания ресурсов.

Потребности пользователей:

1. Поиск и фильтрация информации: удобный и быстрый доступ к каталогу библиотеки для поиска необходимых ресурсов. Функционал поиска должен поддерживать возможность поиска по названиям произведений и именам авторов. Для улучшения точности и релевантности результатов, система должна предусматривать фильтрацию по странам происхождения и жанрам литературы.

2. Просмотр статистики статуса прочтения книг: доступ к информации о завершенных, текущих и запланированных к прочтению книгах.

Это позволяет пользователям отслеживать свои достижения, актуальное состояние чтения и эффективно планировать дальнейшее использование библиотеки.

3. Профиль пользователя: наличие профиля пользователя с данными, такими как имя пользователя, адрес электронной почты, изображение профиля и иной информацией. Пользователь должен иметь возможность редактирования данных профиля для поддержания актуальности и персонализации работы с библиотекой.

4. Обратная связь: возможность отправлять отзывы и предложения разработчикам и администрации веб-приложения. Должен включать форму обратной связи, которая обеспечит оперативное и удобное взаимодействие пользователей с командой проекта.

Анализ потребностей библиотеки и её пользователей является важным этапом в процессе разработки веб-приложения, так как позволяет определить ключевые функции и требования к ресурсу, что в конечном итоге приведет к созданию удобного и эффективного веб-приложения.

## 2. Анализ средств создания веб-приложений

Создание веб-приложений — это многогранный процесс, который включает выбор и использование различных инструментов и технологий. Этот процесс начинается с планирования и проектирования структуры приложения, выбора языков разметки и программирования, и завершается внедрением приложения на сервер и его поддержкой. В данном разделе мы рассмотрим основные средства, используемые для создания веб-приложений, включая языки разметки и стили, языки программирования, фреймворки и библиотеки, системы управления контентом (CMS), графические редакторы и дизайн-инструменты, базы данных, а также хостинг-провайдеры и доменные регистраторы.

### 2.1 Языки разметки и стили

Языки разметки и стилизации являются основой для создания структуры и внешнего вида веб-приложений. Они определяют, как контент будет отображаться в браузере.

HTML (HyperText Markup Language):

Является стандартным языком разметки для создания веб-страниц. Он используется для описания структуры и содержимого веб-документов. HTML позволяет создавать заголовки, абзацы, ссылки, изображения, таблицы и другие элементы. Основные теги HTML включают `<html>`, `<head>`, `<body>`, `<div>`, `<p>` и другие. HTML5, последняя версия языка, добавила новые элементы и атрибуты, такие как `<header>`, `<footer>` и другие, которые улучшают семантику и структуру веб-документов.

CSS (Cascading Style Sheets):

Используется для стилизации HTML-документов. Он позволяет задавать внешний вид и оформление элементов, таких как цвет, шрифт, отступы, границы и расположение.



CSS отделяет структуру документа от его презентации, что позволяет улучшить удобство поддержки и обновления кода. CSS3, последняя версия, добавила множество новых возможностей, включая анимации, переходы, градиенты, гибкие сетки (Flexbox) и сетки CSS Grid.

SASS/SCSS (Syntactically Awesome Style Sheets):

SASS и его синтаксис SCSS являются препроцессорами CSS, которые добавляют такие возможности, как переменные, вложенные правила, миксины и функции. Они упрощают написание и поддержку CSS-кода, делая его более организованным и модульным. Препроцессоры компилируют свой код в обычный CSS, который затем используется в веб-документе.

JavaScript и библиотеки для анимации:

JavaScript также играет роль в стилизации через динамическое управление DOM (Document Object Model). Библиотеки, такие как jQuery, позволяют создавать сложные анимации и интерактивные элементы, улучшая пользовательский опыт.

Пример реализации в проекте

Для разработки веб-приложения библиотеки использовались следующие языки разметки и стилизации:

HTML5: Использован для создания структуры веб-страниц, включая теги `<header>`, `<footer>`, `<article>`, `<section>`, которые обеспечивают семантическую разметку и улучшенную структуру документов.

CSS3: Применен для стилизации элементов на веб-страницах. Flexbox и CSS Grid применялись для организации макетов и обеспечения гибкости интерфейсов.

SASS (SCSS): Выбран в качестве препроцессора CSS для упрощения и структурирования стилей. SASS позволил использовать переменные, вложенные правила и миксины, что облегчило поддержку и масштабирование CSS-кода.

JavaScript: Использован для добавления интерактивности и динамического поведения на веб-страницах.

## 2.2 Языки программирования

Языки программирования являются основным инструментом для разработки логики и функциональности веб-приложений. Они используются как на стороне клиента, так и на стороне сервера.

JavaScript:

Является основным языком программирования на стороне клиента. Он позволяет создавать интерактивные элементы, такие как всплывающие окна, валидация форм, динамическое обновление контента без перезагрузки страницы. Современные библиотеки и фреймворки, такие как React, Angular и Vue.js, расширяют возможности JavaScript и упрощают разработку сложных пользовательских интерфейсов.

PHP:

Hypertext Preprocessor — это популярный серверный язык программирования, используемый для создания динамических веб-страниц и управления серверной логикой. Он часто используется в сочетании с базами данных MySQL и MariaDB. PHP прост в изучении и использовании, что делает его популярным выбором для разработки веб-приложений. К популярным CMS, таким как WordPress, Joomla и Drupal, также относится PHP.

Python:

Высокоуровневый язык программирования, известный своей простотой и читабельностью. В веб-разработке Python часто используется в сочетании с фреймворками Django и Flask. Django обеспечивает высокую скорость разработки и включает в себя многие встроенные функции, такие как ORM (Object-Relational Mapping) и система администрирования. Flask является более легковесным фреймворком, подходящим для небольших проектов и микросервисов.

Ruby:

Язык программирования, известный своей простотой. Ruby on Rails — это фреймворк для веб-разработки, основанный на языке Ruby.

Rails обеспечивает высокую продуктивность разработки благодаря таким концепциям, как DRY (Don't Repeat Yourself) и Convention over Configuration.

Java:

Язык программирования общего назначения, часто используемый в корпоративных веб-приложениях. Фреймворки, такие как Spring и JavaServer Faces (JSF), упрощают разработку веб-приложений на Java. Java отличается высокой производительностью и масштабируемостью, что делает его подходящим для крупных проектов.

C# и ASP.NET:

C# — это язык программирования, разработанный компанией Microsoft. ASP.NET — это фреймворк для веб-разработки на платформе .NET, который использует C#. ASP.NET обеспечивает высокую производительность и интеграцию с другими продуктами Microsoft, такими как SQL Server и Azure.

