

Министерство образования Иркутской области

Государственное бюджетное профессиональное образовательное учреждение

Иркутской области

«Иркутский авиационный техникум»

(ГБПОУИО «ИАТ»)

ДП.09.02.07-3.25.212.02. ПЗ

УТВЕРЖДАЮ

Зам. директора по УР, к.т.н.

_____ Е.А. Коробкова

ВЕБ-ПРИЛОЖЕНИЕ «ПИСАТЕЛЬСКИЙ КЛУБ»

Нормконтролер:

(подпись, дата)

(Е.С. Кудрявцева)

Консультант по

экономической части

(подпись, дата)

(А.П. Юргина)

Руководитель:

(подпись, дата)

(А.С. Александрова)

Студент:

(подпись, дата)

(В.А. Антонинова)

Иркутск 2025

СОДЕРЖАНИЕ

ВВЕДЕНИЕ	3
1 Предпроектное исследование	5
1.1 Описание предметной области	5
1.2 Обзор инструментов, используемых для разработки веб-приложения.....	6
2 Техническое задание	9
3 Проектирование веб-приложения.....	10
3.1 Архитектура веб-приложения.....	10
3.2 Функциональное и структурное проектирование веб-приложения.....	11
3.3 Проектирование базы данных веб-приложения.....	14
3.4 Проектирование пользовательского интерфейса веб-приложения.....	17
4 Реализация веб-приложения.....	31
5 Тестирование веб-приложения	38
6 Стоимость разработки веб-приложения	43
7 Руководство пользователя веб-приложения.....	49
ЗАКЛЮЧЕНИЕ	55
СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ.....	56
Приложение А Техническое задание	58
Приложение Б Листинг главной страницы.....	65

					ДП.09.02.07-3.25.212.02. ПЗ					
Изм.	Лист	№ докум.	Подпись	Дата	ВЕБ-ПРИЛОЖЕНИЕ «ПИСАТЕЛЬСКИЙ КЛУБ» пояснительная записка			Лит.	Лист	Листов
Разраб.		Антонинова В.А.								
Провер.		Александрова А.С.							2	68
Рецент.								ГБПОУИО «ИАТ» ВЕБ-21-2		
Н. контр.		Кудрявцева Е.С.								
Утверд.		Коробкова Е.А.								

ВВЕДЕНИЕ

В современном мире, где информация и творчество становятся важнейшими аспектами жизни, создание платформы для взаимодействия и обмена идеями приобретает особую значимость. Литературное творчество является одним из самых распространенных видов самовыражения, но при этом писатели, особенно начинающие, часто сталкиваются с трудностями в поиске аудитории, получении обратной связи и обмене опытом. В связи с этим разработка веб-приложения «Писательский клуб» актуальна, так как позволит создать комфортную цифровую среду для общения и взаимодействия писателей, поэтов и любителей литературы.

Актуальность разработки веб-приложения обусловлена растущей популярностью онлайн-платформ, предоставляющих возможности для творческого самовыражения и общения. С развитием технологий появляются новые способы взаимодействия между авторами и читателями, что делает необходимым создание специализированных ресурсов, объединяющих единомышленников и упрощающих процесс публикации и обсуждения публикаций пользователей.

Цель дипломного проекта – разработка веб-приложения «Писательский клуб», которое будет служить удобной платформой для проявления творческих амбиций участников, организации мероприятий и публикации текстов. В условиях цифровизации и роста популярности онлайн-сообществ, такое приложение позволит объединить людей с общими интересами и создать благоприятную среду для творчества.

Для достижения поставленной цели необходимо решить следующие задачи:

1) Анализ требований и определение функциональности:

1.1) Изучение предметной области веб-приложения «Писательский клуб».

1.2) Определение основных требований к функциональности веб-приложения.

2) Проектирование базы данных:

					ДП.09.02.07-3.25.212.02. ПЗ	Лист
						3
Изм.	Лист	№ докум.	Подпись	Дата		

2.1) Определение структуры базы данных для хранения информации о пользователях, творческих работах, комментариях и оценках публикаций.

2.2) Разработка схемы базы данных с учётом связей между сущностями.

3) Разработка пользовательского интерфейса:

3.1) Создание дизайна интерфейса для каждой из ролей пользователей (организатор, автор).

3.2) Разработка удобного и интуитивно понятного пользовательского опыта.

4) Реализация и тестирование веб-приложения:

4.1) Разработка серверной и клиентской части веб-приложения с использованием современных технологий.

4.2) Проведение функционального тестирования для проверки соответствия веб-приложения требованиям.

5) Документирование проекта:

5.1) Подготовка руководства для пользователей по использованию веб-приложения.

Разработанное веб-приложение «Писательский клуб» может быть использовано для создания и поддержки литературных сообществ, а также в образовательных и творческих проектах. Веб-приложение позволит начинающим и опытным авторам находить единомышленников, получать конструктивную обратную связь и участвовать в тематических мероприятиях.

					ДП.09.02.07-3.25.212.02. ПЗ	Лист
						4
Изм.	Лист	№ докум.	Подпись	Дата		

1 Предпроектное исследование

1.1 Описание предметной области

Предметная область включает в себя процессы, связанные с деятельностью писательского клуба. Разрабатываемое веб-приложение предназначено для поддержки и развития творчества авторов, предоставляя платформу для обмена публикациями, получения обратной связи и участия в литературных мероприятиях. В приложении предусмотрены две основные роли пользователей: администраторы, которые управляют клубом, создают тематические мероприятия и управляют заданиями к ним, и авторы, которые публикуют свои работы и участвуют в обсуждениях.

Писательский клуб представляет собой сообщество, объединяющее авторов и читателей, заинтересованных в литературном творчестве. Ключевая задача клуба — создание благоприятной среды для обмена публикациями, критики, а также проведение мероприятий, направленных на развитие писательских навыков участников.

Ключевые процессы, происходящие в рамках предметной области, включают:

- Регистрацию и авторизацию пользователей: доступ в приложение осуществляется через Telegram, что упрощает процесс идентификации и обеспечивает безопасность данных.

- Публикацию текстов: авторы могут загружать свои работы, прикреплять изображения и редактировать уже опубликованные тексты.

- Оценку и комментирование: читатели могут оставлять комментарии к публикациям и оценивать их по различным критериям.

- Участие в мероприятиях: в клубе регулярно проводятся конкурсы, тематические марафоны, в рамках которых авторы могут получать задания и представлять свои работы на суд сообщества.

					ДП.09.02.07-3.25.212.02. ПЗ	Лист
						5
Изм.	Лист	№ докум.	Подпись	Дата		

Веб-приложение предусматривает следующие роли и их взаимодействие:

– Автор: основной пользователь платформы, который публикует тексты, участвует в обсуждениях, комментирует и оценивает работы других авторов, а также может принимать участие в конкурсах и мероприятиях.

– Администратор: отвечает за управление клубом, управляет мероприятиями и пользователями, а также может участвовать в обсуждениях и оценке опубликованных текстов.

Разработка веб-приложения для писательского клуба является актуальной задачей, способствующей развитию литературного творчества, улучшению взаимодействия между авторами, а также повышению удобства управления контентом и мероприятиями. Автоматизация ключевых процессов позволяет значительно упростить работу организаторов и повысить уровень вовлеченности пользователей.

1.2 Обзор инструментов, используемых для разработки веб-приложения

Для разработки веб-приложения «Писательский клуб» понадобятся инструменты для выполнения разных задач. Для проектирования веб-приложения удобнее будет использовать Draw.io, а для дизайна – онлайн-сервис Figma.

Инструменты для проектирования:

Draw.io – бесплатный онлайн-сервис для создания диаграмм и схем. Он идеально подходит для проектирования структуры системы, визуализации алгоритмов и моделей. В проекте используется для создания диаграмм, например, ER-диаграмм для отображения взаимосвязей между сущностями базы данных, схем интерфейсов и алгоритмов.

Figma – онлайн-сервис для проектирования пользовательских интерфейсов и прототипирования. С помощью Figma можно разрабатывать как высокоуровневые прототипы, так и детализированные дизайны интерфейсов.

					ДП.09.02.07-3.25.212.02. ПЗ	Лист
						6
Изм.	Лист	№ докум.	Подпись	Дата		

Figma позволяет работать в реальном времени и совместно с другими участниками команды, что делает процесс разработки удобным и гибким. В проекте используется для создания макетов веб-интерфейса, удобных прототипов и взаимодействия с командой дизайнеров.

Технологии для клиентской части:

Vue.js – это JavaScript-фреймворк с открытым исходным кодом, предназначенный для создания пользовательских интерфейсов и одностраничных веб-приложений. Он является прогрессивным фреймворком, который обладает реактивной архитектурой и позволяет эффективно управлять состоянием приложения. Vue.js предлагает интуитивное API, хорошую производительность и масштабируемость, что делает его популярным выбором для разработки веб-интерфейсов.

HTML5 – стандартный язык гипертекстовой разметки, который используется для создания веб-страниц. Он обеспечивает структуру страницы и позволяет интегрировать различные мультимедийные элементы, такие как видео, аудио и анимации, с учетом новых возможностей HTML5.

CSS3 – язык каскадных таблиц стилей, который используется для оформления внешнего вида веб-страниц. CSS3 позволяет задать стили для элементов, таких как шрифты, цвета, размеры, а также для расположения блоков на странице. Для данного проекта используется для стилизации интерфейса, включая использование адаптивных технологий и анимаций.

JavaScript – язык программирования, который позволяет создавать динамичные и интерактивные веб-страницы. В проекте будет использоваться для реализации клиентской логики, обработки событий, взаимодействия с сервером через AJAX и улучшения взаимодействия пользователя с интерфейсом.

Технологии для серверной части:

Node.js – серверная среда выполнения JavaScript, позволяющая создавать масштабируемые сетевые приложения. Благодаря своей событийно-

					ДП.09.02.07-3.25.212.02. ПЗ	Лист
						7
Изм.	Лист	№ докум.	Подпись	Дата		

ориентированной архитектуре Node.js обеспечивает высокую производительность при работе с большим количеством запросов.

Express.js – минималистичный и гибкий фреймворк для Node.js, который упрощает создание маршрутов, работу с запросами, middleware и взаимодействие с базой данных. В проекте Express.js используется как основной инструмент для разработки серверной части и построения REST API.

MySQL – это реляционная система управления базами данных, которая используется для хранения данных проекта. В MySQL будут храниться все данные о пользователях, публикациях, комментариях и другой информации. MySQL известен своей простотой, производительностью и широкими возможностями для работы с большими объемами данных.

Инструменты для тестирования:

Jest – современный фреймворк для тестирования JavaScript-приложений. Поддерживает модульное тестирование, снапшоты и асинхронное тестирование. Используется для проверки корректности работы серверных и клиентских компонентов на ранних этапах разработки.

Инструменты для редактирования кода:

Visual Studio Code – это мощный редактор кода с открытым исходным кодом от Microsoft, который поддерживает различные языки программирования, включая PHP, HTML, CSS и JavaScript. Он включает в себя такие полезные функции, как автозавершение кода, интеграцию с Git, отладку и поддержку плагинов для расширения функционала. VS Code позволяет работать эффективно и с комфортом, используя встроенные инструменты и расширения.

Выбранные инструменты и технологии обеспечивают полноценную поддержку всех этапов разработки веб-приложения: от проектирования и дизайна до реализации серверной и клиентской логики, а также тестирования. Такой стек технологий гарантирует высокую производительность, надежность и удобство в процессе работы над проектом.

					ДП.09.02.07-3.25.212.02. ПЗ	Лист
						8
Изм.	Лист	№ докум.	Подпись	Дата		

2 Техническое задание

В начале разработки создавалось техническое задание, в котором указывались основные требования.

Для создания технического задания использовался стандарт ГОСТ 34.602-2020.

Согласно ГОСТ 34.602-2020 техническое задание должно включать следующие разделы:

- 1) Введение.
- 2) Основания для разработки
- 3) Назначение системы.
- 4) Требования к системе.
- 5) Требования к техническому оборудованию.
- 6) Требования к программному обеспечению.
- 7) Организационно-технические требования.

Техническое задание на веб-приложение представлено в приложении А.

					ДП.09.02.07-3.25.212.02. ПЗ	Лист
						9
Изм.	Лист	№ докум.	Подпись	Дата		

3 Проектирование веб-приложения

3.1 Архитектура веб-приложения

Для визуализации архитектуры веб-приложения и взаимодействия ее компонентов были разработаны диаграммы, помогающие понять структуру, функциональные возможности приложения и взаимодействие между его элементами.

Диаграмма компонентов изображена на рисунке 1. Эта диаграмма является ключевой для визуализации архитектуры приложения и взаимодействия ее основных частей. Она отображает, как организованы модули системы, как они связаны между собой и как взаимодействуют с внешними сервисами.

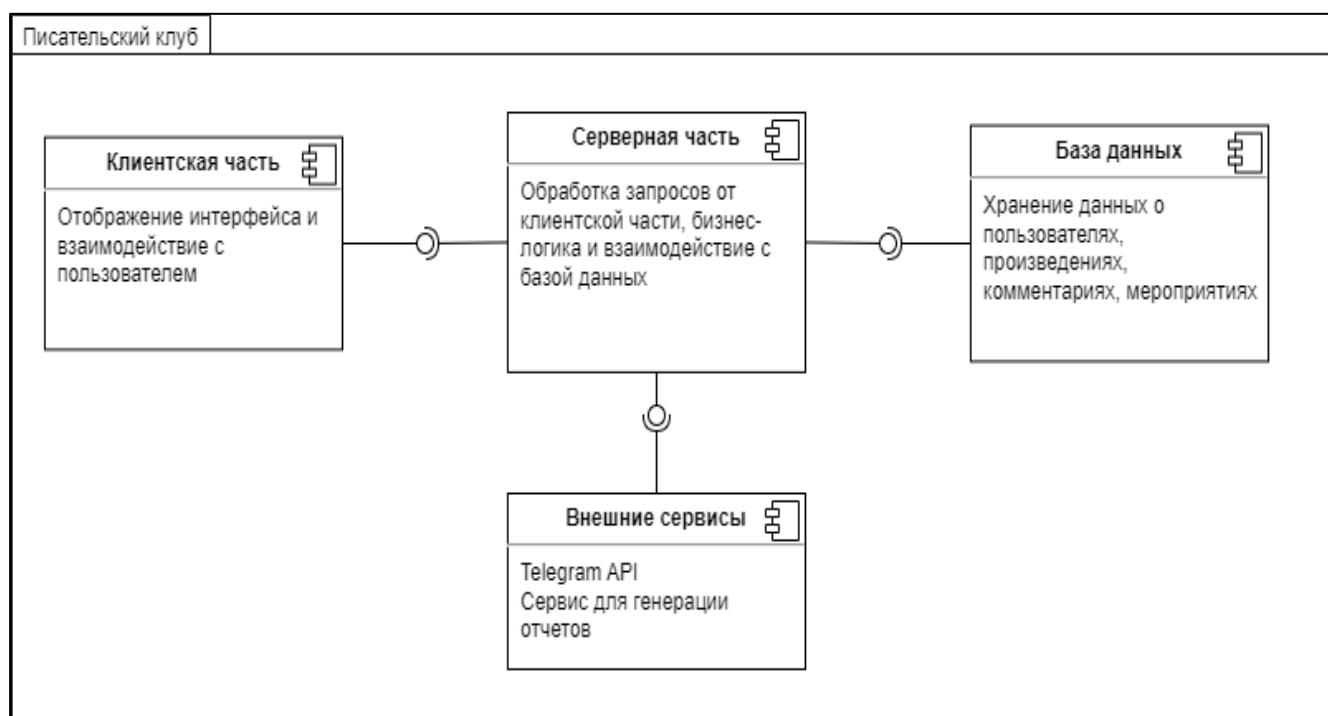


Рисунок 1 – Диаграмма компонентов веб-приложения

Клиентская часть отвечает за отображение пользовательского интерфейса и взаимодействие с пользователем, отправляет запросы на серверную часть через API, получает данные от серверной части и отображает их пользователю. Серверная часть обрабатывает поступающие от клиентской части запросы,

выполняет бизнес-логику (управление пользователями, публикациями, комментариями, оценками, мероприятиями и сборниками), взаимодействует с базой данных для хранения и извлечения данных, а также интегрируется с внешними сервисами (Telegram API для авторизации, сервис для генерации отчетов). База данных хранит все данные веб-приложения и обеспечивает целостность данных и быстрый доступ к информации. Внешние сервисы используются для авторизации пользователей через Telegram и создания отчетов в форматах PDF и Excel, используются для анализа активности пользователей и популярности публикаций.

3.2 Функциональное и структурное проектирование веб-приложения

Диаграмма прецедентов (Use Case Diagram) визуализирует функциональные возможности писательского клуба и взаимодействие пользователей с ним. На рисунке 2 изображена диаграмма прецедентов, которая описывает возможные действия с системой для таких актеров (пользователей), как Администратор и Автор.

Главными правами пользователя приложения являются: регистрация и авторизация в приложении через Telegram, в личном кабинете пользователь может изменить личную информацию, просмотреть написанные работы. Также пользователь может воспользоваться поиском, чтобы найти, прочитать чужую публикацию и прокомментировать его. Автор может просмотреть список мероприятий, заданий по мероприятиям, выбрать задание, выполнить его, написав по нему текст и опубликовать его. Также автор может написать текст вне мероприятия и опубликовать его.