Пример реализации в проекте

Для разработки веб-приложения библиотеки использовались следующие языки программирования:

JavaScript:

Применялся для создания интерактивных элементов и динамического обновления контента. Использование JavaScript позволило реализовать такие функции, как всплывающие окна, валидация форм и асинхронное обновление данных (AJAX).

PHP:

Использовался для разработки серверной логики веб-приложения. PHP обеспечил взаимодействие с базой данных MySQL и обработку пользовательских запросов.

PHP был выбран за его простоту и широкую распространенность, что ускорило разработку и интеграцию с другими компонентами системы.

## 2.3 Фреймворки и библиотеки

Фреймворки и библиотеки облегчают разработку веб-приложений, предоставляя готовые компоненты и инструменты для решения типичных задач. Они ускоряют процесс разработки и улучшают качество кода.

### Frontend-фреймворки и библиотеки

#### React:

Библиотека для создания пользовательских интерфейсов, разработанная Facebook. React использует компонентный подход и виртуальный DOM для высокой производительности. React позволяет разработчикам создавать сложные интерфейсы путем построения компонентов, которые могут быть использованы повторно и комбинированы.

#### Angular:

Фреймворк для создания одностраничных приложений (SPA), разработанный Google. Angular использует TypeScript и предоставляет множество встроенных инструментов и компонентов. Angular позволяет создавать масштабируемые приложения с хорошо структурированным кодом.

#### Vue.js:

Прогрессивный фреймворк для создания пользовательских интерфейсов. Vue.js сочетает в себе лучшие качества React и Angular, предлагая гибкость и простоту. Vue.js позволяет легко интегрироваться в проекты и создавать компоненты, которые могут быть использованы повторно.

### Backend-фреймворки

#### Django:

Фреймворк для веб-разработки на Python, известный своей скоростью разработки и множеством встроенных функций. Django включает ORM (Object-Relational Mapping), систему администрирования, аутентификацию и другие инструменты, что позволяет быстро создавать сложные веб-приложения.

#### Flask:

Фреймворк на Python, подходящий для небольших проектов и микросервисов. Flask предоставляет минимальный набор инструментов, позволяя разработчикам гибко добавлять нужные компоненты по мере необходимости.

Ruby on Rails:

Фреймворк на Ruby, известный своей простотой и элегантностью. Ruby on Rails обеспечивает высокую продуктивность разработки благодаря концепциям DRY (Don't Repeat Yourself) и Convention over Configuration, что упрощает создание и поддержку веб-приложений.

Spring:

Фреймворк на Java, подходящий для корпоративных приложений. Spring предлагает множество модулей для работы с базами данных, безопасности, интеграции и других задач, что делает его мощным инструментом для разработки сложных приложений.

ASP.NET Core:

Фреймворк на C#, обеспечивающий высокую производительность и интеграцию с экосистемой Microsoft. ASP.NET Core позволяет создавать масштабируемые и высокопроизводительные веб-приложения, поддерживая множество современных технологий и стандартов.

CSS-фреймворки и библиотеки

Bootstrap:

Популярный CSS-фреймворк, разработанный Twitter. Bootstrap предоставляет множество готовых компонентов и стилей, упрощая создание адаптивных веб-приложений. Bootstrap включает сеточную систему, кнопки, формы, навигацию и другие элементы интерфейса.

Tailwind CSS:

Утилитарный CSS-фреймворк, позволяющий создавать индивидуальные стили с использованием классов. Tailwind CSS предоставляет гибкость в стилизации, позволяя разработчикам быстро настраивать внешний вид компонентов без написания дополнительного CSS-кода.

Bulma:

Современный CSS-фреймворк, основанный на Flexbox, который обеспечивает простоту и гибкость при создании адаптивных интерфейсов. Bulma включает множество готовых стилей и компонентов, что ускоряет процесс разработки.

Пример реализации в проекте

Для разработки нашего веб-приложения библиотеки использовались следующие фреймворки и библиотеки:

Frontend:

Bootstrap: Применялся для стилизации интерфейса и создания адаптивного дизайна. Готовые компоненты Bootstrap упростили процесс верстки и стилизации.

Эти фреймворки и библиотеки значительно упростили процесс разработки и позволили создать высококачественное веб-приложение, удовлетворяющее потребности пользователей и библиотеки.

## **2.4 Системы управления контентом (CMS)**

CMS (Content Management Systems) — это платформы, которые позволяют создавать и управлять веб-сайтами без необходимости написания кода. CMS предоставляют удобные интерфейсы для добавления и редактирования контента, управления пользователями и настройки сайта.

WordPress:

Самая популярная CMS в мире, используемая для создания более 40% всех веб-сайтов. WordPress предоставляет широкий выбор тем и плагинов, которые позволяют легко настраивать внешний вид и функциональность сайта. Он подходит как для блогов и новостных сайтов, так и для интернет-магазинов и корпоративных сайтов.

Joomla:

Мощная и гибкая CMS, которая позволяет создавать сложные веб-сайты и порталы. Joomla предлагает множество расширений и шаблонов, что делает её подходящей для различных типов проектов.

Drupal:

Известна своей масштабируемостью и безопасностью. Она используется для создания крупных и сложных веб-сайтов, таких как государственные порталы, образовательные ресурсы и корпоративные сайты. Drupal предоставляет широкие возможности для настройки и расширения функциональности через модули.

Magento:

Специально разработанная CMS для создания интернет-магазинов. Magento предлагает мощные инструменты для управления товарами, заказами и клиентами, а также поддерживает интеграцию с различными платёжными системами и службами доставки.

Пример реализации в проекте

В нашем проекте CMS не использовалась, так как было принято решение создать кастомное веб-приложение для библиотеки с использованием PHP и MySQL для серверной части и управления данными. Это позволило более точно удовлетворить специфические требования проекта и обеспечить гибкость в разработке функционала.

## **2.5 Графические редакторы и дизайн-инструменты**

Графические редакторы и дизайн-инструменты используются для создания макетов, дизайна пользовательских интерфейсов и разработки графических элементов для веб-приложений.

Adobe Photoshop:

Мощный графический редактор, используемый для редактирования растровых изображений и создания графики для веб-приложений.

Photoshop предлагает широкий набор инструментов для обработки изображений, создания макетов и прототипирования интерфейсов.

Adobe Illustrator:

Инструмент для создания векторной графики. Он используется для разработки логотипов, иконок, иллюстраций и других графических элементов. Векторная графика имеет преимущество в том, что её можно масштабировать без потери качества.

Figma:

Облачный инструмент для дизайна и прототипирования, который позволяет командам работать совместно в реальном времени. Figma предлагает мощные инструменты для создания макетов, прототипов и анимаций, а также поддержку плагинов для расширения функциональности.

InVision:

Платформа для прототипирования и разработки пользовательских интерфейсов. InVision позволяет создавать интерактивные прототипы, проводить тестирование с пользователями и управлять дизайном проекта. Платформа также поддерживает интеграцию с другими инструментами, такими как Sketch и Photoshop.

Пример реализации в проекте

Для разработки веб-приложения библиотеки использовались следующие графические редакторы и дизайн-инструменты:

Adobe Photoshop:

Использовался для создания и редактирования растровых изображений, таких как обложки книг и другие графические элементы, необходимые для визуального оформления веб-приложения.

Adobe Illustrator:

Применялся для разработки векторных графических элементов, таких как логотипы, иконки и иллюстрации, что обеспечило их высокое качество при масштабировании.



## 2.6 Базы данных

Базы данных используются для хранения и управления информацией на веб-приложениях. Они позволяют организовать данные в структурированном виде и обеспечивают быстрый доступ к ним.

### MySQL:

Реляционная база данных, которая является одной из самых популярных в мире. MySQL используется в сочетании с языком программирования PHP для создания динамических веб-приложений. Она предлагает высокую производительность, надёжность и масштабируемость.

### PostgreSQL:

Мощная реляционная база данных с открытым исходным кодом, известная своей расширяемостью и поддержкой сложных запросов. PostgreSQL поддерживает широкий спектр функций, таких как индексы, транзакции, триггеры и функции на стороне сервера, что делает её подходящей для сложных и требовательных приложений.

### MongoDB:

Нереляционная (NoSQL) база данных, которая хранит данные в виде документов JSON-подобных объектов. MongoDB подходит для приложений, требующих высокой производительности и гибкости в управлении данными. Она используется для хранения больших объёмов данных и поддержки горизонтального масштабирования.

### Пример реализации в проекте

Для разработки веб-приложения библиотеки использовалась база данных MySQL:

MySQL была выбрана для хранения и управления данными веб-приложения. Она обеспечила высокую производительность, надёжность и простоту интеграции с серверной частью, написанной на PHP. MySQL использовалась для организации и хранения данных о книгах, пользователях и другой необходимой информации.

## 2.7 Хостинг-провайдеры и доменные регистраторы

Для того чтобы веб-приложение стало доступным пользователям через Интернет, необходимо разместить его на сервере хостинг-провайдера и зарегистрировать доменное имя. Хотя текущий проект находится только локально, установка на хостинг возможна при дальнейшем развитии проекта.

### Хостинг-провайдеры

#### Shared Hosting:

Общий хостинг — это наиболее доступный вариант, при котором несколько веб-сайтов размещаются на одном сервере. Это экономичный вариант для небольших сайтов с низкой нагрузкой, но он имеет ограниченные ресурсы и производительность.

#### VPS Hosting (Virtual Private Server):

Виртуальный частный сервер предоставляет больше ресурсов и возможностей по сравнению с общим хостингом. VPS хостинг разделяет физический сервер на несколько виртуальных машин, каждая из которых работает независимо. Это подходящий вариант для средних и крупных сайтов с умеренной нагрузкой.

#### Dedicated Hosting:

Выделенный сервер предоставляет все ресурсы физического сервера для одного веб-сайта. Это дорогой, но мощный вариант, который подходит для крупных проектов с высокой нагрузкой и особыми требованиями к безопасности и производительности.

#### Cloud Hosting:

Облачный хостинг использует ресурсы нескольких серверов для обеспечения высокой доступности и масштабируемости. Он предлагает гибкость в управлении ресурсами и оплату за использование. Это подходящий вариант для проектов с переменной нагрузкой и высокими требованиями к отказоустойчивости.

### Managed Hosting:

Управляемый хостинг предоставляет услуги по управлению сервером, обновлению программного обеспечения, резервному копированию и безопасности. Это позволяет сосредоточиться на разработке и управлении сайтом, оставив технические задачи хостинг-провайдеру.

### Доменные регистраторы

Доменное имя — это уникальный адрес сайта в Интернете. Доменные регистраторы предоставляют услуги по регистрации и управлению доменными именами.

### GoDaddy:

Один из крупнейших доменных регистраторов в мире, предлагающий широкий спектр услуг, включая регистрацию доменов, веб-хостинг и SSL-сертификаты.

### Namecheap:

Популярный регистратор доменов, известный своими конкурентоспособными ценами и качественным обслуживанием клиентов.

### Google Domains:

Услуга регистрации доменов от Google, предлагающая простую интеграцию с другими сервисами Google, такими как G Suite и Google Cloud.

### HostGator:

Регистратор доменов и веб-хостинг провайдер, предлагающий различные планы хостинга, включая общий хостинг, VPS и выделенные серверы.

### Пример реализации в проекте

На данный момент проект находится только локально, но в будущем, при дальнейшем развитии, возможна установка на хостинг и регистрация доменного имени. В зависимости от потребностей проекта, можно будет выбрать подходящий хостинг-провайдер и доменного регистратора для обеспечения доступности и надежной работы веб-приложения.

### **3 Разработка веб-приложения**

Разработка веб-приложения библиотеки представляет собой многослойный процесс, охватывающий несколько ключевых этапов, каждый из которых направлен на создание удобного и функционального ресурса.

#### **3.1 Описание предметной области и требований к веб-приложению**

Предметная область веб-приложения — это библиотека, предоставляющая доступ к широкому спектру книг и других печатных материалов в электронном формате. Интернет-библиотека призвана удовлетворять потребности пользователей, предоставляя удобный доступ к ресурсам, упрощая процесс поиска и чтения книг, а также обеспечивая взаимодействие между библиотекой и её пользователями.

Основные элементы предметной области:

- Книги и печатные материалы: основное содержимое библиотеки, включающее книги, журналы, статьи и другие материалы, доступные в электронном виде.
- Пользователи: читатели, которые могут просматривать, искать книги, оставлять отзывы и предлагать свои пожелания.
- Администраторы: сотрудники библиотеки, ответственные за управление контентом, пользователями и обработку запросов на бронирование.

Требования к веб-приложению

Для успешной реализации проекта интернет-библиотеки, веб-приложение должно соответствовать следующим функциональным и нефункциональным требованиям:

Функциональные требования:

Регистрация и авторизация пользователей:

- Возможность создания нового аккаунта.
- Авторизация существующих пользователей.

Управление профилем пользователя:

- Просмотр и редактирование личной информации.
- Просмотр статистики прочитанных книг.

Каталог книг:

- Отображение списка доступных книг.
- Фильтрация книг по различным критериям (жанр, страна).
- Просмотр общей информации о каждой книге.

Поиск книг:

- Поиск по ключевым словам.
- Фильтрация результатов поиска.

Обратная связь:

- Форма для отправки отзывов и предложений.
- Возможность задать вопросы администрации библиотеки.

Нефункциональные требования:

Удобство использования:

- Интуитивно понятный и удобный интерфейс.

Производительность:

- Быстрая загрузка страниц.
- Оптимизированный код для минимизации времени загрузки.