Основные права организатора: управление пользователями, управление мероприятиями и получение статистики использования сайта.

					ДП.09.02.07-3.25.212.02. ПЗ	Лист
						11
Изм.	Лист	№ докум.	Подпись	Дата		

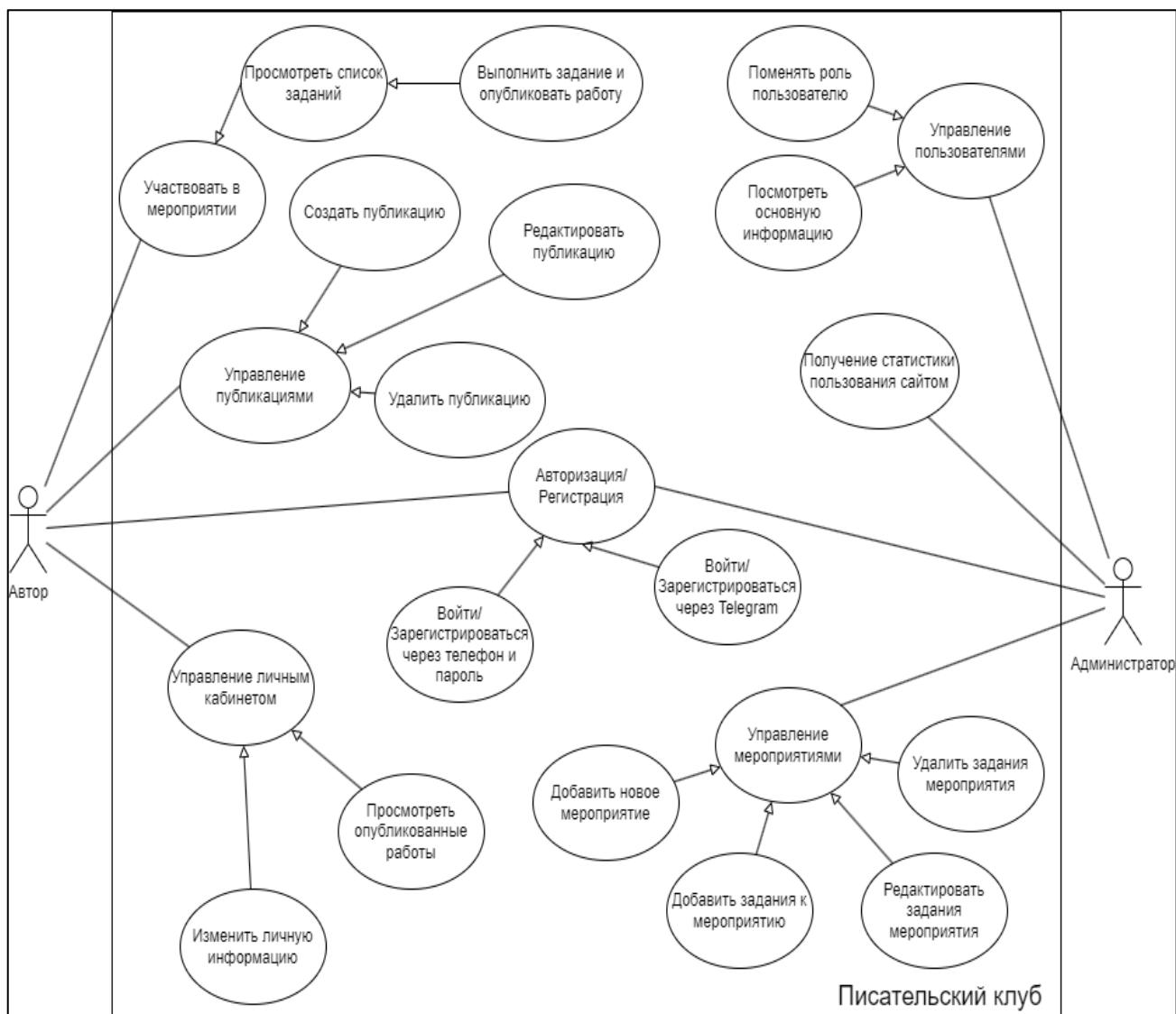


Рисунок 2 – Диаграмма прецедентов Use Case

Диаграмма деятельности описывает последовательность действий или поток выполнения в приложении. В данном проекте такая диаграмма помогает визуализировать, какие шаги выполняются для регистрации, авторизации, публикации и других процессов, а также показывает возможные параллельные или альтернативные пути выполнения. Диаграмма деятельности позволяет детально проработать шаги каждого процесса, выявить возможные сложности, а также организовать последовательность и условия выполнения действий.

На Рисунке 3 изображена диаграмма деятельности, которая показывает основной процесс регистрации и авторизации пользователя через Telegram.

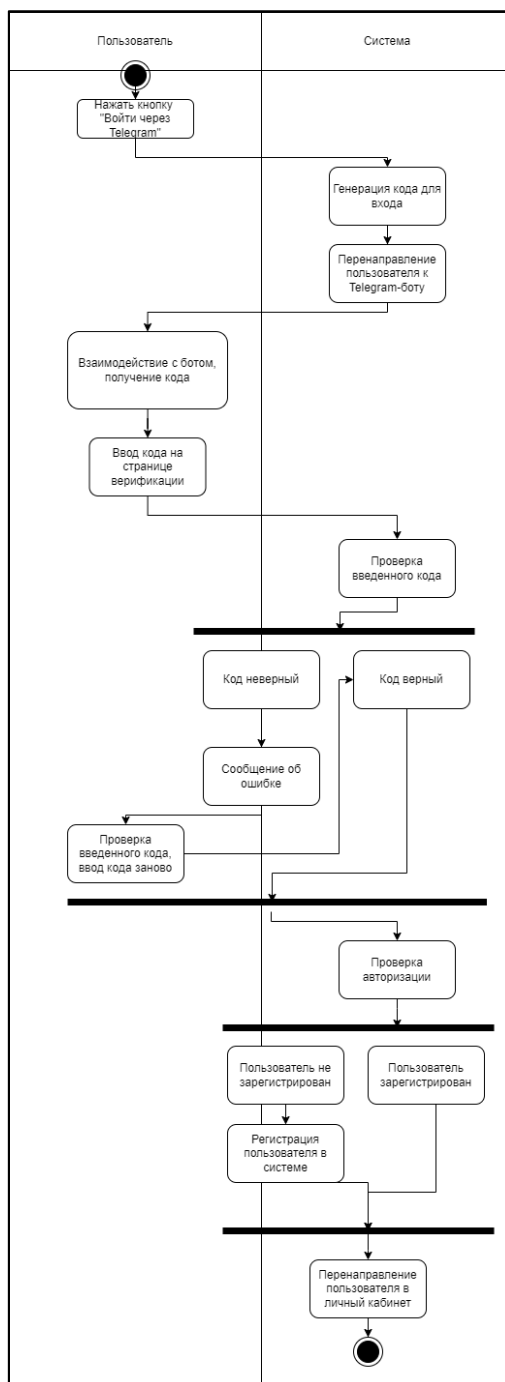


Рисунок 3 – Диаграмма деятельности

Представленные диаграммы дают визуальное представление о взаимодействии пользователя с веб-приложением «Писательский клуб», основных процессах работы с профилем, мероприятиями и публикациями, а также представлен основной процесс регистрации и авторизации пользователя в системе через Telegram.

3.3 Проектирование базы данных веб-приложения

Создание ER-модели также является важным этапом разработки веб-приложения. ER-модель помогает описать структуру данных, сущности и их взаимосвязи в базе данных информационной системы.

ER-модель (Entity-Relationship) – это концептуальное представление структур данных, используемое для организации и структурирования данных в базе данных. Это графическое представление сущностей, атрибутов и отношений между ними. ER-модели необходимы для проектирования и реализации баз данных, обеспечения согласованности данных и облегчения общения между заинтересованными лицами.

При проектировании базы данных для веб-приложения была создана ER-модель базы данных, изображенная на рисунке 4, в которой указаны таблицы БД, их атрибуты и типы данных.

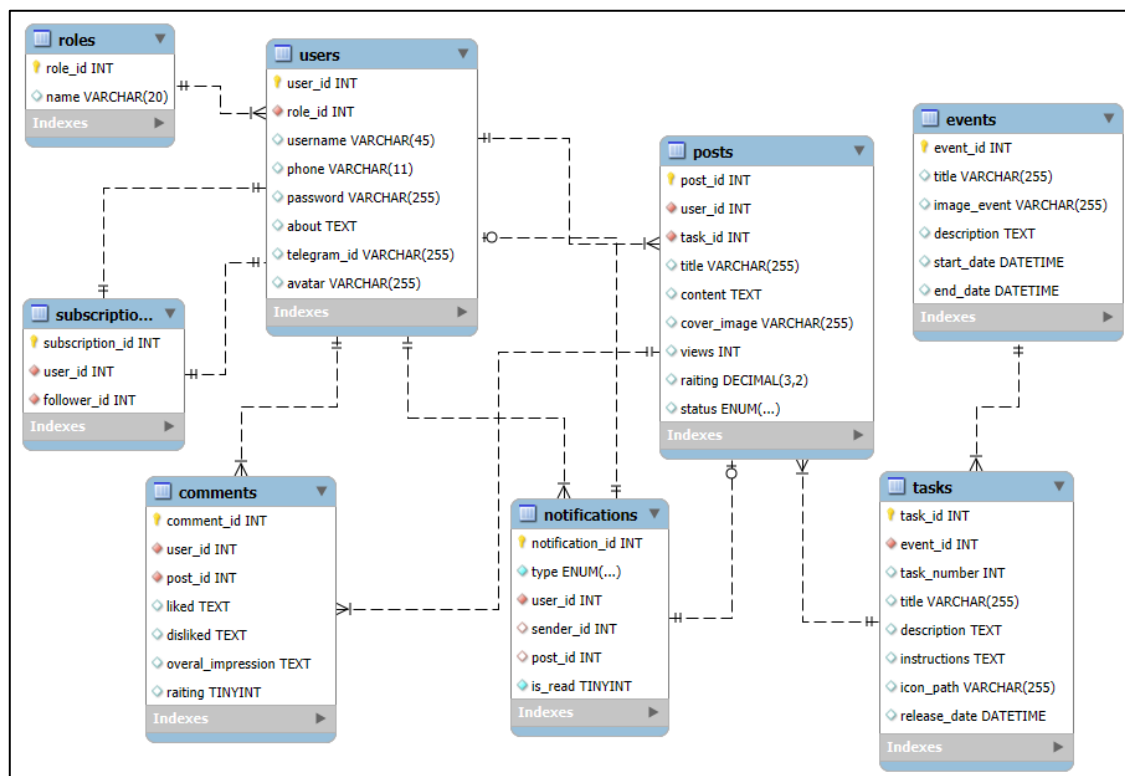


Рисунок 4 – ER-модель базы данных

В таблицах 1-8 представлено описание ER-модели.

Таблица 1 – Пользователи (users)

Поле	Тип данных	Обозначение
user_id	Целое число (INT)	Уникальный идентификатор пользователя
role_id	Целое число (INT)	Внешний ключ на таблицу ролей
username	Текст (VARCHAR(45))	Имя пользователя
phone	Текст (VARCHAR(11))	Номер телефона пользователя
password	Текст (VARCHAR(255))	Хэшированный пароль пользователя
about	Длинный текст (TEXT)	Информация пользователя о себе
telegram_id	Текст (VARCHAR(255))	Идентификатор для авторизации через Telegram
avatar	Текст (VARCHAR(255))	Путь к аватару пользователя

Таблица 2 – Публикации (posts)

Поле	Тип данных	Обозначение
post_id	Целое число (INT)	Уникальный идентификатор публикации
user_id	Целое число (INT)	Идентификатор пользователя-автора
task_id	Целое число (INT)	Идентификатор задания, если текст был написан для конкурса
title	Текст (VARCHAR(255))	Заголовок публикации
content	Длинный текст (TEXT)	Основной текст публикации
cover_image	Текст (VARCHAR(255))	Путь к обложке публикации
views	Целое число (INT)	Количество просмотров
rating	Десятичное число (DECIMAL(3,2))	Средний рейтинг публикации
status	Перечисляемый тип (ENUM)	Статус текста ('published', 'draft')

Таблица 3 – Комментарии (comments)

Поле	Тип данных	Обозначение
comment_id	Целое число (INT)	Уникальный идентификатор комментария
user_id	Целое число (INT)	Идентификатор пользователя-комментатора
post_id	Целое число (INT)	Идентификатор публикации, к которому относится комментарий

Продолжение таблицы 3

Поле	Тип данных	Обозначение
liked	Длинный текст (TEXT)	Что понравилось
disliked	Длинный текст (TEXT)	Что не понравилось
overall_impression	Длинный текст (TEXT)	Впечатления комментатора
rating	Маленькое целое число (TINYINT)	Рейтинг комментария (от 1 до 5)

Таблица 4 – Мероприятия (events)

Поле	Тип данных	Обозначение
event_id	Целое число (INT)	Уникальный идентификатор мероприятия
title	Текст (VARCHAR(255))	Название мероприятия
image_event	Текст (VARCHAR(255))	Путь к изображению мероприятия
description	Длинный текст (TEXT)	Краткое описание мероприятия
start_date	Дата и время (DATETIME)	Дата начала мероприятия
end_date	Дата и время (DATETIME)	Дата окончания мероприятия

Таблица 5 – Задания мероприятий (tasks)

Поле	Тип данных	Обозначение
task_id	Целое число (INT)	Уникальный идентификатор задания
event_id	Целое число (INT)	Идентификатор мероприятия, к которому принадлежит задание
task_number	Целое число (INT)	Номер задания в рамках мероприятия
Поле	Тип данных	Обозначение
title	Текст (VARCHAR(255))	Название задания
description	Длинный текст (TEXT)	Краткое описание задания
instructions	Длинный текст (TEXT)	Подробное описание задания
icon_path	Текст (VARCHAR(255))	Путь к иконке задания
release_date	Дата и время (DATETIME)	Дата выхода задания

Таблица 6 – Уведомления (notifications)

Поле	Тип данных	Обозначение
notification_id	Целое число (INT)	Уникальный идентификатор уведомления
user_id	Целое число (INT)	Идентификатор пользователя-получателя уведомления
type	Перечисляемый тип (ENUM)	Тип уведомления ('subscription', 'comment', 'system')
sender_id	Целое число (INT)	Идентификатор пользователя-отправителя уведомления
post_id	Целое число (INT)	Идентификатор поста, связанного с уведомлением
is_read	Маленькое число (TINYINT)	Флаг, обозначающий, было ли уведомление прочитано

Таблица 7 – Подписки (subscriptions)

Поле	Тип данных	Обозначение
subscription_id	Целое число (INT)	Уникальный идентификатор подписки
user_id	Целое число (INT)	Идентификатор пользователя
follower_id	Целое число (INT)	Идентификатор подписчика

Таблица 8 – Роли (roles)

Поле	Тип данных	Обозначение
role_id	Целое число (INT)	Уникальный идентификатор роли
name	Текст (VARCHAR(20))	Название роли

3.4 Проектирование пользовательского интерфейса веб-приложения

3.4.1 Разработка прототипов пользовательского интерфейса

Для разработки пользовательского интерфейса веб-приложения «Писательский клуб» был использован такой инструмент профессиональной разработки прототипов пользовательского интерфейса (UI) и дизайна интерфейсов, как онлайн-сервис Figma.

Прототип главной страницы писательского клуба изображен на рисунке 5.

В верхней левой части располагается название клуба и навигационная панель со ссылками на разные страницы. Рядом с таймером расположен блок с анонсом этого мероприятия. При нажатии на кнопку «Подробнее» пользователю открывается модальное окно с кратким описанием предстоящего мероприятия. Посередине вверху располагается секция с социальными сетями, которые предоставляет владелец клуба. Справа вверху расположен контейнер с именами последних семи зарегистрированных пользователей.

Ниже этой секции располагаются популярные тексты пользователей, которые набрали наибольшее количество просмотров на все время существования на платформе.

Посередине изображен логотип писательского клуба и под ним располагается слоган.

Ниже находится секция с полем для поиска текста по ключевому слову.

Кнопка авторизации может меняться в зависимости от того, авторизован ли пользователь. Если пользователь не авторизован, то он переходит на страницу авторизации, где сможет зарегистрироваться или войти в личный кабинет. Если пользователь авторизован, то вместо кнопки для регистрации там кнопка для перехода в личный кабинет с надписью: «Привет, username!».