Безопасность:

- Защита личных данных пользователей.
- Защита от взломов и атак (например, SQL-инъекции, XSS-атаки).

Масштабируемость:

- Возможность добавления новых функций и расширения функционала без значительных изменений в архитектуре веб-приложения.

Надёжность:

- Минимизация времени простоя и сбоев в работе веб-приложения.
- Регулярное резервное копирование данных.

Таким образом, веб-приложение интернет-библиотеки должно обеспечивать полный спектр функций для удобного доступа к библиотечным ресурсам, предоставлять пользователям возможность эффективно взаимодействовать с библиотекой, а также соответствовать современным стандартам безопасности, производительности и удобства использования.

### **3.2 Выбор технологического стека для разработки сайта**

При разработке веб-приложения библиотеки важно выбрать технологический стек, который обеспечит стабильность, производительность и масштабируемость проекта.

Целевая платформа

Веб-приложение разрабатывается для корректной работы на ПК с возможностью адаптации на другие платформы в будущем. Это обеспечит масштабируемость проекта и его доступность для более широкой аудитории.

Языки программирования и разметки

Для фронтенда и бэкенда используются следующие языки:

- HTML: используется для создания структуры веб-страниц.
- CSS: отвечает за стилизацию и внешний вид элементов.
- JavaScript: добавляет интерактивность и динамическое поведение.
- PHP: серверный язык программирования, используемый для обработки запросов и управления данными.

Фреймворки и библиотеки

Для улучшения разработки и стилизации использованы следующие инструменты:

- Bootstrap: CSS-фреймворк, который упрощает создание адаптивного и современного дизайна.

### Система управления базами данных (DBMS)

Для хранения и управления данными выбрана база данных MySQL, используемая через phpMyAdmin в OpenServer. Это популярный выбор для веб-приложений, обеспечивающий надежность и производительность.

### Система управления контентом (CMS)

Использование CMS не планируется, что позволяет создать более уникальное решение, точно соответствующее требованиям проекта.

### Хостинг и домен

На текущем этапе разработки проект размещается на локальном сервере. В дальнейшем планируется использовать облачный хостинг или выделенный сервер, что обеспечит высокую доступность и масштабируемость ресурса.

### Инструменты для дизайна и прототипирования

Для разработки интерфейса и графических элементов используются:

- Adobe Photoshop: для создания и редактирования растровых изображений.
- Adobe Illustrator: для создания и редактирования векторных логотипов и элементов управления.
- Bootstrap: для стилизации интерфейса.

### Интеграция и API

На данном этапе интеграция с внешними сервисами и API не планируется, что упрощает архитектуру проекта.

### Безопасность

Для обеспечения безопасности данных пользователей используются следующие меры:

- Шифрование паролей: на данный момент используется алгоритм MD5. В дальнейшем планируется переход на более надежные алгоритмы шифрования, такие как bcrypt или Argon2, для повышения безопасности.

Таким образом, выбранный технологический стек обеспечивает выполнение всех необходимых функций для разработки веб-приложения библиотеки, а также закладывает основу для его дальнейшего масштабирования и улучшения.

### **3.3 Создание структуры сайта**

Создание структуры веб-приложения является важным этапом разработки, так как от нее зависит удобство навигации, доступность информации и общее впечатление пользователей.

Главная страница

Главная страница включает:

- Список книг: Книги отображаются с пагинацией по 10 книг на страницу.
- Фильтры: Фильтрация книг по странам и жанрам.
- Поле поиска: Поиск книг по названию или автору.

Детали книги

Страница с детальной информацией о книге предоставляет:

- Изображение книги: Обложка книги.
- Основные данные: Название, автор, жанр, страна, издательство, год выпуска, описание.
- Кнопки действий: Кнопка "Читать" открывает страницу для чтения книги, кнопка "Скачать" загружает PDF-файл книги.
- Статус прочтения: Выпадающий список позволяет авторизованному пользователю установить статус прочтения книги.
- Похожие книги: Отображение 4 случайных книг, схожих по автору, жанру, стране и другим критериям.



## Регистрация и авторизация

Процессы регистрации и авторизации реализованы на отдельных страницах:

- Страница регистрации: Форма для создания нового аккаунта.
- Страница авторизации: Форма для входа в существующий аккаунт.

## Профиль пользователя

Страница профиля пользователя предоставляет следующие возможности:

- Информация пользователя: Имя, фамилия, имя пользователя, электронная почта.
- Редактирование профиля: Перенаправление на страницу изменения данных профиля для обновления информации.
- Списки книг: Блоки со списками прочитанных, запланированных и читаемых книг.

## Обратная связь

Страница обратной связи включает:

- Форма обратной связи: Пользователи могут отправлять отзывы и предложения, которые сохраняются в базе данных.
- Методы связи: Дополнительные способы связи с разработчиком.

## Панель управления

Панель управления предоставляет следующие функции для администраторов:

- Формы ввода данных: Добавление новых записей в таблицы "Книг", "Авторов", "Жанров", "Стран", "Издательств".
- Таблицы данных: Управление и редактирование существующих данных.

Эта структура веб-приложения обеспечивает удобный и интуитивно понятный интерфейс как для пользователей, так и для администраторов, способствуя эффективному использованию библиотеки.

### 3.4 Создание базы данных и реализация функционала хранения и управления данными

База данных проекта интернет-библиотеки реализована на MySQL и содержит следующие таблицы: Authors, Books, Countries, Genres, Publishers, Users, Feedback, User\_Book\_Status. Эти таблицы обеспечивают хранение информации о книгах, авторах, пользователях и других элементах системы.

#### Структура базы данных

Таблицы в базе данных реализованы так, чтобы хранить максимально подробную информацию. Также структура базы данных соответствует третьей нормальной форме (3NF), которая требует, чтобы:

1. Таблица была в первой нормальной форме (1NF), что означает, что все значения атрибутов являются атомарными.
2. Таблица была во второй нормальной форме (2NF), что означает, что все неключевые атрибуты полностью зависят от первичного ключа.
3. В таблице не должно быть транзитивных зависимостей, что означает, что нет атрибутов, которые зависят от других неключевых атрибутов.

Для этого все атрибуты таблиц были тщательно нормализованы, чтобы избежать избыточности данных и обеспечить целостность данных.

Таблица 'Books' содержит следующие поля:

- book\_id (int, PRIMARY KEY): уникальный идентификатор книги.
- title (varchar(100)): название книги.
- image (varchar(100)): изображение книги
- text (text): текст описания книги.
- author\_id (int): идентификатор автора книги.
- country\_id (int): идентификатор страны создания книги.
- genre\_id (int): идентификатор жанра книги.
- publisher\_id (int): идентификатор издателя книги.
- year\_published (date): год публикации книги.

Таблица 'Authors' содержит следующие поля:

- author\_id (int, PRIMARY KEY): уникальный идентификатор автора.
- first\_name (varchar(50)): имя автора.
- last\_name (varchar(50)): фамилия автора.
- country\_id (int): идентификатор страны, связанного с автором.

Таблица 'Country' содержит следующие поля:

- country\_id (int, PRIMARY KEY): уникальный идентификатор страны.
- country\_name (varchar(50)): название страны.

Таблица 'Genres' содержит следующие поля:

- genre\_id (int, PRIMARY KEY): уникальный идентификатор жанра.
- genre\_name (varchar(50)): название жанра.

Таблица 'Publishers' содержит следующие поля:

- publisher\_id (int, PRIMARY KEY): уникальный идентификатор издателя.

- name (varchar(100)): название издательства.
- address (varchar(100)): адрес издательства.

Таблица 'Users' содержит следующие поля:

- user\_id (int, PRIMARY KEY): уникальный идентификатор пользователя.

- username (varchar(50)): имя аккаунта пользователя.
- password (varchar(255)): хешированный пароль пользователя.
- first\_name (varchar(50)): имя пользователя.
- last\_name (varchar(50)): фамилия пользователя.
- email (varchar(100)): электронная почта пользователя.
- Avatar (varchar(100)): изображение профиля пользователя.

Таблица 'Feedback' содержит следующие поля:

- feedback\_id (int, PRIMARY KEY): уникальный идентификатор отзыва.
- sender\_name (varchar(50)): имя аккаунта пользователя.
- sender\_email (varchar(100)): электронная почта пользователя.

- subject (varchar(50)): тема обращения.
- message\_text (text): текст обращения.
- Submission\_datetime (datetime): время обращения.

Таблица 'User\_book\_status' содержит следующие поля:

- id (int, PRIMARY KEY): уникальный идентификатор статуса прочтения.

- user\_id (int): служит для хранения идентификатора пользователя.
- book\_id (int): служит для хранения идентификатора книги.
- status (varchar(50)): служит для хранения статуса чтения книги.

### Реализация функционала

Для управления данными и обеспечения взаимодействия с пользователями были реализованы следующие функции:

#### Добавление записей на странице администратора

На странице администрирования проекта расположено несколько форм для удобного обновления данных в таблицах, без необходимости открывать базу данных напрямую. Это позволяет администраторам эффективно управлять контентом библиотеки и оперативно обновлять информацию.

#### Добавление авторов

Форма ввода данных для добавления авторов на странице администратора. Обработчик формы, который вставляет данные в таблицу Authors..

```
php
if ($_SERVER['REQUEST_METHOD'] === 'POST') {
    $first_name = mysqli_real_escape_string($connection,
    $_POST['first_name']);
    $last_name = mysqli_real_escape_string($connection,
    $_POST['last_name']);
    $query = "INSERT INTO authors (first_name, last_name) VALUES
    ('$first_name', '$last_name')";
    $result = mysqli_query($connection, $query);
    if ($result) {
        header('Location: ../administration.php?conf=Успешно добавлено');
        exit();
    } else {
        header("Location: ../administration.php?error=Ошибка: " .
        $connection->error);
        exit();
    }
}
```

```

    }
} else {
    header('Location: ../administration.php');
    exit();
}

```

## Добавление книг

Форма ввода данных для добавления книг на странице администратора.

Обработчик формы, который вставляет данные в таблицу Books.

```

php
if ($_SERVER['REQUEST_METHOD'] === 'POST') {
    $title = mysqli_real_escape_string($connection, $_POST['title']);
    $text = mysqli_real_escape_string($connection, $_POST['text']);
    $author_id = intval($_POST['author_id']);
    $genre_id = intval($_POST['genre_id']);
    $country_id = intval($_POST['country_id']);
    $publication_date = mysqli_real_escape_string($connection,
$_POST['publication_date']);
    $publisher_id = intval($_POST['publisher_id']);
    if (!empty($_FILES['image']['name'])) {
        $image = $_FILES['image']['name'];
        $image_tmp_name = $_FILES['image']['tmp_name'];
        $image_folder = '../assets/images/';
        if (move_uploaded_file($image_tmp_name, $image_folder . $image)) {
            $image = mysqli_real_escape_string($connection, $image);
        } else {
            echo '<div class="alert alert-danger text-center">Ошибка
загрузки файла.</div>';
            $image = '';
        }
    } else {
        $image = ''; // Нет нового файла, используем пустое значение
    }
    $query = "INSERT INTO books (title, text, image, author_id,
genre_id, country_id, publication_date, publisher_id) VALUES
('$title', '$text', '$image', $author_id, $genre_id, $country_id,
'$publication_date', $publisher_id)";
    $result = mysqli_query($connection, $query);
    if ($result) {
        header('Location: ../administration.php?conf=Успешно добавлено');
        exit();
    } else {
        header("Location: ../administration.php?error=Ошибка: " .
$connection->error);
        exit();
    }
} else {
    header('Location: ../administration.php');
    exit();
}

```

## Добавление новых жанров

Форма ввода данных для добавления новых жанров на странице администратора. Обработчик формы, который вставляет данные в таблицу Genres.

```
php
if ($_SERVER['REQUEST_METHOD'] === 'POST') {
    $genre_name = $_POST['genre_name'];
    $query = "INSERT INTO genres (genre_name) VALUES ('$genre_name')";
    $result = mysqli_query($connection, $query);
    if ($result) {
        header('Location: ../administration.php?conf=Успешно добавлено');
        exit();
    } else {
        header("Location: ../administration.php?error=Ошибка: " .
$connection->error);
        exit();
    }
} else {
    header('Location: ../administration.php');
    exit();
}
```

## Добавление стран

Форма ввода данных для добавления стран на странице администратора. Обработчик формы, который вставляет данные в таблицу Country.

```
php
if ($_SERVER['REQUEST_METHOD'] === 'POST') {
    $country_name = mysqli_real_escape_string($connection,
$_POST['country_name']);
    $query = "INSERT INTO country (country_name) VALUES
('$country_name')";
    $result = mysqli_query($connection, $query);
    if ($result) {
        header('Location: ../administration.php?conf=Успешно добавлено');
        exit();
    } else {
        header("Location: ../administration.php?error=Ошибка: " .
$connection->error);
        exit();
    }
} else {
    header('Location: ../administration.php');
    exit();
}
```

## Добавление издателей

Форма ввода данных для добавления издателей на странице администратора. Обработчик формы, который вставляет данные в таблицу Publishers.

```
php
if ($_SERVER['REQUEST_METHOD'] === 'POST') {
    $name = $_POST['name'];
    $address = $_POST['address'];
    $query = "INSERT INTO publishers (name, address) VALUES ('$name',
' $address')";
    $result = mysqli_query($connection, $query);
    if ($result) {
        header('Location: ../administration.php?conf=Успешно добавлено');
        exit();
    } else {
        header("Location: ../administration.php?error=Ошибка: " .
$connection->error);
        exit();
    }
} else {
    header('Location: ../administration.php');
    exit();
}
```

## Поиск и фильтрация книг

Функционал поиска и фильтрации книг позволяет пользователям легко находить и сортировать книги по различным критериям. Это важно для улучшения пользовательского опыта и обеспечения быстрого доступа к нужной информации.