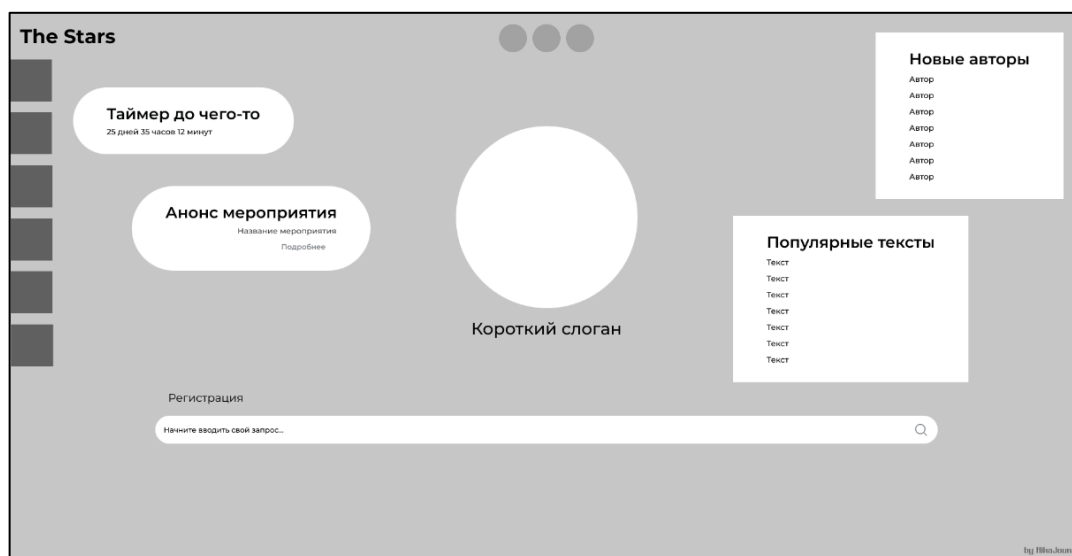


Рисунок 5 – Прототип главной страницы писательского клуба

					ДП.09.02.07-3.25.212.02. ПЗ	Лист
						18
Изм.	Лист	№ докум.	Подпись	Дата		

Прототип страницы результатов поиска изображен на рисунке 6.

Здесь также присутствует навигационная панель, а также поле для поиска с параметрами для сортировки результатов.

Результаты поиска располагаются ниже, они располагаются в два ряда на странице в компоненте с горизонтальной прокруткой.

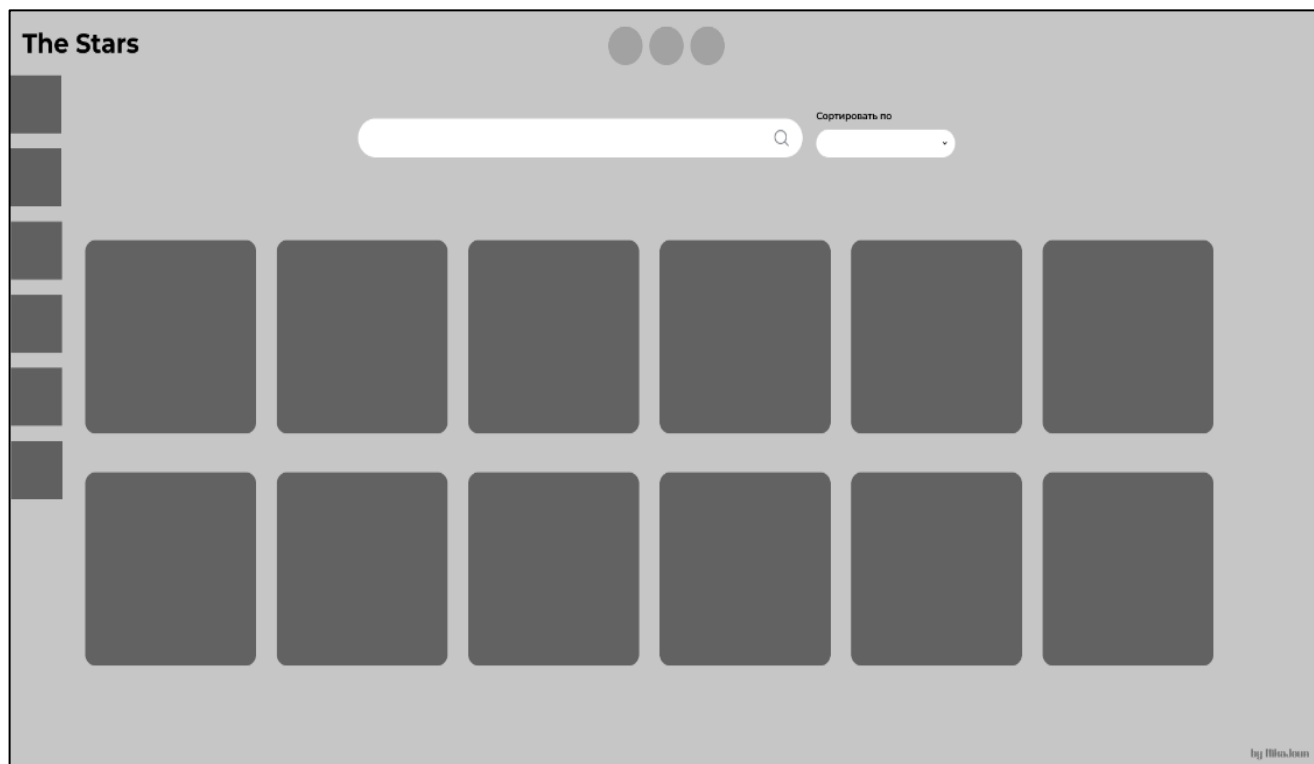


Рисунок 6 – Прототип страницы результатов поиска

На рисунке 7 изображен прототип страницы публикации.

Текст находится в отдельной главной секции, у него есть автор и название.

Здесь пользователь может написать комментарий, подробно поделившись, что ему понравилось, что не понравилось в тексте, выразить свое впечатление от прочитанного и поставить звездный рейтинг. Такой формат комментария даст хорошую обратную связь автору и станет мягкой критикой.

Автор

Название

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nunc malesuada, enim sed semper scelerisque, tellus nulla lacinia nisi, eget viverra urna odio nec leo. Curabitur eleifend libero a nunc vehicula ullamcorper. Ut a fringilla urna, ac condimentum massa. Ut euismod est massa, sed porta dolor rhoncus nec. Nunc pellentesque id turpis at pulvinar. Phasellus justo ante, gravida et turpis quis, feugiat placerat mauris. In ultrices, turpis non malesuada vehicula, elit lectus blandit metus, at ultricies mi quam eget orci. Praesent mi nulla, vehicula sit amet congue eu, luctus sit amet nisi. Donec a massa at eros ultrices consequat. Suspendisse molestie commodo eros nec lacinia. Aenean vel nulla sodales, rhoncus dolor quis, porta eros. Vivamus sit amet convallis tellus, non pellentesque dui.

Ut ac libero a augue placerat venenatis. Morbi ipsum nibh, placerat consectetur est et, rhoncus vestibulum massa. Curabitur gravida massa eu facilisis molestie. Proin sit amet lectus gravida lectus suscipit vestibulum auctor eu elit. Pellentesque tincidunt in lorem eget rutrum. Curabitur sed lacus malesuada, convallis diam nec, feugiat arcu. Proin viverra tempus congue. Maecenas purus tortor, facilisis id laoreet ut, laoreet sed dui. Aliquam fringilla consectetur arcu, id tristique massa pharetra vel. Vivamus interdum sem nec metus tristique dictum. Sed fringilla diam a turpis scelerisque tempor. Sed dapibus quis elit et porttitor. Nulla ipsum mi, tristique non pharetra non, congue in enim. Duis lacinia pharetra vestibulum. Integer et tincidunt ex.

Donec efficitur, tellus a pulvinar pharetra, sem odio egestas nunc, eget posuere lorem lectus eu eros. Donec quis turpis a dolor cursus fermentum ornare sodales felis. Etiam ut eros libero. Proin pellentesque sit amet tortor quis maximus. Mauris a finibus tortor, ac volutpat nisi. Vivamus porttitor malesuada diam, a accumsan nisi iaculis vitae. Suspendisse convallis porttitor leo a egestas. Nullam iaculis lectus nec posuere malesuada. Lorem ipsum dolor sit amet, consectetur adipiscing elit. Sed sed leo et est rhoncus varius vel id est. Cras ut ornare nibh. Quisque sit amet ex vitae est varius iaculis. Curabitur sagittis bibendum cursus. Nunc ultrices odio vehicula rhoncus molestie. In scelerisque vel mauris non interdum. Nunc quis ligula eget lorem volutpat ullamcorper eu eu risus.

Phasellus interdum, libero vel auctor vestibulum, nisi dui vehicula urna, a consequat sem nibh nec justo. Duis bibendum ut metus id fermentum. Aliquam erat volutpat. Etiam lorem metus, volutpat eget orci et, euismod aliquet odio. Nullam consequat metus turpis. Aliquam condimentum magna ac ante varius finibus. Pellentesque nec dignissim odio. Proin vestibulum, orci eget gravida blandit, ante lacus posuere libero, quis gravida quam leo nec dolor.

Pellentesque ac libero fermentum, malesuada tortor et, congue ipsum. Etiam vitae nisi vel dui pulvinar porta. Nulla sed libero mi. In ut risus vel erat posuere pulvinar. Etiam eget tristique massa. Nunc dictum varius nibh vel lacinia. Aliquam venenatis enim sed ante congue, vel mattis elit auctor. Fusce nec turpis ac risus mollis venenatis in at tortor. Vivamus arcu ligula, volutpat vehicula convallis in, auctor sed lacus. Morbi est ligula, iaculis at bibendum id, ornare nec turpis. Fusce sit amet lectus quis erat sodales convallis eleifend vitae mauris. Aenean vitae rutrum urna, non fermentum ex. Cras porttitor quam lectus, sit amet tempor augue maximus gravida. Etiam sit amet hendrerit lectus, at malesuada arcu. Integer et purus lectus. Proin dapibus dignissim lectus, eget tincidunt nulla vestibulum vitae.

Curabitur ullamcorper dictum magna convallis semper. Suspendisse non sem dolor. Suspendisse euismod venenatis tortor, ac suscipit dui finibus et. Nullam vitae nisi non massa tempor malesuada. Donec et augue eget mi mollis cursus ut quis purus. Quisque feugiat pulvinar commodo. Morbi ultricies blandit libero nec ultrices. Nunc semper dolor lacus, sit amet efficitur eros dictum eget. Donec at magna vitae quam dictum volutpat.

Ваш комментарий ☆☆☆☆

Расскажите, что вам понравилось

Расскажите, что вам не понравилось

Расскажите ваши впечатления

Опубликовать

Комментатор

☆☆☆☆

Понравилось:
Шень Цинъю действительно очень строгий учитель, однако именно с Бинкэ он был очень мил

Не понравилось:
ООС совсем не в характере Бинкэ. Понимаю, что в жанрах есть ООС, но все же, не мое...

Впечатления:
Чуткий и нежный текст. Несмотря на мой нелюбимый жанр, готова прочитать его еще раз

by Hhshh.com

Рисунок 7 – Прототип страницы публикации

Прототип страницы авторизации изображен на рисунке 8.

На странице авторизации пользователю предлагается войти или зарегистрироваться через такую социальную сеть, как Telegram.

Прототип страницы личного кабинета изображен на рисунке 9.

В личном кабинете присутствуют такие секции, как информация о пользователе, его подписки и подписчики, накопленная внутренняя валюта, отдельная секция со ссылками на другие страницы, а также список его текстовых работ.

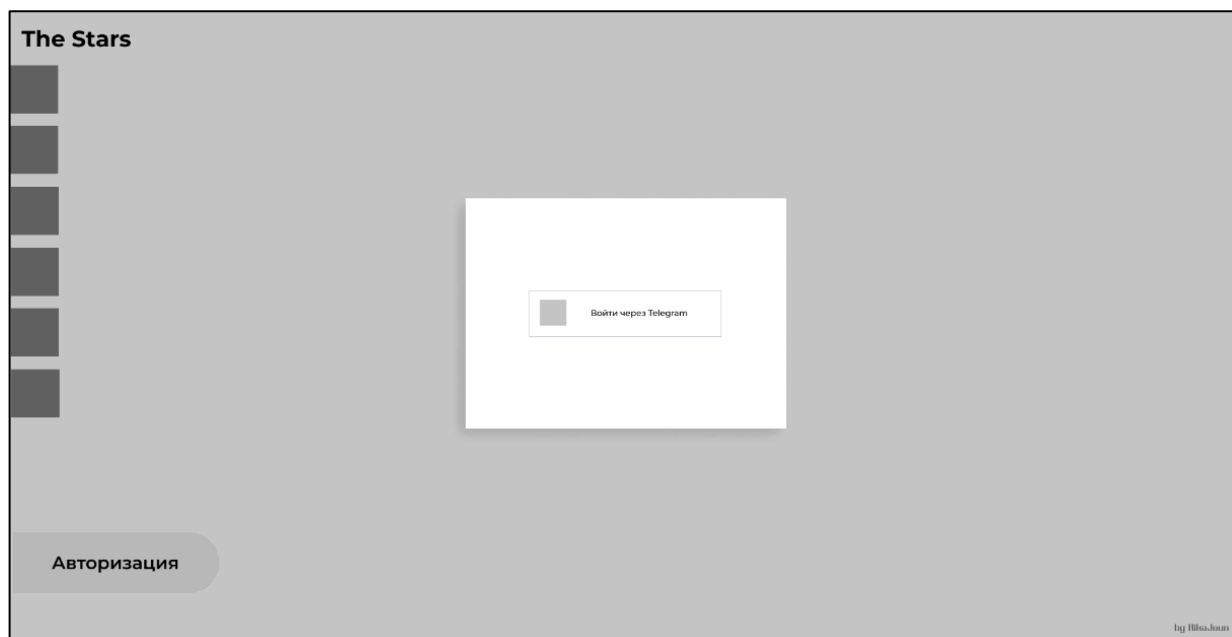


Рисунок 8 – Прототип страницы авторизации

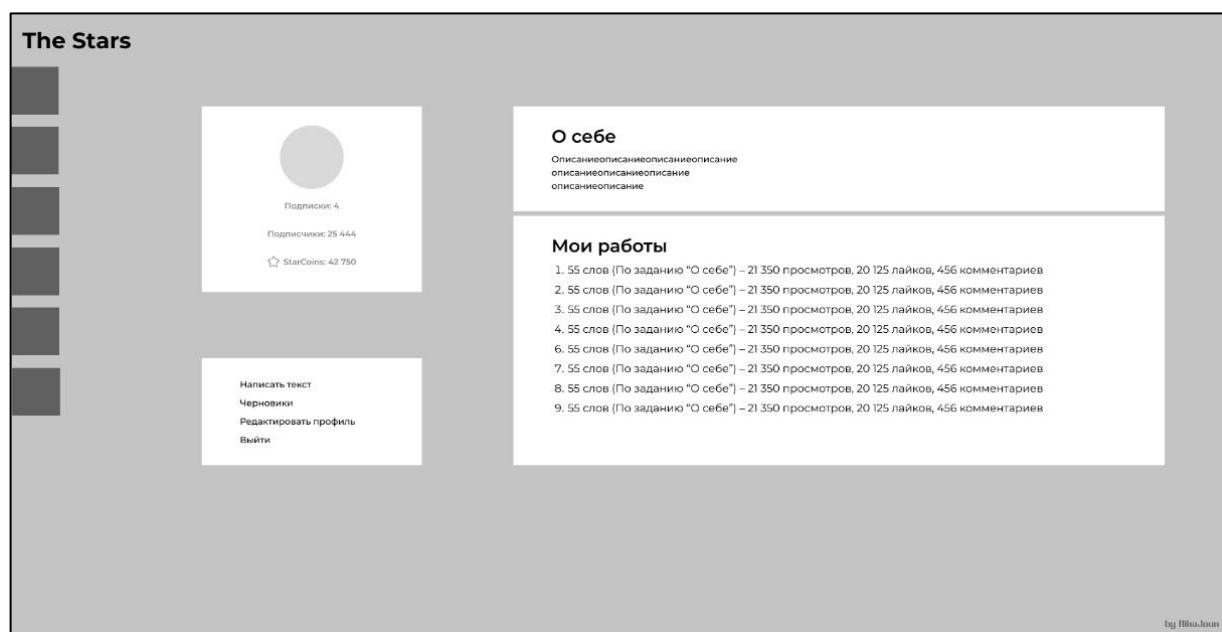


Рисунок 9 – Прототип страницы личного кабинета

На рисунке 10 изображен прототип страницы редактора текста.

На странице присутствует большое текстовое поле для ввода основного текста по заданию, текстовое поле для ввода заголовка, кнопка для добавления обложки к тексту и кнопка «Опубликовать».

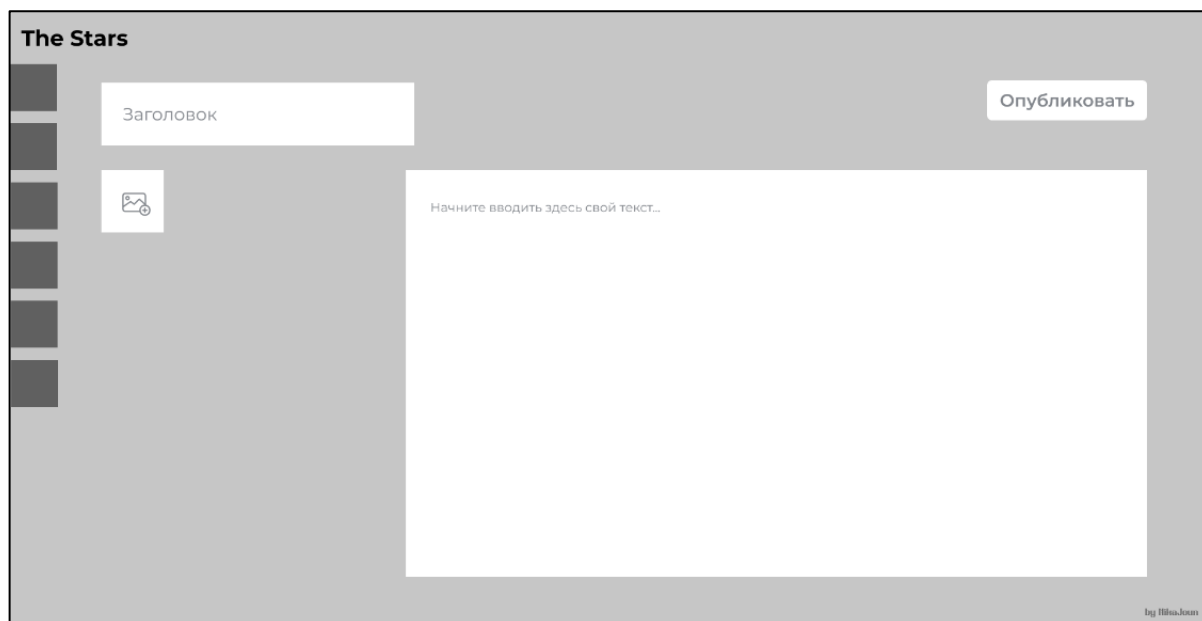


Рисунок 10 – Прототип страницы редактора текста

На рисунке 11 изображен прототип страницы активного мероприятия и его заданий. Здесь располагается краткая информация о тематическом конкурсе, включающая его картинку, название, краткое описание и даты проведения. В отдельной секции находятся задания – их название, описание, инструкции, награда.

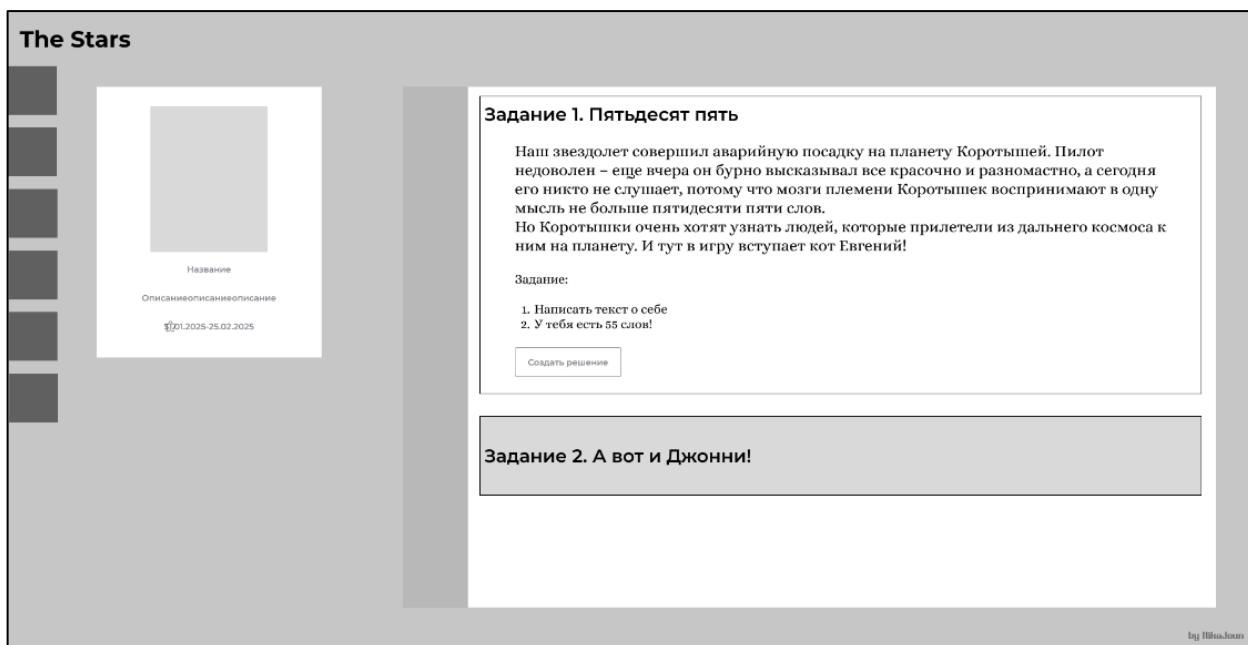


Рисунок 11 – Прототип страницы мероприятий

3.4.2 Выбор цветовой гаммы и шрифтов

Для проектирования дизайн-макетов веб-приложения «Писательский клуб» была выбрана следующая цветовая гамма, изображенная на рисунке 12.

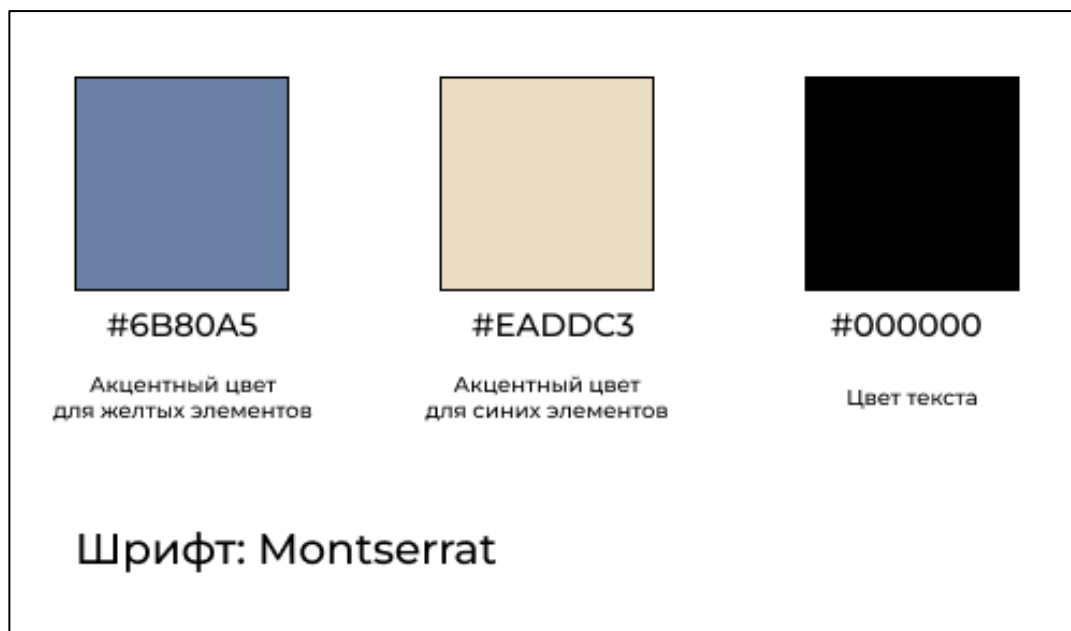


Рисунок 12 – Цветовая гамма и шрифт

Выбранная цветовая гамма:

- Акцентный цвет №1: #6B80A5.
- Акцентный цвет №2: #EADDC3.

Поскольку сам писательский клуб выбрал свое название «The Stars», эти цвета как нельзя кстати создают ощущение вечернего неба. Первый цвет создает ощущение стабильности и профессионализма, он напоминает оттенки вечернего неба или глубоких вод, что придает интерфейсу стильный и сдержанный вид, подчеркивая серьезность проекта и ориентацию на литературную аудиторию. Ему в противовес стоит второй цвет: теплый оттенок слоновой кости, что придает дизайну чувство уюта и мягкости. Этот цвет добавляет легкость, создавая приятный контраст с более глубоким тоном первого цвета. Вместе они формируют гармоничную цветовую палитру, которая настраивает пользователей на

комфортное взаимодействие с информационной системой и способствует легкому восприятию контента.

Шрифт Montserrat, с его современным и минималистичным стилем, прекрасно сочетается с выбранной цветовой гаммой. Его чистые линии и сбалансированные пропорции добавляют стиль и элегантность к дизайну, делая его более привлекательным и современным для восприятия.

Эти элементы цветовой гаммы и шрифта могут быть использованы для создания единого стиля и атмосферы на сайте писательского клуба, не только придав уникальность и красоту, но также передавая определенные эмоциональные и эстетические характеристики, которые могут усилить воздействие на аудиторию и создать запоминающийся пользовательский опыт.

3.4.3 Разработка элементов интерфейса

Для создания привлекательного дизайна веб-приложения с созданными прототип-макетами и выбранным цветовым решением, были созданы такие элементы интерфейса.

Логотип писательского клуба изображен на рисунке 13 и представляет собой два варианта: большой, круглый с изображением звезд и облака логотип создан специально для главной страницы и название, написанное шрифтом Natigek Batagor для верхней части страницы.



Рисунок 13 – Логотипы

					ДП.09.02.07-3.25.212.02. ПЗ	Лист
						24
Изм.	Лист	№ докум.	Подпись	Дата		

На рисунке 14 изображено текстовое поле, используемое для поиска публикаций. Оно полупрозрачное, подсказка для ввода серая, есть кнопка в виде лупы.



Рисунок 14 – Текстовое поле поиска

На рисунке 15 предоставлен элемент результат поиска. На фоне изображение обложки, само название, имя автора и количество просмотров.

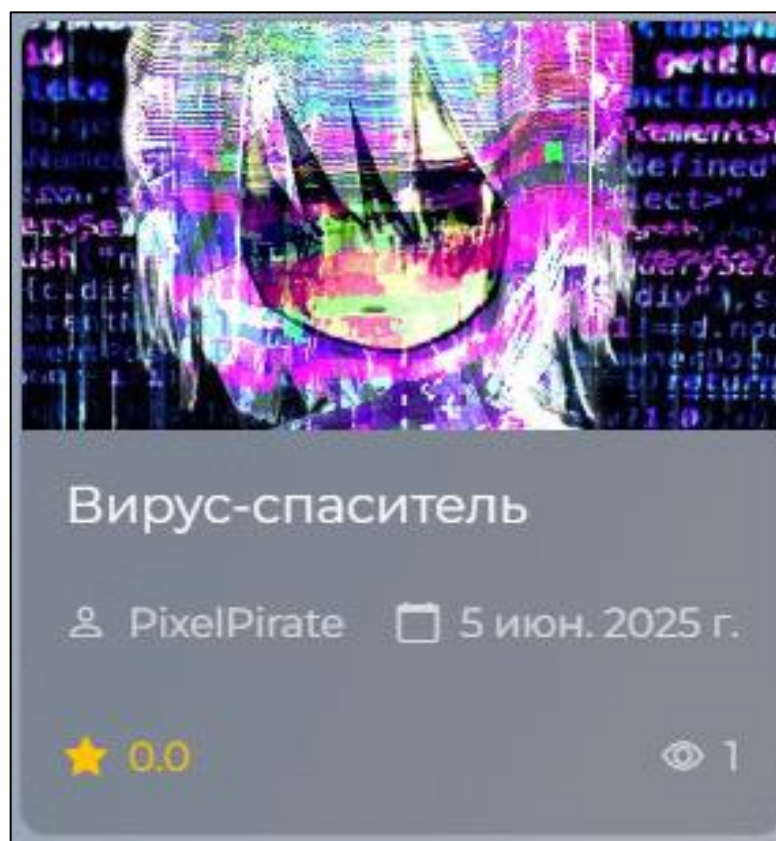


Рисунок 15 – Элемент результат поиска

Элемент анонса события изображен на рисунке 16. Здесь есть заголовок, описание события, дата начала и дата окончания мероприятия, и изображение, которое подходит событию.

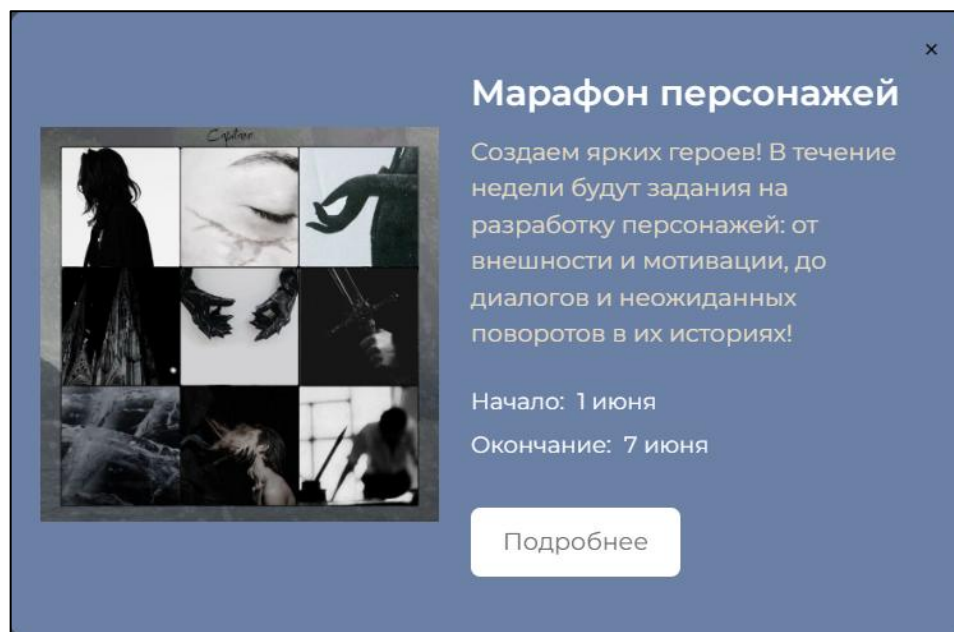


Рисунок 16 – Элемент анонса мероприятия

3.4.4 Разработка дизайн-макетов интерфейса пользователя

Прежде чем приступать к разработке дизайн-макетов веб-приложения, была разработана схема навигации, изображенная на рисунке 17. На схеме изображены переходы между страницами сайта писательского клуба.



Рисунок 17 – Схема навигации

На рисунке 18 изображен дизайн-макет главной страницы писательского клуба. На фоне изображен градиент с облаками, который создает ощущение вечернего неба. Применены дизайн логотипов, поискового текстового поля, а также выбранные цветовая гамма и шрифт.

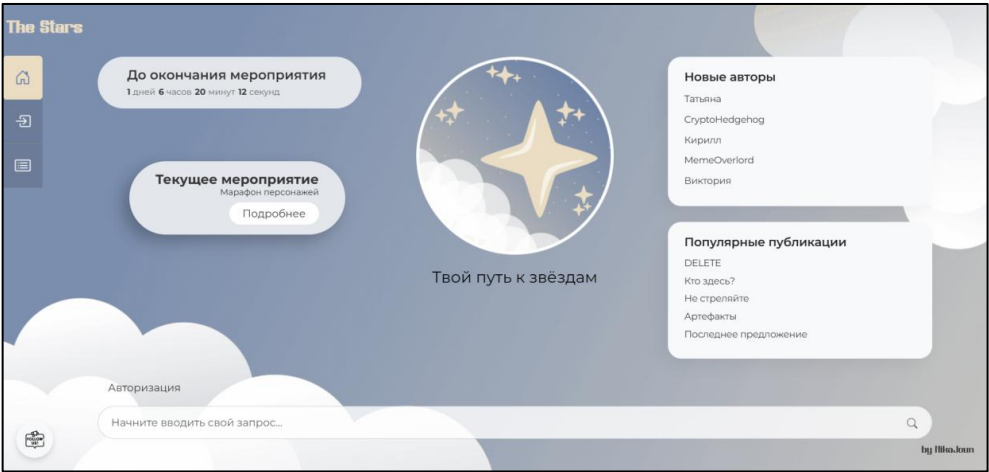


Рисунок 18 – Дизайн-макет главной страницы

Страница результатов поиска изображена на рисунке 19. Здесь также применены выбранная цветовая гамма и шрифт, элементы для результатов поиска, а также маленький логотип и текстовые поля.

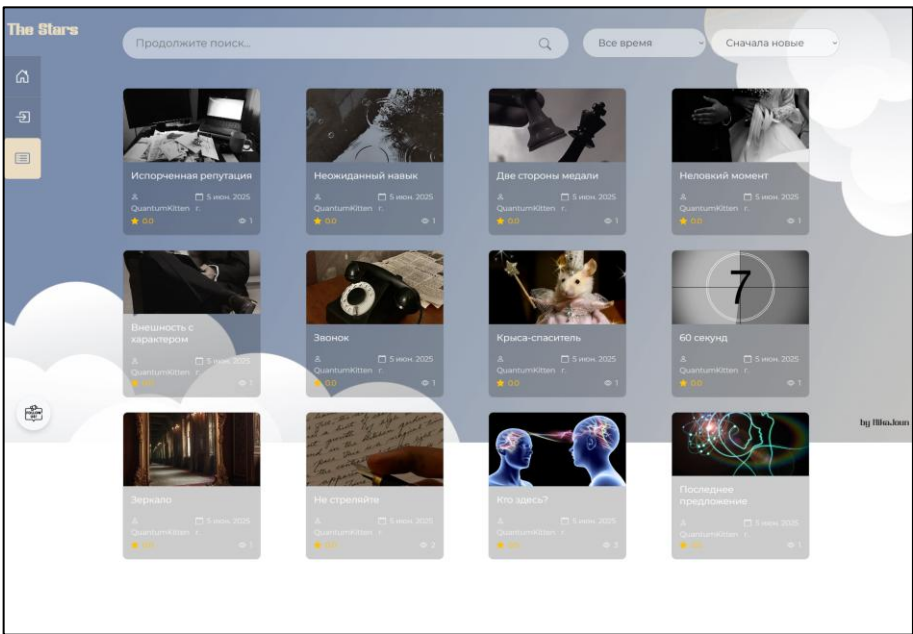


Рисунок 19 – Дизайн-макет страницы результатов поиска

Дизайн-макет страницы публикаций изображен на рисунке 20. На этом макете также применены цветовая гамма и шрифт, логотип, применены минималистичные стили к кнопкам. У секции с текстом полупрозрачный белый фон.