### Поиск книг

Поиск книг реализован через форму на главной странице. Пользователи могут вводить ключевые слова, и система будет искать соответствия по названиям книг и именам авторов.

```
php
$sort = isset($_GET['sort']) ? $_GET['sort'] : '';
$genre_id = isset($_GET['genre_id']) ?
mysqli_real_escape_string($connection, $_GET['genre_id']) : '';
$country_id = isset($_GET['country_id']) ?
mysqli_real_escape_string($connection, $_GET['country_id']) : '';
$searchTerm = isset($_GET['search']) ?
mysqli_real_escape_string($connection, $_GET['search']) : '';
$query = "SELECT * FROM books";
$countQuery = "SELECT COUNT(*) AS total FROM books";
$whereClause = [];
if (!empty($sort) && $sort === 'genre' && !empty($genre_id)) {
    $whereClause[] = "genre_id = $genre_id";
```

```

}
if (!empty($sort) && $sort === 'country' && !empty($country_id)) {
    $whereClause[] = "country_id = $country_id";
}
if (!empty($searchTerm)) {
    $whereClause[] = "title LIKE '%$searchTerm%' OR author_id IN (SELECT
author_id FROM authors WHERE CONCAT(first_name, ' ', last_name) LIKE
'%$searchTerm%')";
}
if (!empty($whereClause)) {
    query .= " WHERE " . implode(" AND ", $whereClause);
    $countQuery .= " WHERE " . implode(" AND ", $whereClause);
}
$query .= " LIMIT $offset, $itemsPerPage";
$result = mysqli_query($connection, $query);
if (!$result) {
    die("Ошибка запроса: " . mysqli_error($connection));
}

```

### Фильтрация книг

Фильтрация книг позволяет пользователям отсортировать книги по жанру и стране. Это особенно полезно для пользователей, которые ищут книги определенного типа или из определенной области.

```

php
$sort = isset($_GET['sort']) ? $_GET['sort'] : '';
$genre_id = isset($_GET['genre_id']) ?
mysqli_real_escape_string($connection, $_GET['genre_id']) : '';
$country_id = isset($_GET['country_id']) ?
mysqli_real_escape_string($connection, $_GET['country_id']) : '';
$searchTerm = isset($_GET['search']) ?
mysqli_real_escape_string($connection, $_GET['search']) : '';
$query = "SELECT * FROM books";
$countQuery = "SELECT COUNT(*) AS total FROM books";
$whereClause = [];
if (!empty($sort) && $sort === 'genre' && !empty($genre_id)) {
    whereClause[] = "genre_id = $genre_id";
}
if (!empty($sort) && $sort === 'country' && !empty($country_id)) {
    whereClause[] = "country_id = $country_id";
}
if (!empty($searchTerm)) {
    whereClause[] = "title LIKE '%$searchTerm%' OR author_id IN (SELECT
author_id FROM authors WHERE CONCAT(first_name, ' ', last_name) LIKE
'%$searchTerm%')";
}
if (!empty($whereClause)) {
    $query .= " WHERE " . implode(" AND ", $whereClause);
    $countQuery .= " WHERE " . implode(" AND ", $whereClause);
}
$query .= " LIMIT $offset, $itemsPerPage";
$result = mysqli_query($connection, $query);
if (!$result) {
    die("Ошибка запроса: " . mysqli_error($connection));
}

```



## Регистрация и авторизация пользователей

Функционал регистрации и авторизации пользователей обеспечивает доступ к персонализированным функциям сайта. Это включает возможность создания учетной записи, входа в систему и управления профилем.

### Регистрация пользователей

Форма регистрации включает поля для имени пользователя, пароля, электронной почты и других данных. После успешной регистрации пользовательские данные сохраняются в базе данных.

```
php
if ($_SERVER["REQUEST_METHOD"] == "POST" && isset($_POST['register']))
{
    $username = $_POST['username'];
    $password = md5($_POST['password']);
    $first_name = $_POST['first_name'];
    $last_name = $_POST['last_name'];
    $email = $_POST['email'];

    $sql = "INSERT INTO Users (username, password, first_name,
last_name, email) VALUES ('$username', '$password', '$first_name',
'$last_name', '$email')";
    if (mysqli_query($conn, $sql)) {
        echo "Регистрация успешна.";
    } else {
        echo "Ошибка: " . $sql . "<br>" . mysqli_error($conn);
    }
}
```

### Авторизация пользователей

Форма авторизации включает поля для имени пользователя и пароля. После успешного входа пользователь перенаправляется на страницу профиля.

```
php
if ($_SERVER["REQUEST_METHOD"] == "POST") {
    $first_name = mysqli_real_escape_string($connection,
$_POST["first_name"]);
    $last_name = mysqli_real_escape_string($connection,
$_POST["last_name"]);
    $username = mysqli_real_escape_string($connection,
$_POST["username"]);
    $email = mysqli_real_escape_string($connection, $_POST["email"]);
    $password = mysqli_real_escape_string($connection,
$_POST["password"]);
    $hashed_password = md5($password);
    $check_username = "SELECT * FROM users WHERE username='$username'";
    $result = $connection->query($check_username);
    if ($result->num_rows > 0) {
        header("Location: ../registration.php?error=Никнейм занят");
        exit();
    } else {
```

```

        $sql = "INSERT INTO users (first_name, last_name, username, email,
password) VALUES ('$first_name', '$last_name', '$username', '$email',
'$hashed_password')";
        if ($connection->query($sql) === TRUE) {
            header("Location: ../login.php");
            exit();
        } else {
            header("Location: ../registration.php?error=Ошибка при
регистрации: " . $connection->error);
            exit();
        }
    }
}

```

## Изменение данных профиля

Пользователь может изменять данные профиля на странице профиля.