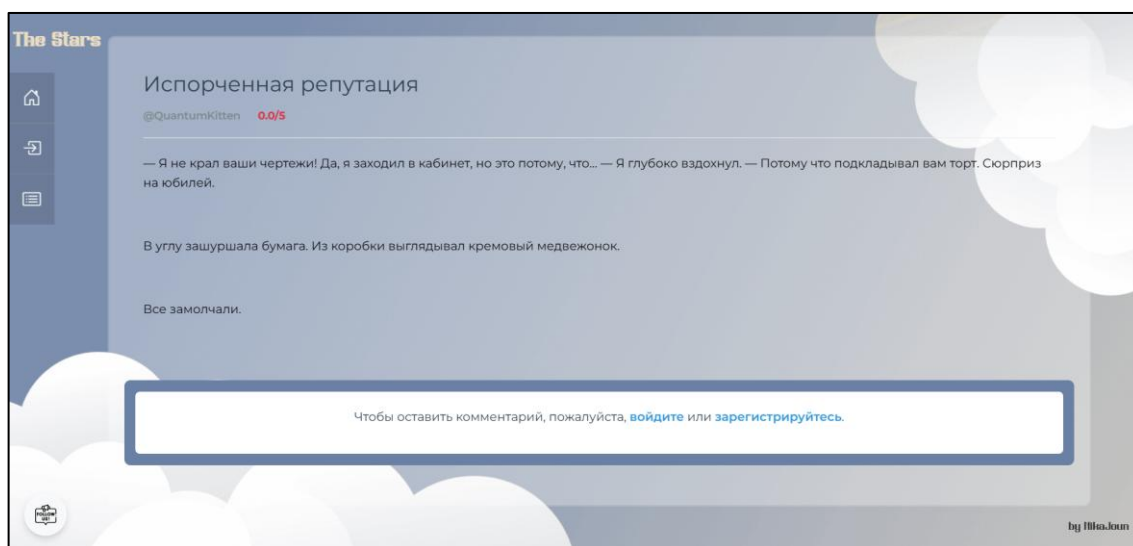


Рисунок 20 – Дизайн-макет страницы публикации

На рисунке 21 изображен дизайн-макет страницы авторизации. Добавлены поля для возможности авторизации и регистрации через номер телефона и пароль, а не Telegram.

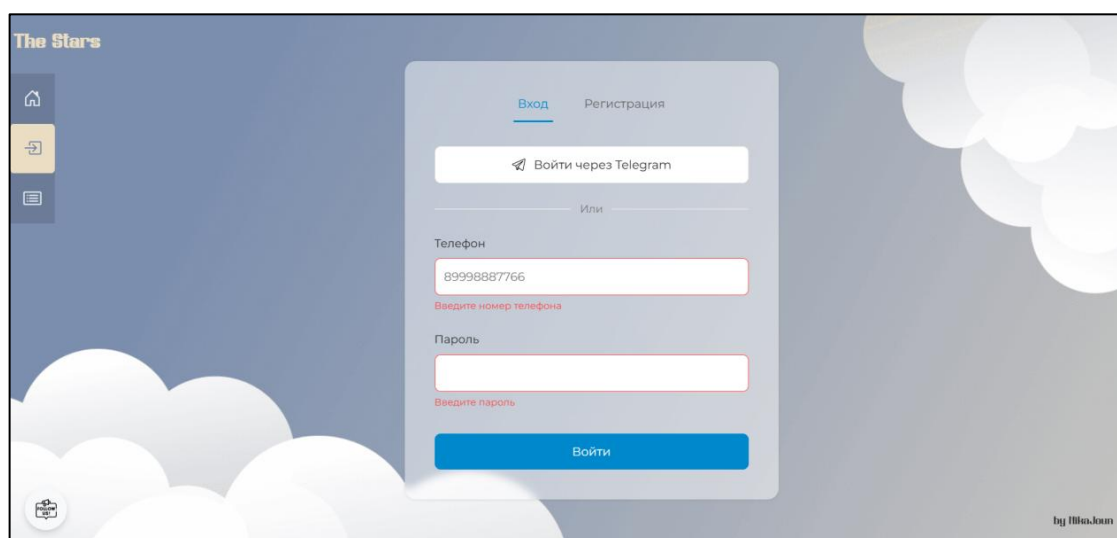


Рисунок 21 – Дизайн-макет страницы авторизации

На рисунке 22 представлен дизайн-макет страницы личного кабинета пользователя.

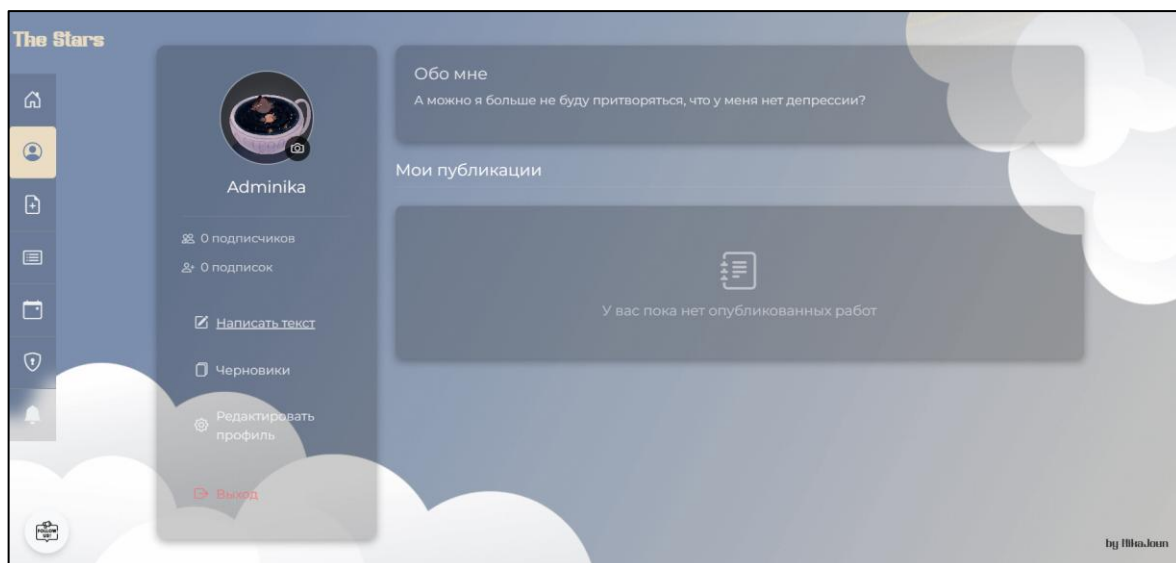


Рисунок 22 – Дизайн-макет личного кабинета пользователя

На рисунке 23 изображена страница внутреннего текстового редактора. Ей автор может воспользоваться, чтобы написать собственный текст и опубликовать его в системе. Немного изменена структура в угоду удобства для пользователя.

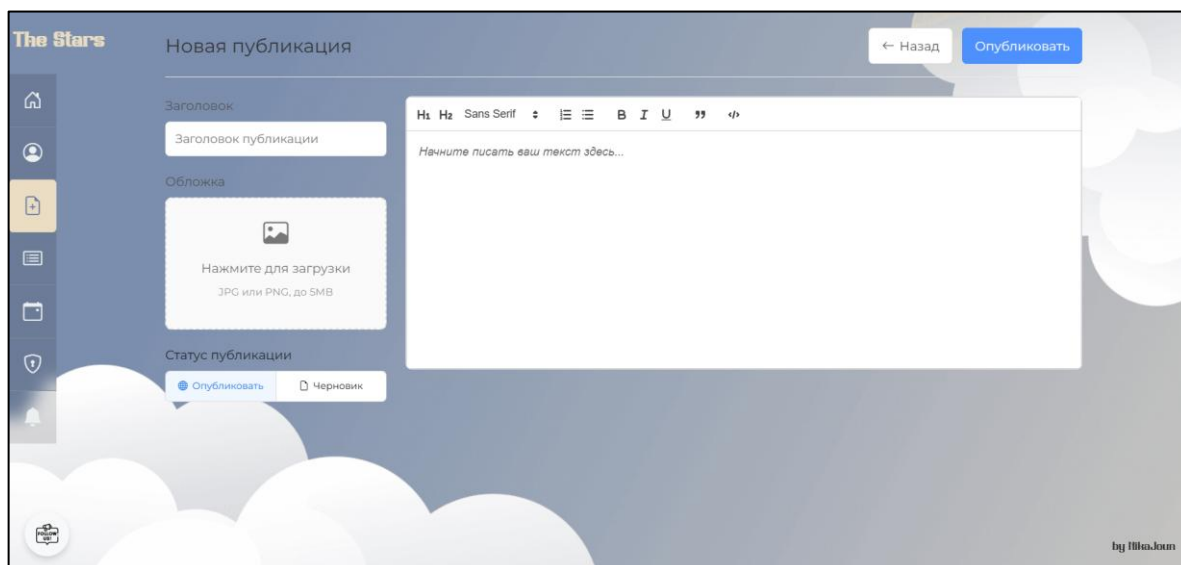


Рисунок 23 – Дизайн-макет страницы текстового редактора

Дизайн-макет страницы мероприятий изображен на рисунке 24.

					ДП.09.02.07-3.25.212.02. ПЗ	Лист
						29
Изм.	Лист	№ докум.	Подпись	Дата		

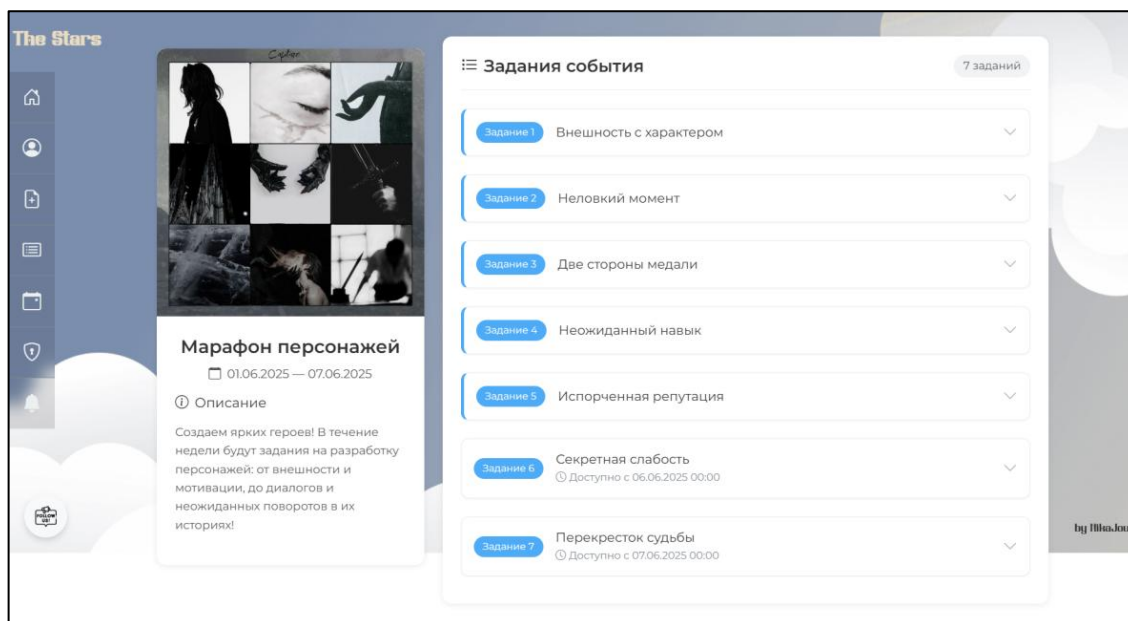


Рисунок 24 – Дизайн-макет страницы мероприятий

Был выбран инструмент для разработки пользовательского интерфейса веб-приложения, были созданы прототипы страниц, выбраны цветовая гамма и шрифт, созданы элементы интерфейса пользователя и дизайн-макеты всех страниц.

4 Реализация веб-приложения

4.1 Разработка интерфейса веб-приложения

На этапе проектирования интерфейса веб-приложения было принято решение использовать современный JavaScript-фреймворк Vue.js. Vue.js представляет собой компонентно-ориентированную архитектуру, которая позволяет разбивать интерфейс на независимые, переиспользуемые модули. Каждый компонент включает в себя разметку, логику и стили, что способствует высокой модульности, изоляции и масштабируемости проекта.

Базовая структура компонента на Vue.js включает три основные секции:

1) `<template>` – отвечает за разметку интерфейса. Здесь описывается HTML-структура компонента, которая может включать как статические элементы, так и динамические связанные данные и события.

2) `<script setup>` – используется для описания логики компонента. Здесь подключаются другие компоненты, определяются переменные, методы, события и взаимодействие с маршрутизатором.

3) `<style>` – отвечает за оформление интерфейса. Используется CSS. Стили можно ограничить областью действия текущего компонента с помощью директивы `scoped`.

Ключевым компонентом интерфейса является обертка страницы – `App.vue`, которая объединяет основные визуальные и логические элементы. На рисунке 25 представлен фрагмент кода с описанием его структуры.

В компоненте реализована плавающая кнопка быстрого доступа к социальным сетям. Состояние открытия кнопки управляется через реактивную переменную `isFabOpen`, определенную с помощью Composition API.

Компоненты интерфейса:

1) `AppNavbar.vue` – компоненты навигационной панели, включающий логотип и ссылки на страницы.


```

1  <template>
2    <div class="app-container">
3      <AppNavbar />
4
5      <div class="fab-bottom-left" :class="{ open: isFabOpen }">
6        <button class="fab-main" @click="toggleFab">
7          
8        </button>
9        <div class="fab-actions-right">
10         <a href="https://vk.com/nikajoun" target="_blank" class="fab-icon vk">
11           
12         </a>
13         <a href="https://t.me/nikajoun" target="_blank" class="fab-icon tg">
14           
15         </a>
16         <a href="mailto:viktoriaantonin@gmail.com" target="_blank" class="fab-icon email">
17           
18         </a>
19       </div>
20     </div>
21
22     <div class="content">
23       <router-view />
24     </div>
25
26     <div class="footer">by NikaJoun</div>
27   </div>
28 </template>

```

Рисунок 25 – Структура главного компонента App.vue

2) <router-view /> – маршрутизатор Vue Router подставляет в этот блок контент в зависимости от активного маршрута. Это позволяет реализовать SPA (Single Page Application) без перезагрузки страницы.

3) Footer – фиксированный элемент с подписью разработчика.

4.2 Реализация базы данных веб-приложения

База данных разработана для хранения информации о пользователях, мероприятиях, текстах, комментариях и уведомлениях в веб-приложении писательского клуба.

На рисунке 26 изображена таблица users, которая используется для хранения информации о пользователях. Поля таблицы включают: id – идентификатор пользователя, username – имя пользователя в системе, phone –

номер телефона пользователя, password – хэш пароля пользователя, about – информация «о себе», которую пользователя сам задает, role_id – идентификатор роли пользователя (на таблицу roles, которая представляет собой три роли пользователя: «admin», «moderator» и «user»), telegram_id – идентификатор из мессенджера Telegram, avatar – путь к аватару пользователя.

id	username	phone	password	created_at	about	role_id	telegram_id	avatar
1	Adminika	89647367238	\$2b\$10\$uq42l	2025-06-05 14:18:48	А можно я больше не буду притворяться что у меня ...	1	NULL	/uploads/avata 17491043623t 778567793.pn

Рисунок 26 – Таблица users

Таблица posts, изображенная на рисунке 27, используется для хранения информации о текстах пользователя. Ее поля: id – идентификатор текста, user_id – внешний ключ на таблицу пользователей для указания авторства, task_id – если текст был выполнен по заданию, то здесь прописывается внешний ключ на таблицу заданий, title – название текста, content – содержание текста, cover_image – обложка, которую пользователь задал тексту, views – количество просмотров, rating – рейтинг текста, вычисленный из комментариев, status – статус текста (черновик или опубликованный).

id	user_id	task_id	title	content	cover_image	views	rating	updated_at	created_at	status
1	2	1	Последнее предложен	<p>Лаборатория была тихой, только мониторы мерцали...	/uploads/covers/cover 1749111388952- 768913401.jpg	1	0.00	2025-06-05 16:16:29	2025-06-05 16:16:29	published

Рисунок 27 – Таблица posts

Таблица comments изображена на рисунке 28 и используется для хранения комментариев пользователей. Ее основные поля: post_id – текст, к которому относится комментарий, user_id – пользователь, который оставил комментарий, liked – что понравилось пользователю, disliked – что не понравилось пользователю,

overall_impression – общее впечатление пользователя, rating – оценка, которую поставил пользователь (от 1 до 5).

id	post_id	user_id	liked	disliked	overall_impression	rating	created_at
1	15	1	Лаконично и атмосферно. История в духе хорро...	Жаль, что коротко... Может быть, однажды узнаю поб...	Жутковато и оригинально, как creepypasta про ошибок...	5	2025-06-05 17:32:16

Рисунок 28 – Таблица comments

Таблица events изображена на рисунке 29. Ее основные поля: title – название мероприятия, description – описание мероприятия, start_date – дата начала мероприятия, end_date – дата окончания мероприятия, image_path – путь к изображению мероприятия.

id	title	description	start_date	end_date	image_path
1	Неделя микрофикш	Краткость – сестра таланта! Будем писать микрорасс...	2025-05-25	2025-05-30	/uploads/events/17cf5bd90ff47222e64

Рисунок 29 – Таблица events

Таблица tasks изображена на рисунке 30. Ее основное назначение: хранить данные о заданиях к ивентам. Ее поля: event_id – мероприятие, к которому относится задание, task_number – номер задания, short_description – краткое описание задания, instructions – инструкции к выполнению, reward – награда за выполнение, icon_path – иконка задания, release_date – дата выхода задания.

id	event_id	task_number	title	short_description	instructions	icon_path	created_at	release_date
1	1	1	Последнее предложение	Вам предстоит написать рассказ, который заканчивае...	1. Напишите микрорассказ (до 300 слов) 2. Последн...	/uploads/event-tasks/17491063icons8-B%D%4D%Dм...	2025-05-25 14:52:54	2025-05-25

Рисунок 30 – Таблица tasks

4.3 Разработка веб-приложения

Разработка серверной части веб-приложения на Node.js начинается с организации подключения к базе данных. Для этого используется файл db.js, в котором создается пул подключений к базе данных MySQL с использованием библиотеки mysql2/promise, поддерживающей асинхронную работу через промисы. На рисунке 31 приведен фрагмент кода, реализующий это подключение.

```
const mysql = require('mysql2/promise');

const dbConfig = {
  host: 'localhost',
  user: 'root',
  password: '',
  database: 'starsclub',
  waitForConnections: true,
  connectionLimit: 10,
  queueLimit: 0
};

const pool = mysql.createPool(dbConfig);

module.exports = pool;
```

Рисунок 31 – Класс db.js

Созданное ранее подключение к базе данных через модуль db.js используется в различных частях серверной логики системы для выполнения операций с данными. На рисунке 32 изображен пример инициализации соединения с базой данных.

```
const db = require('../config/db');
```

Рисунок 32 – Инициализация соединения с базой данных

Также важной частью работы с базой данных является возможность добавления, редактирования и удаления текстов пользователей. Для таких

операций в проекте реализован отдельный модуль – postController.js, а также соответствующие маршруты и обработчики запросов. На рисунках 33, 34, 35 изображены функции создания, редактирования и удаления текстов.

```

1  static async createPost(req, res) {
2    const { title, content, status, coverImage, taskId } = req.body;
3    const userId = req.userId;
4
5    let connection;
6    try {
7      connection = await db.getConnection();
8      await connection.beginTransaction();
9
10   const [postResult] = await connection.query(
11     'INSERT INTO posts (user_id, title, content, status, cover_image, task_id) VALUES (?, ?, ?, ?, ?, ?)',
12     [
13       userId,
14       title,
15       content,
16       status || 'draft',
17       coverImage || null,
18       taskId || null
19     ]
20   );
21
22   await connection.commit();
23   res.status(201).json({
24     message: status === 'published'
25       ? 'Публикация опубликована'
26       : 'Публикация сохранена как черновик',
27     postId: postResult.insertId
28   });
29 } catch (error) {
30   if (connection) await connection.rollback();
31   console.error('Ошибка при создании публикации:', error);
32   res.status(500).json({ error: 'Ошибка сервера' });
33 } finally {
34   if (connection) connection.release();
35 }
36 }

```

Рисунок 33 – Функция createPost из PostController.js

```

1  static async updatePost(req, res) {
2    const postId = req.params.id;
3    const { title, content, status, coverImage, taskId } = req.body;
4    const userId = req.userId;
5
6    try {
7      const [post] = await db.query(
8        'SELECT user_id FROM posts WHERE id = ?',
9        [postId]
10      );
11
12      if (!post.length) {
13        return res.status(404).json({ error: 'Публикация не найдена' });
14      }
15
16      if (post[0].user_id !== userId) {
17        return res.status(403).json({ error: 'Нет прав на редактирование' });
18      }
19
20      await db.query(
21        'UPDATE posts SET title = ?, content = ?, status = ?, cover_image = ?, task_id = ? WHERE id = ?',
22        [title, content, status, coverImage || null, taskId || null, postId] // Добавлено taskId
23      );
24
25      res.json({
26        message: 'Публикация обновлена',
27        postId: postId
28      });
29 } catch (error) {
30   console.error('Ошибка при обновлении публикации:', error);
31   res.status(500).json({ error: 'Ошибка сервера' });
32 }
33 }

```

Рисунок 34 – Функция updatePost из PostController.js



```
1  static async deletePost(req, res) {
2      const postId = req.params.id;
3      const userId = req.userId;
4
5      try {
6          const [post] = await db.query(
7              'SELECT user_id FROM posts WHERE id = ?',
8              [postId]
9          );
10
11         if (!post.length) {
12             return res.status(404).json({ error: 'Публикация не найдена' });
13         }
14
15         if (post[0].user_id !== userId) {
16             return res.status(403).json({ error: 'Нет прав на удаление' });
17         }
18
19         await db.query(
20             'DELETE FROM posts WHERE id = ?',
21             [postId]
22         );
23
24         res.json({ message: 'Публикация удалена' });
25     } catch (error) {
26         console.error('Ошибка при удалении публикации:', error);
27         res.status(500).json({ error: 'Ошибка сервера' });
28     }
29 }
```

Рисунок 35 – Функция deletePost из PostController.js

5 Тестирование веб-приложения

Одним из важных этапов разработки программного продукта является тестирование и отладка.

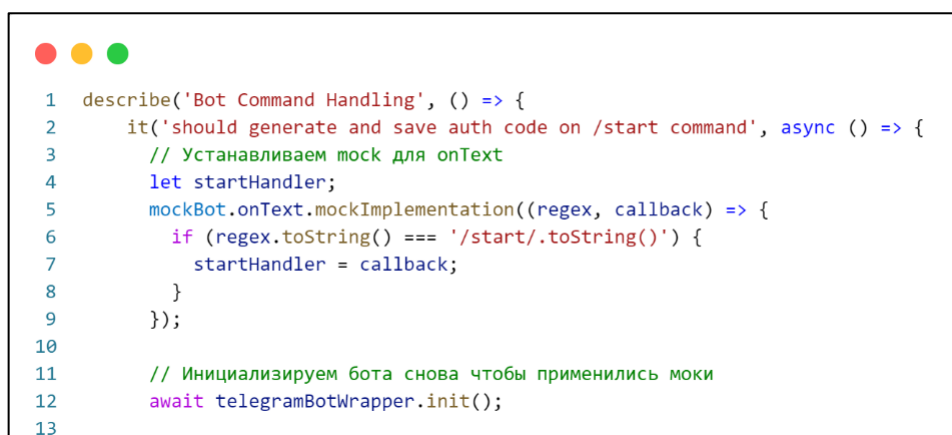
Тестирование – это процесс проверки работоспособности, надежности и безопасности веб-приложения.

Отладка – это выявление и устранение ошибок в ходе разработки путем пошагового запуска отдельных компонентов, анализа логов и консольного вывода.

Для тестирования backend-части, реализованной на Node.js с использованием Express.js, применялся фреймворк Jest – один из самых популярных инструментов для модульного и интеграционного тестирования в JavaScript-среде.

Цель модульного тестирования – убедиться, что каждый отдельный модуль работает корректно и не вызывает ошибок при взаимодействии с другими модулями.

На рисунках 36 и 37 представлен фрагмент кода файла telegramAuthController.test.js, в котором тестируется регистрация и авторизация через Telegram API. Этот фрагмент кода представляет проверку на работоспособность кода при инициализации команды «/start».



```
1 describe('Bot Command Handling', () => {
2   it('should generate and save auth code on /start command', async () => {
3     // Устанавливаем mock для onText
4     let startHandler;
5     mockBot.onText.mockImplementation((regex, callback) => {
6       if (regex.toString() === '/start/.toString()') {
7         startHandler = callback;
8       }
9     });
10
11     // Инициализируем бота снова чтобы применились моки
12     await telegramBotWrapper.init();
13   });
14 })
```

Рисунок 36 – Фрагмент кода теста проверки работоспособности бота (часть 1)

```

14 // Симулируем команду /start
15 const msg = {
16   chat: {
17     id: testChatId
18   },
19   text: '/start'
20 };
21
22 // Вызываем обработчик
23 await startHandler(msg);
24
25 // Проверяем что sendMessage был вызван
26 expect(mockBot.sendMessage).toHaveBeenCalledWith(
27   testChatId,
28   expect.stringContaining('Ваш код авторизации'),
29   { parse_mode: 'HTML' }
30 );
31
32 // Проверяем что код сохранился в БД
33 const [codes] = await connection.query(
34   'SELECT * FROM telegram_auth_codes WHERE chat_id = ?',
35   [testChatId]
36 );
37
38 expect(codes.length).toBe(1);
39 expect(codes[0].code).toHaveLength(6);
40 expect(codes[0].is_used).toBe(0);
41 });
42 });

```

Рисунок 37 – Фрагмент кода теста проверки работоспособности бота (часть 2)

На рисунках 38 и 39 представлен фрагмент кода для проверки регистрации через Telegram API.

```

1 describe('Code Verification', () => {
2   it('should reject invalid code', async () => {
3     const response = await request(app)
4       .get('/api/verify-tg-code?code=INVALID')
5       .expect(404);
6
7     expect(response.body.error).toBe('Неверный код');
8   });
9
10  it('should register new user with valid code', async () => {
11    // Insert test code
12    await connection.query(
13      'INSERT INTO telegram_auth_codes (chat_id, code, expires_at) VALUES (?, ?, DATE_ADD(NOW(), INTERVAL 5 MINUTE))',
14      [testChatId, testCode]
15    );
16
17    // Mock Telegram API response
18    axios.get.mockResolvedValue({
19      data: {
20        result: {
21          username: testUsername,
22          bio: 'Test bio'
23        }
24      }
25    });
26
27    // Mock bot sendMessage for success
28    axios.post.mockResolvedValue({ data: { ok: true } });
29

```

Рисунок 38 – Тестирование регистрации через Telegram API (часть 1)

```

30     const response = await request(app)
31       .get(`/api/verify-tg-code?code=${testCode}`)
32       .expect(200);
33
34     expect(response.body.success).toBe(true);
35     expect(response.body.token).toBeDefined();
36     expect(response.body.user.username).toBe(testUsername);
37
38     // Check user was created
39     const [users] = await connection.query(
40       'SELECT * FROM users WHERE telegram_id = ?',
41       [testChatId]
42     );
43
44     expect(users.length).toBe(1);
45     expect(users[0].username).toBe(testUsername);
46     expect(users[0].about).toBe('Test bio');
47
48     // Check code was marked as used
49     const [codes] = await connection.query(
50       'SELECT is_used FROM telegram_auth_codes WHERE code = ?',
51       [testCode]
52     );
53
54     expect(codes[0].is_used).toBe(1);
55   });
56
57   it('should login existing user with valid code', async () => {
58     // Create test user
59     await connection.query(
60       'INSERT INTO users (telegram_id, username, role_id) VALUES (?, ?, ?)',
61       [testChatId, testUsername, 3]
62     );
63
64     // Insert test code
65     await connection.query(
66       'INSERT INTO telegram_auth_codes (chat_id, code, expires_at) VALUES (?, ?, DATE_ADD(NOW(), INTERVAL 5 MINUTE))',
67       [testChatId, testCode]
68     );
69
70     // Mock Telegram API response
71     axios.get.mockResolvedValue({
72       data: {
73         result: {
74           username: testUsername,
75           bio: 'Updated bio'
76         }
77       }
78     });

```

Рисунок 39 – Тестирование регистрации через Telegram API (часть 2)

Результат тестирования представлен на рисунке 40.

```

Test Suites: 1 passed, 1 total
Tests:      6 passed, 6 total
Snapshots:  0 total
Time:       4.105 s
Ran all test suites.

```

Рисунок 40 – Результаты telegramAuthController.test.js

Для тестирования веб-приложения был разработан сценарий тестирования для роли. В таблице 15 представлен сценарий для роли администратора.

					ДП.09.02.07-3.25.212.02. ПЗ	Лист
						40
Изм.	Лист	№ докум.	Подпись	Дата		

Таблица 9 – Сценарий тестирования

Поле	Описание
Дата теста	30.04.2025
Приоритет тестирования	Средний
Заголовок/название теста	Добавление заданий к мероприятию
Этапы теста	1. Вход в веб-приложение для роли организатор 2. Выбор мероприятия 3. Добавление задания
Тестовые данные	Данные организатора для входа в приложение: логин и пароль.
Ожидаемый результат	Веб-приложение обновляет данные по мероприятию и показывает список всех заданий для мероприятия.
Фактический результат	Веб-приложение добавило задание к мероприятию.

Кроме того, в рамках тестирования создан чек-лист для роли автора, представленный в таблице 16.

Таблица 10 – Чек лист

Тест	Входные данные	Ожидаемый результат	Фактический результат	Результат тестирования	Комментарий
Регистрация	Данные пользователя: Имя, номер телефона, пароль.	Данные пользователя добавлены в базу данных	Данные пользователя добавлены в базу данных	Успешно	-
Авторизация	Данные пользователя: Номер телефона, пароль	Пользователь успешно вошел в приложение	Пользователь успешно вошел в приложение	Успешно	-
Добавление текста	Данные пользователя: данные авторизации (идентификатор пользователя). Данные поста (заголовок, текст)	Пользователь успешно добавил текст Текст отображается в личном кабинете и в результатах поиска	Пользователь успешно добавил текст Текст отображается в личном кабинете и в результатах поиска	Успешно	-

Тестирование проводилось в несколько этапов. На первом этапе проверялись основные функции приложения, такие как аутентификация пользователей. Далее проводилось тестирование прав доступа и ролей, чтобы убедиться, что только авторизованные пользователи могут выполнять операции с творческими работами. В процессе тестирования были выявлены и исправлены ошибки, после чего веб-приложение было готово к запуску.

					ДП.09.02.07-3.25.212.02. ПЗ	Лист
						42
Изм.	Лист	№ докум.	Подпись	Дата		

6 Стоимость разработки веб-приложения

Разработка веб-приложения для писательского клуба – это комплексный и поэтапный процесс, включающий как технические, так и организационные аспекты. В условиях цифровизации культурной и творческой сферы создание онлайн-платформы для объединения авторов и литературных энтузиастов открывает новые возможности для развития сообщества: удобная коммуникация, публикация творческих работ, организация мероприятий и конкурсов. Такой ресурс способствует не только популяризации литературного творчества, но и созданию комфортной среды для самореализации участников. Однако для успешного запуска проекта необходимо провести экономическое обоснование, рассчитать затраты, оценить трудоемкость реализации и спрогнозировать эффективность использования платформы.

6.1 Расчет трудоемкости разработки программного обеспечения

Трудоемкость является одним из ключевых показателей экономической эффективности проекта. Она отражает объем трудовых затрат, необходимых для производства единицы продукции или оказания услуги, и позволяет оценить рациональность использования рабочего времени. Данный показатель особенно важен при планировании и контроле ресурсов на всех этапах разработки веб-приложения.

Суммарные затраты труда рассчитываются как сумма составных затрат труда по формуле (1):

$$\sum T = t_{\text{оп}} + t_{\text{аи}} + t_{\text{пр}} + t_{\text{тз}} + t_{\text{разб}} + t_{\text{т}} + t_{\text{д}} \quad (1)$$

Таблица 11 – Суммарные затраты времени

Вид работ	Часы (всего)	Машинное время
Анализ предметной области проекта и описание структуры	15	15

Продолжение таблицы 11

Вид работ	Часы (всего)	Машинное время
Обзор инструментов	25	20
Проектирование проекта и описание этапов проектирования	35	30
Создание технического задания на разработку проекта	25	20
Разработка программного продукта	100	100
Тестирование программного продукта	35	35
Подготовка документации	45	45
Итого	280	265

$$\sum T = 15 + 25 + 35 + 25 + 100 + 35 + 45 = 280$$

Таким образом, общая трудоемкость проекта составляет 240 часов, из которых:

265 часов отнесены к машинному времени (разработка, тестирование и аналитика),

15 часов – к ручной работе (творческие и организационные этапы: анализ, проектирование и документация.)

6.2 Затраты на оплату

Затраты на оплату (ЗОТ) труда разработчика программного обеспечения включают оплату труда и отчисления от фонда заработной платы.

Затраты на заработную плату определяются произведение часовой тарифной ставки разработчика и трудоемкости разработки программного продукта по следующей формуле:

Основная заработная плата рассчитывается в рублях по формуле (2)

$$З_{\text{осн}} = \sum t * TC_{\text{мес}} \quad (2)$$

Где $\sum t$ – суммарные затраты труда, вычисляемые по формуле (1), час;

					ДП.09.02.07-3.25.212.02. ПЗ	Лист
						44
Изм.	Лист	№ докум.	Подпись	Дата		

$ТС_{МЕС}$ – часовая тарифная ставка, руб.