Это включает обновление имени, фамилии, электронной почты и пароля.

```

php
if ($_SERVER['REQUEST_METHOD'] === 'POST') {
    $first_name = mysqli_real_escape_string($connection,
$_POST['first_name']);
    $last_name = mysqli_real_escape_string($connection,
$_POST['last_name']);
    $username = mysqli_real_escape_string($connection,
$_POST['username']);
    $email = mysqli_real_escape_string($connection, $_POST['email']);
    $password = !empty($_POST['password']) ? md5($_POST['password']) :
null;
    if (!empty($_FILES['avatar']['name'])) {
        $avatar = $_FILES['avatar']['name'];
        $avatar_tmp_name = $_FILES['avatar']['tmp_name'];
        $avatar_folder = 'uploads/avatars/';
        if (move_uploaded_file($avatar_tmp_name, $avatar_folder .
$avatar)) {
            $avatar = mysqli_real_escape_string($connection, $avatar);
        } else {
            echo '<div class="alert alert-danger text-center">Ошибка
загрузки файла.</div>';
            $avatar = $_SESSION['avatar'];
        }
    } else {
        $avatar = $_SESSION['avatar'];
    }
    $update_query = "UPDATE users SET first_name = '$first_name',
last_name = '$last_name', username = '$username', email = '$email',
avatar = '$avatar'";
    if ($password) {
        $update_query .= ", password = '$password'";
    }
    $update_query .= " WHERE id_user = " . $_SESSION['user_id'];
    $result = mysqli_query($connection, $update_query);
    if ($result) {
        $_SESSION['first_name'] = $first_name;
        $_SESSION['last_name'] = $last_name;
        $_SESSION['username'] = $username;
    }
}

```

```

$_SESSION['email'] = $email;
$_SESSION['avatar'] = $avatar;
echo '<div class="alert alert-success text-center">Данные успешно
обновлены.</div>';
} else {
    echo '<div class="alert alert-danger text-center">Ошибка
обновления данных: ' . mysqli_error($connection) . '</div>';
}
}

```

Эти компоненты обеспечивают необходимый функционал для хранения и управления данными в интернет-библиотеке, предоставляя удобные инструменты для взаимодействия пользователей с ресурсом.

### 3.5 Разработка пользовательского интерфейса и его компонентов

Пользовательский интерфейс (UI) интернет-библиотеки разработан с учетом потребностей пользователей всех возрастов. Основное внимание уделено интуитивно понятному и простому интерфейсу, чтобы обеспечить легкость использования и избежать перегруженности элементов. В данном разделе рассмотрены детали разработки интерфейса для основных страниц веб-приложения и используемые компоненты.

#### Целевая аудитория

Веб-приложение интернет-библиотеки предназначено для людей всех возрастов. Основные требования к интерфейсу заключаются в его интуитивной понятности и простоте, чтобы не перегружать восприятие пользователей. Это особенно важно для обеспечения удобства использования веб-приложения как детьми, так и пожилыми людьми.

#### Цветовая схема и брендинг

Веб-приложение использует бело-голубую цветовую схему, так как эти цвета просты для восприятия и создают приятное визуальное впечатление. Брендинг включает название и логотип библиотеки, которые размещены в шапке (header) и подвале (footer) веб-приложения для узнаваемости и консистентности.

## Навигация по веб-приложению

Навигация в веб-приложении организована таким образом, чтобы пользователи могли легко и быстро находить нужные разделы и функции. Основные элементы навигации включают:

- Главная страница: отображает список всех доступных книг с возможностью поиска и фильтрации.
- Авторизация и регистрация: отдельные страницы для входа в систему и создания нового аккаунта.
- Профиль пользователя: страница с информацией о пользователе и его статистикой чтения книг.
- Панель управления: страница для администраторов с функциями управления данными в базе данных.
- Обратная связь: форма для отправки отзывов, предложений и критики, а также контактные данные.
- О проекте: страница с обобщенной информацией о веб-приложении и его разработке.

## Адаптивность интерфейса

На текущем этапе развития проекта адаптивность интерфейса отсутствует, однако в дальнейшем планируется добавить поддержку различных устройств для улучшения пользовательского опыта.

## Основные страницы и их компоненты

### Главная страница

Главная страница является основным входом для пользователей. Она включает следующие компоненты:

- Список книг: представлен в виде карточек с пагинацией по 10 книг на страницу.
- Форма поиска: поле для ввода ключевых слов.
- Фильтры: списки стран и жанров, по которым осуществляется фильтрация получаемых ресурсов.

Главная страница предоставляет пользователю возможность ознакомиться с основным контентом веб-приложения — доступными книгами. Навигация и функциональные элементы, такие как поиск и фильтры, упрощают процесс нахождения нужных книг.

#### Авторизация и регистрация

Эти страницы содержат формы для ввода данных:

- Страница авторизации: форма, содержащая поля для ввода имени пользователя и пароля.
- Страница регистрации: форма, содержащая поля для ввода имени пользователя, пароля, электронной почты и других данных.

Страницы авторизации и регистрации обеспечивают безопасный доступ к персонализированным функциям веб-приложения и позволяют новым пользователям создавать учетные записи.

#### Профиль пользователя

Страница профиля пользователя включает:

- Информация о пользователе: отображает изображение профиля, имя, фамилию, имя пользователя и электронную почту.
- Функция редактирования: позволяет пользователю изменить данные профиля.
- Списки книг: блоки со списками прочитанных, запланированных и читаемых книг.

Страница профиля предоставляет пользователям возможность управлять своими данными и отслеживать прогресс чтения книг. Возможность редактирования данных профиля позволяет пользователям поддерживать актуальность своей информации.

#### Панель управления

Панель управления предназначена для управления данными в базе данных администраторами:

- Формы для добавления данных: формы для добавления новых авторов, книг, стран, жанров и издателей.

- Таблицы данных: отображение существующих записей.

Панель управления предоставляет администраторам удобные инструменты для управления содержимым веб-приложения, позволяя добавлять, редактировать и удалять записи в базе данных без необходимости прямого взаимодействия с SQL.

#### Обратная связь

Страница обратной связи включает:

- Форма обратной связи: поля для ввода имени пользователя, электронной почты и текста сообщения.
- Контактные данные: информация о способах связи с администрацией веб-приложения.

Страница обратной связи предоставляет пользователям возможность оставить свои отзывы и предложения, а также получить необходимую информацию для связи с командой проекта.

#### О проекте

Страница "О проекте" предоставляет информацию о веб-приложении:

- Описание проекта: обобщенные данные о целях, задачах и функциях веб-приложения.
- Информация о разработке: данные о разработчиках и процессе создания веб-приложения.

Страница "О проекте" информирует пользователей о целях создания веб-приложения и дает представление о команде разработчиков и этапах разработки.

#### Использование компонентов Bootstrap

Веб-приложение активно использует компоненты фреймворка Bootstrap для создания адаптивного и современного интерфейса. Среди основных компонентов Bootstrap, используемых в веб-приложении, можно выделить:

- Navbar: для создания шапки веб-приложения с навигационными ссылками.

- Forms: для авторизации, регистрации и обратной связи.
- Buttons: для различных действий, таких как отправка форм или переход между страницами.
- Tables: для отображения данных в панели управления.

Эти элементы обеспечивают удобный и современный пользовательский интерфейс, соответствующий специфическим потребностям интернет-библиотеки.

### **3.6 Оптимизация и производительность**

Оптимизация и производительность веб-приложения играют ключевую роль в обеспечении быстрого и стабильного доступа к ресурсам для пользователей. В данном разделе рассмотрены методы и практики, применяемые для улучшения производительности сайта, включая загрузку страниц, оптимизацию баз данных, загрузку ресурсов, тестирование производительности, использование функций Bootstrap, оптимизацию изображений и меры безопасности.

#### **Загрузка страниц**

Для ускорения загрузки страниц применяются следующие методы:

- Сжатие CSS и JavaScript: используются минимизированные версии файлов CSS и JavaScript для уменьшения их размера и ускорения загрузки.

#### **Кэширование**

На сайте реализовано кэширование на стороне клиента:

- Кэширование на стороне клиента: использованы HTTP-заголовки для управления кэшированием браузера, что позволяет хранить статические ресурсы (CSS, JavaScript, изображения) в кэше браузера и уменьшает количество запросов к серверу.

## Оптимизация баз данных

Для оптимизации работы базы данных применяются следующие методы:

- Индексация: создание индексов для часто используемых полей, что ускоряет выполнение запросов.
- Нормализация: база данных нормализована до третьей нормальной формы (3NF), что минимизирует избыточность данных и повышает эффективность запросов.
- Оптимизация запросов: использование подготовленных выражений для повышения производительности и безопасности.

## Загрузка ресурсов

Загрузка ресурсов осуществляется с учетом следующих практик:

- Использование атрибутов `async` и `defer`: для скриптов, не критичных для начальной загрузки страницы, используются атрибуты `async` или `defer`, что позволяет загружать их параллельно с рендерингом страницы.

## Тестирование производительности

Для тестирования производительности сайта использовались следующие инструменты:

- Google PageSpeed Insights: инструмент для анализа производительности страниц и получения рекомендаций по улучшению.
- GTmetrix: сервис для тестирования скорости загрузки страниц и анализа производительности.

Результаты тестирования показали, что сайт загружается достаточно быстро и соответствует основным рекомендациям по оптимизации.

## Использование компонентов Bootstrap

На сайте активно используются компоненты фреймворка Bootstrap, что позволяет улучшить производительность за счет:

- Минимизированных версий файлов: использование сжатых (minified) версий файлов CSS и JavaScript Bootstrap.



- Сетка Bootstrap: применение адаптивной сетки Bootstrap для организации макета страниц, что обеспечивает консистентность и оптимизацию кода (GitHub).

#### Оптимизация изображений

Для оптимизации изображений применяются следующие техники:

- Сжатие изображений: перед загрузкой на сервер изображения проходят предварительное сжатие. Основным форматом является PNG.

#### Безопасность

Для обеспечения безопасности данных и предотвращения атак реализованы следующие меры:

- Использование подготовленных выражений для SQL-запросов, что защищает от SQL-инъекций.

#### Мониторинг и логирование

Для отслеживания производительности и выявления проблем используются системы мониторинга и логирования:

- Журналы ошибок: ведение логов ошибок приложения для быстрого выявления и устранения проблем.

Эти методы и практики обеспечивают высокую производительность и надежность работы сайта интернет-библиотеки, обеспечивая пользователям быстрый и стабильный доступ к ресурсам.

## Заключение

В ходе выполнения данной дипломной работы была поставлена и успешно решена задача разработки веб-сайта "SideeVerse: Knowledge Hub" — многофункциональной интернет-библиотеки, предоставляющей доступ к обширной коллекции литературных ресурсов. Проект, начавшийся с идеи улучшения доступа к образовательным материалам для студентов, эволюционировал в общедоступную платформу, ориентированную на широкий круг пользователей.

Основной целью проекта было создание удобного, интуитивно понятного и функционального веб-ресурса, обеспечивающего пользователей возможностью быстрого поиска, бронирования и доступа к различным литературным произведениям. В ходе работы над проектом были проведены следующие ключевые этапы:

Анализ потребностей пользователей и библиотеки — исследование требований, предъявляемых к современным библиотечным веб-приложениям, и определение функциональных задач, которые необходимо реализовать.

Выбор технологического стека — определение оптимальных инструментов и технологий для разработки веб-приложения.

Проектирование и разработка структуры сайта — создание логической схемы, разработка основных разделов и страниц, обеспечение удобной навигации и интерфейса.

Разработка пользовательского интерфейса и функционала — реализация интерфейса для поиска и фильтрации информации, механизмы для работы с данными.

Оптимизация производительности — улучшение производительности сайта, уменьшение времени загрузки страниц и обеспечение стабильной работы под нагрузкой.

Проект "SideeVerse: Knowledge Hub" доказал свою актуальность и значимость, предоставив пользователям удобный инструмент для доступа к

литературным ресурсам в любое время и из любого места. Это особенно важно в условиях цифровой трансформации и удаленного доступа к информации. Разработанный сайт соответствует современным стандартам и требованиям, обеспечивая высокий уровень функциональности и удобства использования.

В перспективе планируется дальнейшее развитие проекта, включая расширение коллекции книг, внедрение новых возможностей для пользователей и повышение уровня безопасности и надежности сайта. Мы верим, что "SideeVerse: Knowledge Hub" станет полезным инструментом для всех, кто стремится к саморазвитию и хочет иметь доступ к качественным литературным ресурсам.

## Список использованных источников

1	Книга с одним автором	Лоусон, Брюс. Изучаем HTML5. Библиотека специалиста [] / Брюс Лоусон – Санкт-Петербург: Питер, 2012. – 304 с.
2	Книга с одним автором	Дакетт, Джон. HTML и CSS. Разработка и дизайн веб-сайтов [] / Джон Дакетт – Москва: Эксмо, 2017. – 480 с.
3	Книга с одним автором	Веру, Лия. Секреты CSS. Идеальные решения ежедневных задач [] / Лия Веру – Санкт-Петербург: Питер, 2017. – 336 с.
4	Книга с одним автором	Никсон, Робин. Создаем динамические веб-сайты с помощью PHP, MySQL, JavaScript, CSS и HTML5 [] / Робин Никсон – СПб: Питер, 2023. – 832 с.
5	Книга с одним автором	Зандстра, Мэтт. PHP. Объекты, шаблоны и методики программирования [] / Мэтт Зандстра – М.: Диалектика-Вильямс, 2021. – 864 с.
6	Книга с одним автором	Морето, Сильвио. Bootstrap в примерах [] / Сильвио Морето, ДМК Пресс – М.: ДМК Пресс, 2017. – 314 с.
7	Книга с тремя авторами	Шварц, Б. MySQL по максимуму [] / Б. Шварц, В. Ткаченко, П. Зайцев – СПб.: Питер, 2023. – 432 с.