В расчеты зарплаты учитывается модальная заработная плата по Иркутску – 77 700 руб. Ставка за час – 471 руб. (при условии среднего количества рабочих часов при 40-часовой недели – 165 ч.).

$$З_{осн} = 240 * 471 = 113\,040 \text{ руб.}$$

Налог на физический лиц (НДФЛ-13%) – удержание производится из доходов, начисленных в пользу сотрудника.

Итоговый расчет заработной платы для разработчика ПО рассчитывается по формуле (3):

$$З_{зп} = З_{осн} - \left(\frac{З_{осн}}{100} * ЕСН \right) \quad (3)$$

$$З_{зп} = 113\,040 - 14\,695,20 = 98\,344,80 \text{ руб.}$$

6.3 Затраты на амортизацию оборудования

Амортизация – это процесс периодического переноса начальной стоимости основного средства или нематериального актива на производственные, коммерческие или общехозяйственные расходы – в зависимости от того, как этот актив используется.

Срок полезного использования согласно классификатору основных средств равен 3-5 года.

Для разработки использовался ноутбук изначальной стоимостью 82 299 рублей.

Расчет амортизации производится при помощи данных формул (4-7):

$$A_n = \frac{100\%}{K_k} = \frac{100\%}{5} = 20\% \quad (4)$$

где:

					ДП.09.02.07-3.25.212.02. ПЗ	Лист
						45
Изм.	Лист	№ докум.	Подпись	Дата		

Ан – годовая норма амортизации;

Кк – срок полезного использования в соответствии с классификатором.

$$A_{\Gamma} = C_0 * A_n = 83\,299 * 20\% = 16\,659,8 \quad (5)$$

где:

Аг – ежегодная сумма амортизации;

Со – начальная стоимость оборудования.

$$A_m = \frac{A_{\Gamma}}{12 \text{ мес}}, = \frac{16\,659,8}{12} = 1\,388,3 \quad (6)$$

где:

Ам – ежемесячная сумма амортизации.

Далее необходимо сумму ежемесячной амортизации умножить на количество месяцев эксплуатации оборудования.

$$Z_{\text{амор}} = A_m * m = 1\,388,3 * 1,66 = 2\,304,6 \quad (7)$$

Из таблицы 1 указано, что машинное время работы на компьютере 265 часов что равно 1,66 месяца.

Стоимость амортизации ноутбука равна 2 304,6 рублей.

6.4 Расчет затрат на электроэнергию

Для расчета затрат на электроэнергию первоочередное необходимо рассчитать расход электроэнергии оборудования по следующей формуле (8):

$$E = P * t \quad (8)$$

Где:

Р – электрическая мощность в киловаттах;

t – время в часах.

					ДП.09.02.07-3.25.212.02. ПЗ	Лист
						46
Изм.	Лист	№ докум.	Подпись	Дата		

К числу устройств, потребляющих электроэнергию, в ходе выполнения работы относится ноутбук Maibenben X556 с блоком питания. Суммарная потребляемая энергия для данного устройства составляет 0,385 кВт·ч.

Согласно таблице 1, машинное время работы ноутбука равно 225 часов. Стоимость одного киловатт-часа равна 1,58 рубля для городского населения Иркутской области. Общее потребление равно 86,625 кВт·ч.

Таким образом, затраты на электроэнергию составят:

$$C_9 = E * Ц = 86,625 * 1,58 = 136,87 \quad (9)$$

6.5 Затраты на материалы, израсходованные при проведении разработки и прочие

Затраты на материалы складывается из суммы стоимости материалов, которые сводятся в таблице 2.

Таблица 12 – Затраты на материалы

Наименование	Цена за единицу (руб.)	Кол-во (шт.)	Всего (руб.)
Бумага (500 листов)	500	1	500
Сшивание диплома	800	1	800
Итого			1300

6.6 Расчет затрат на разработку программного обеспечения

Затраты на разработку программного обеспечения включают в себя:

- 1) затраты на зарплату разработчика ($Z_{зп}$);
- 2) затраты на амортизацию оборудования ($Z_{аморт}$);
- 3) затраты на эксплуатацию оборудования (электроэнергия) ($Z_{экспл}$);
- 4) затраты на материалы, израсходованные при проведении разработки (бумага, картридж, и т.п.) ($Z_{мат}$);
- 5) Прочие затраты (использование платного ПО и тд.).

Затраты на разработку рассчитываются путем суммы всех затрат по формуле 10.

$$З_{разр} = З_{зп} + З_{аморт} + З_{экспл} + З_{мат} + З_{пр} \quad (10)$$

Таблица 13 – Общие расходы

Статьи затрат	Индекс	Сумма, руб.	Удельный вес затрат, %
Заработная плата	Зп.	98 344,80	96,33
Амортизация оборудования	Заморт.	2 304,6	2,26
Затраты на электроэнергию	Зэкспл.	136,87	0,13
Затраты на материалы	Змат.	1300	1,27
Прочие затраты	Зпр.	0	0
Итого:	Зразр.	102 086,27	100

6.7 Расчет цены

Определим расчетную цену при предполагаемом (плановом) размере прибыли на уровне 9%.

$$Пр = 102\,086,27 \times 9\% = 9\,187,76 \text{ руб.}$$

$$НДС = 0,2 \times (102\,086,27 + 9\,187,76) = 22\,254,8 \text{ руб.}$$

Итоговая цена:

$$Ц = 102\,086,27 + 9\,187,76 + 22\,254,8 = 133\,528,83 \text{ руб.}$$

Итоговая стоимость программного продукта составляет 133 528,83 рублей.

7 Руководство пользователя веб-приложения

Для запуска веб-приложения необходимо в терминал среды разработки backend прописать «`npx nodemon server.js`», а в терминал среды разработки frontend «`npm run serve`». После того, как запустится сервер разработки, веб-приложение будет доступно в веб-браузере по адресу «`http://localhost:8080`». Для того, чтобы попасть на домашнюю страницу веб-приложения, необходимо ввести этот адрес в строку веб-браузера.

Главная страница показана на рисунке 41.

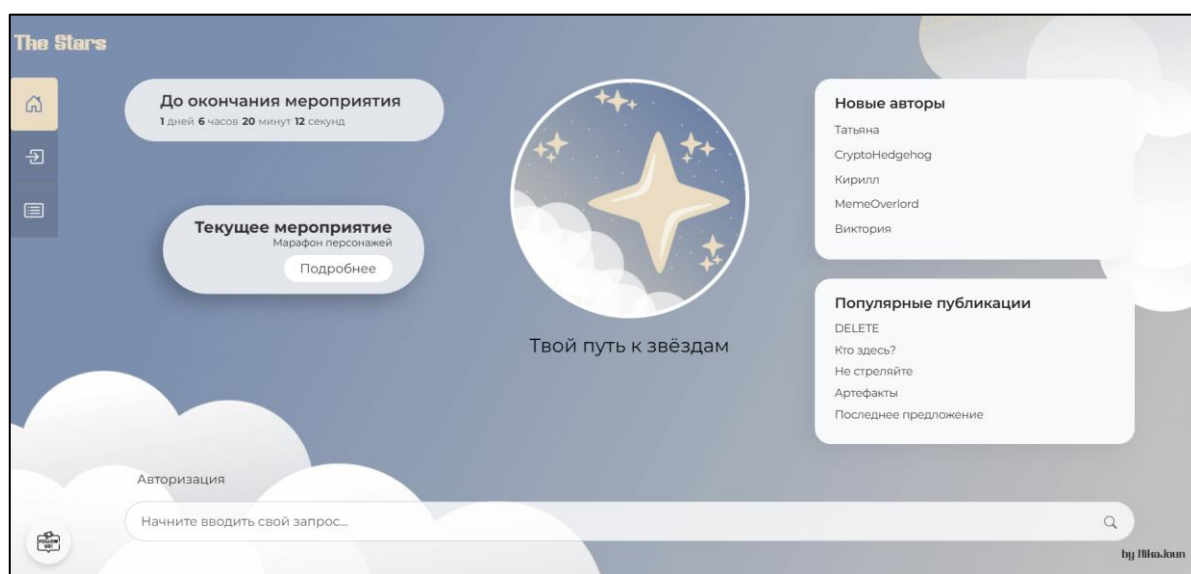


Рисунок 41 – Главная страница

Сценарий 1: Авторизация и Регистрация в веб-приложении

На главной странице пользователь может ознакомиться с новыми авторами, популярными текстами, творческими мероприятиями, а также вписать свой запрос в поисковую строку и найти текст. Здесь же есть кнопка авторизации, по которой пользователь попадает на страницу авторизации (рисунок 42, 43). При регистрации пользователь вводит имя, по которому его все будут узнавать, телефон и пароль. При авторизации только телефон и пароль.

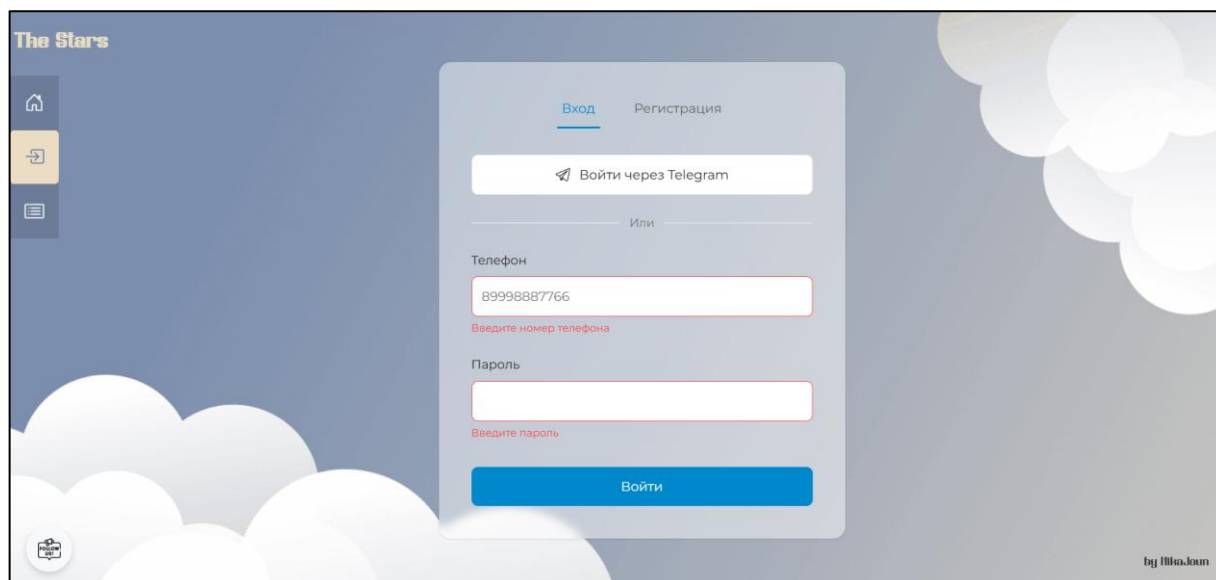


Рисунок 42 – Вход

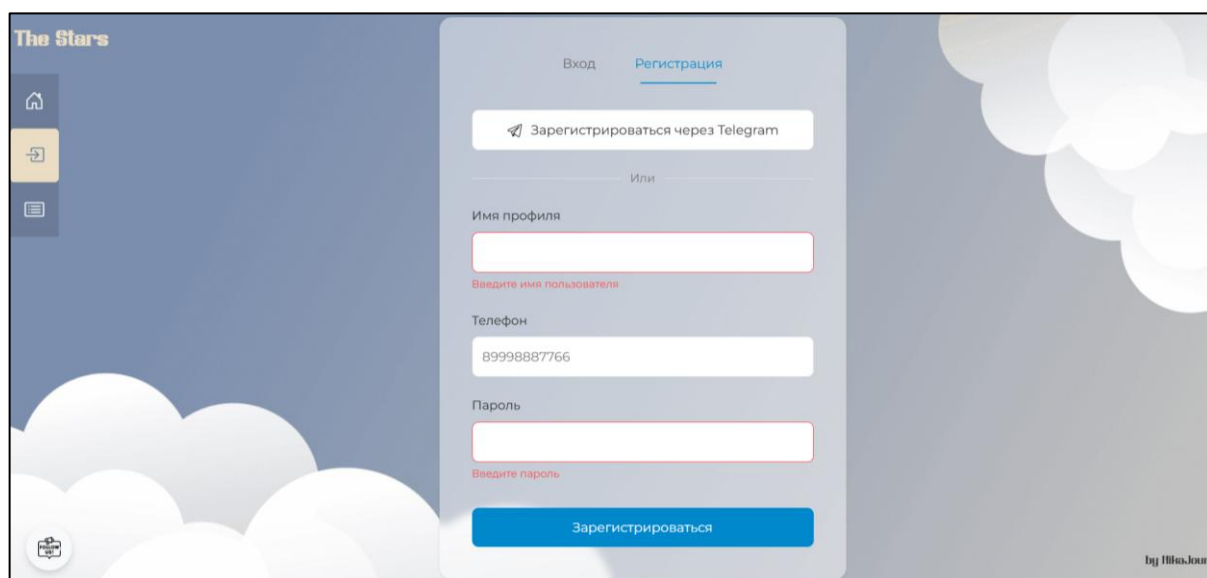


Рисунок 43 – Регистрация

При авторизации через Telegram API пользователь нажимает на кнопку «Войти/Зарегистрироваться через Telegram», перед ним открывается модальное окно со ссылкой на переход в Telegram (рисунок 44). Пользователь переходит по ссылке в чат со специальным ботом, берет у него код аутентификации и вводит его на странице авторизации (рисунок 45).

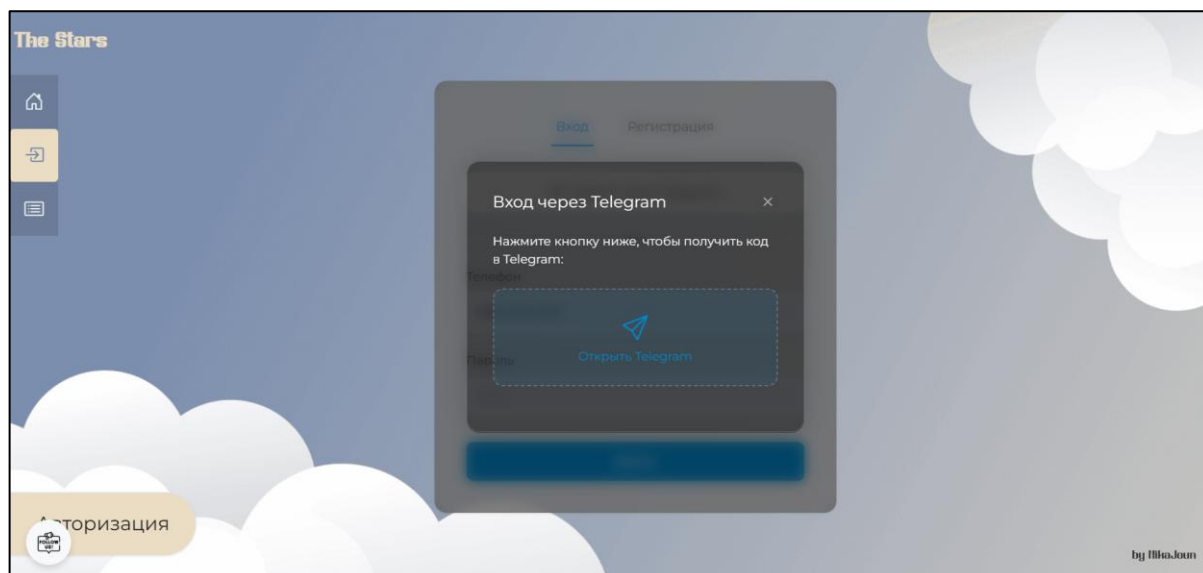


Рисунок 44 – Модальное окно Telegram API

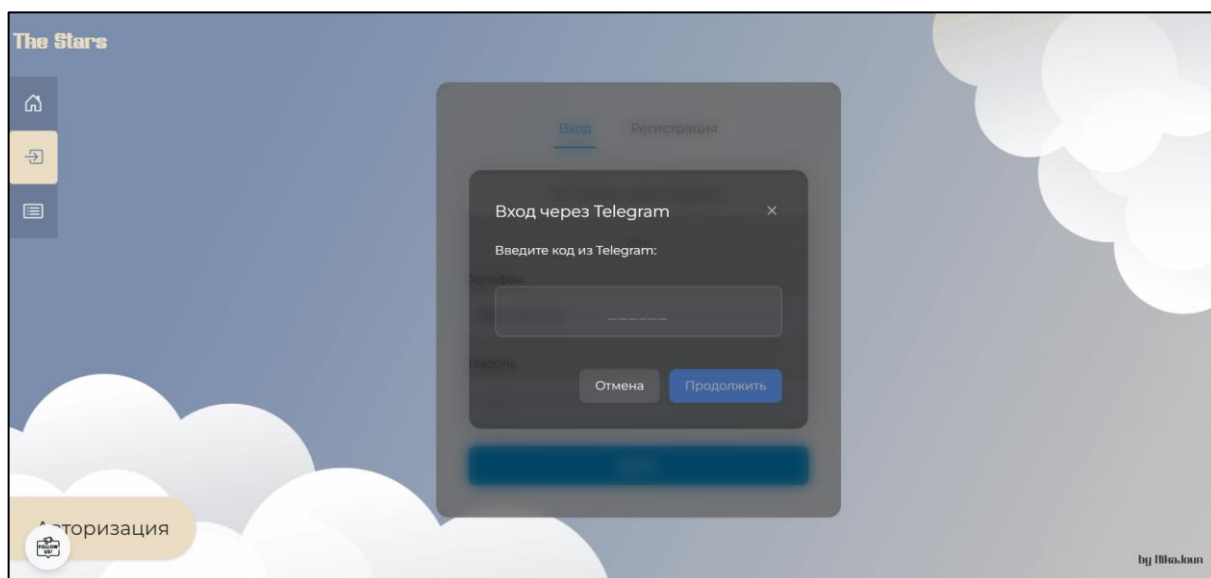


Рисунок 45 – Модальное окно ввода кода аутентификации

После успешной авторизации в веб-приложении пользователю становится доступны все функции веб-приложения.

Сценарий 2: Публикация творческих работ

Находясь в личном кабинете (рисунок 46) пользователь может нажать на кнопку «Написать текст» и сразу оказаться на странице Текстового редактора (рисунок 47).

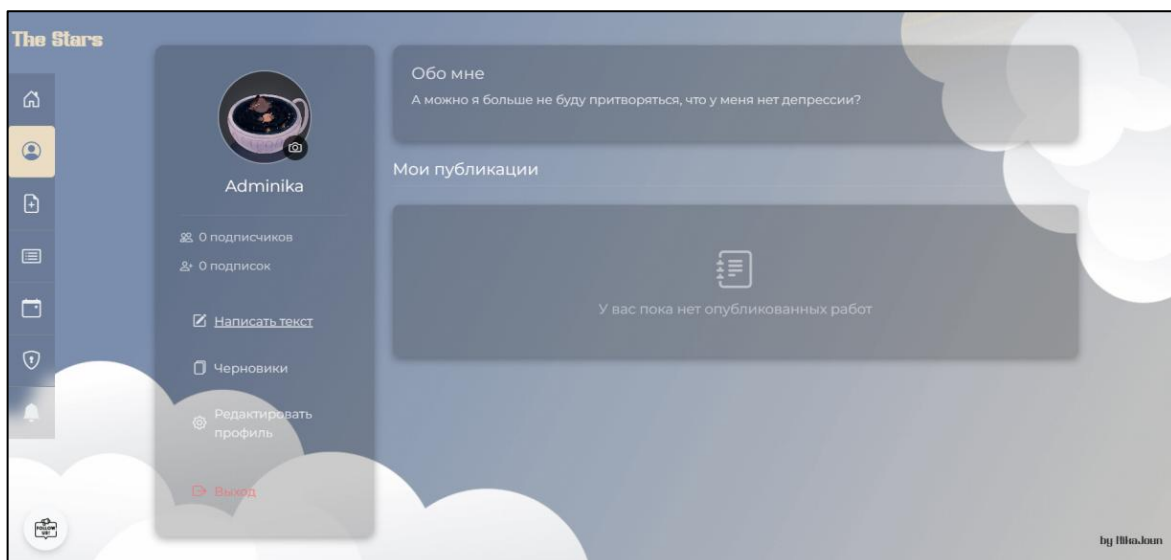


Рисунок 46 – Личный кабинет

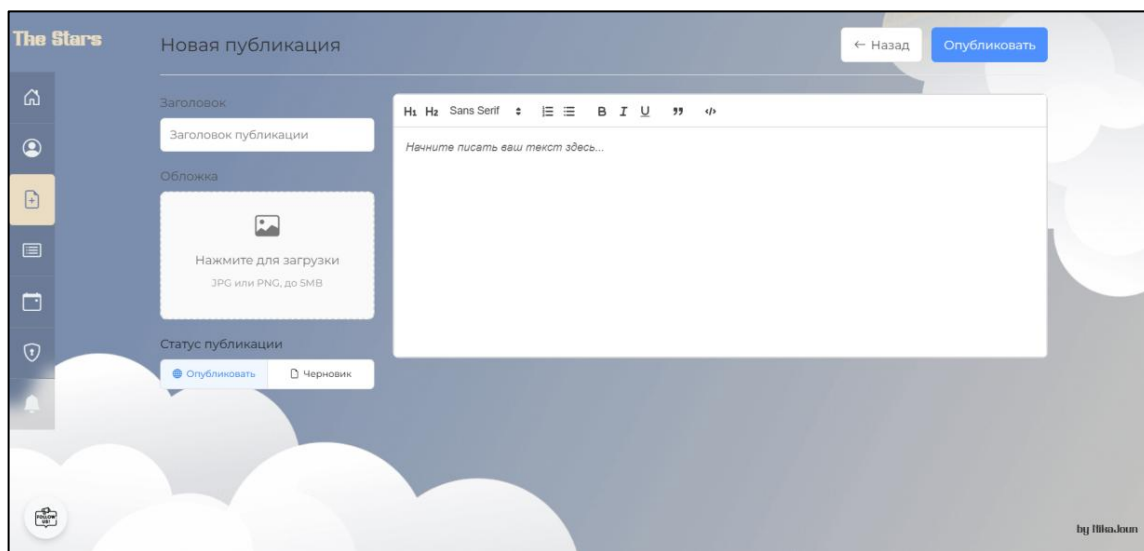


Рисунок 47 – Текстовый редактор

На странице текстового редактора пользователь заполняет заголовок текста, пишет текст, отмечает, хочет ли сохранить текст в черновики или опубликовать публично. Прикрепление обложки к тексту не является обязательным.

После написания текста, пользователь нажимает на кнопку «Опубликовать» и видит страницу своего текста (рисунок 48).

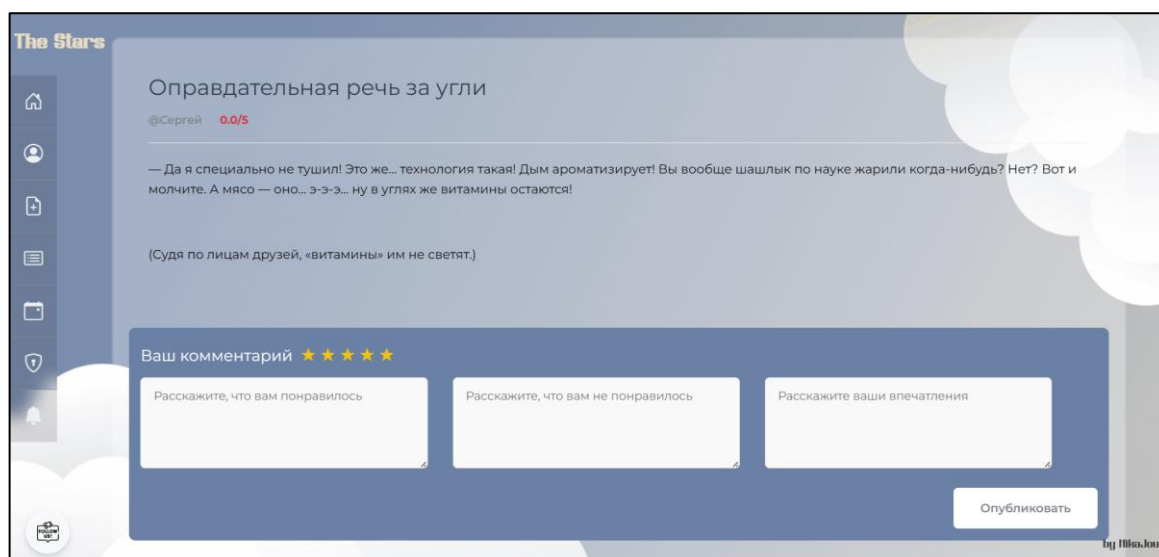


Рисунок 48 – Страница публикации

Сценарий 3: Участие в творческом конкурсе

Находясь на главной страничке, пользователь может увидеть таймер и анонс события. Нажав на кнопку «Подробнее», откроется модальное окно с полным превью мероприятия (рисунок 49).

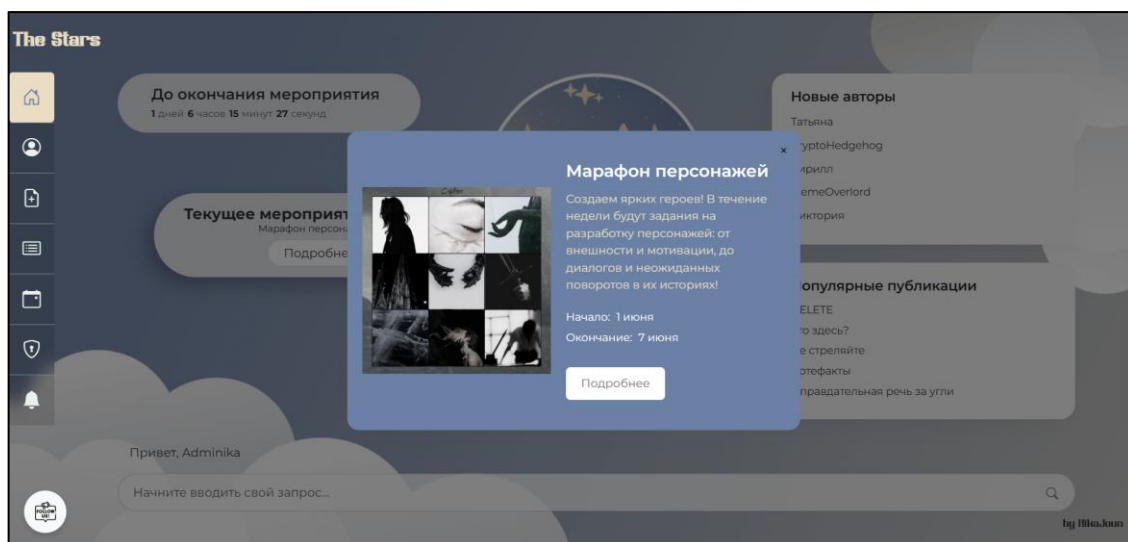


Рисунок 49 – Превью творческого конкурса

Пользователь нажимает на кнопку «Поучаствовать» и переходит на страницу мероприятия с заданиями по ним (рисунок 50).

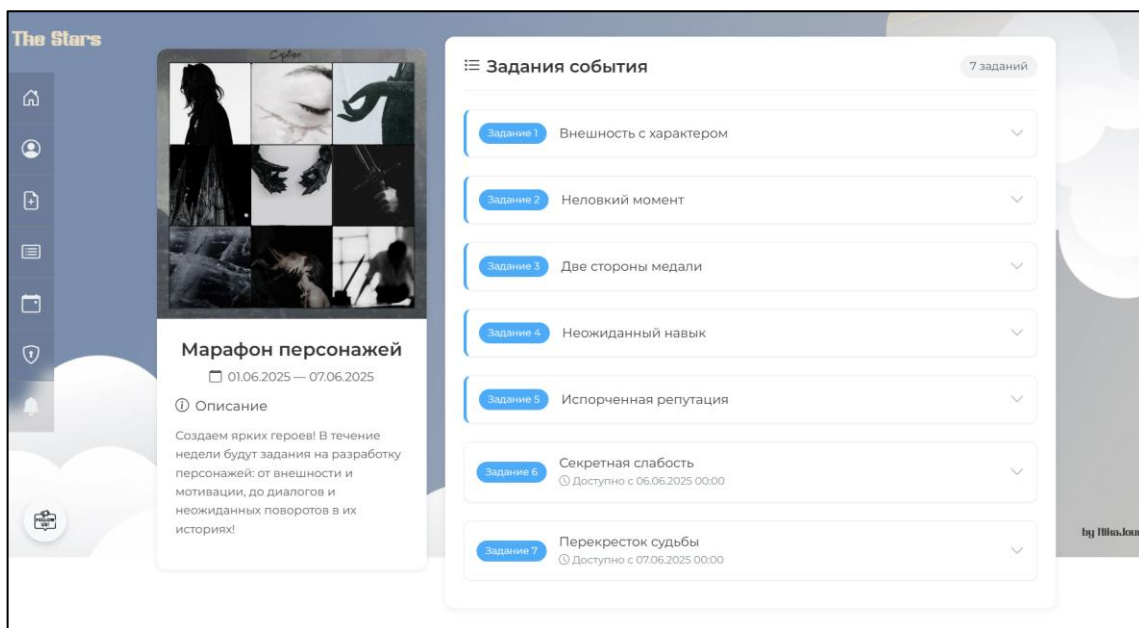


Рисунок 50 – Страница мероприятия

Выбрав задание себе по вкусу, пользователь нажимает на кнопку «Создать решение» и переходит на страницу текстового редактора, чтобы создать творческую работу на тематику конкурсного задания.

ЗАКЛЮЧЕНИЕ

В ходе выполнения дипломного проектирования была успешно достигнута основная цель — разработка веб-приложения «Писательский клуб». Для её достижения были выполнены все ключевые задачи, включающие предпроектное исследование, выбор инструментальных средств, составление технического задания, проектирование базы данных, разработку пользовательского интерфейса, тестирование системы и подготовку программной документации, включая руководство пользователя.

Созданное веб-приложение представляет собой удобную и функциональную платформу, обеспечивающую пользователям возможность публиковать и оценивать публикации и участвовать в литературных мероприятиях. Она обладает всеми необходимыми функциями для комфортного взаимодействия участников и объединения людей с общими интересами, предоставляя возможности для творческого самовыражения и развития литературного сообщества.

В результате проделанной работы веб-приложение «Писательский клуб» стало инструментом, который способен вдохновлять, объединять и поддерживать творческих людей, подтверждая важность и значимость подобных проектов в условиях цифровизации и стремительного роста популярности онлайн-сообществ.

					ДП.09.02.07-3.25.212.02. ПЗ	Лист
						55
Изм.	Лист	№ докум.	Подпись	Дата		

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

- 1 developer.mozilla.org – MDN Web Docs – URL: <https://developer.mozilla.org/> (дата обращения: 29.04.2025). – Текст: электронный.
- 2 expressjs.com – Express – Node.js web application framework – URL: <https://expressjs.com/> (дата обращения: 26.04.2025). – Текст: электронный.
- 3 fontawesome.com – Font Awesome: Icons and Toolkit – URL: <https://fontawesome.com/> (дата обращения: 27.04.2025). – Текст: электронный.
- 4 getbootstrap.com – Bootstrap · The most popular HTML, CSS, and JS library – URL: <https://getbootstrap.com/> (дата обращения: 01.05.2025). – Текст: электронный.
- 5 jsonwebtoken.io – JSON Web Tokens – URL: <https://jwt.io/> (дата обращения: 29.04.2025). – Текст: электронный.
- 6 lighthouse.dev – Lighthouse | Google Developers – URL: <https://developer.chrome.com/docs/lighthouse/> (дата обращения: 10.05.2025). – Текст: электронный.
- 7 mysql.com – MySQL :: World's Most Popular Open Source Database – URL: <https://www.mysql.com/> (дата обращения: 24.04.2025). – Текст: электронный.
- 8 nodejs.org – Node.js – URL: <https://nodejs.org/> (дата обращения: 25.04.2025). – Текст: электронный.
- 9 npmjs.com – npm: Node Package Manager – URL: <https://www.npmjs.com/> (дата обращения: 24.04.2025). – Текст: электронный.
- 10 router.vuejs.org – Vue Router Documentation – URL: <https://router.vuejs.org/> (дата обращения: 24.04.2025). – Текст: электронный.
- 11 sequelize.org – Sequelize: Node.js ORM for Postgres, MySQL, MariaDB, SQLite and Microsoft SQL Server – URL: <https://sequelize.org/> (дата обращения: 25.04.2025). – Текст: электронный.
- 12 vuejs.org – Vue.js: The Progressive JavaScript Framework – URL: <https://vuejs.org/> (дата обращения: 28.04.2025). – Текст: электронный.

					ДП.09.02.07-3.25.212.02. ПЗ	Лист
						56
Изм.	Лист	№ докум.	Подпись	Дата		

13 w3schools.com – W3Schools Online Web Tutorials – URL: <https://www.w3schools.com/> (дата обращения: 23.04.2025). – Текст: электронный.

14 web.dev – web.dev by Google: Guidance for modern web development – URL: <https://web.dev/> (дата обращения: 26.04.2025). – Текст: электронный.

15 webaim.org – WebAIM: Web Accessibility Guidelines – URL: <https://webaim.org/> (дата обращения: 26.04.2025). – Текст: электронный.

					ДП.09.02.07-3.25.212.02. ПЗ	Лист
						57
Изм.	Лист	№ докум.	Подпись	Дата		

Приложение А Техническое задание

Министерство образования Иркутской области

Государственное бюджетное профессиональное

образовательное учреждение Иркутской области

«Иркутский авиационный техникум»

(ГБПОУИО «ИАТ»)

Техническое задание

ВЕБ-ПРИЛОЕЖНИЕ «ПИСАТЕЛЬСКИЙ КЛУБ»

Руководитель: _____ (А.С. Александрова)
(подпись, дата)

Студент: _____ (В.А. Антонинова)
(подпись, дата)

Иркутск, 2025

					ДП.09.02.07-3.25.212.02. ПЗ	Лист
						58
Изм.	Лист	№ докум.	Подпись	Дата		

1 Введение

1.1 Общие сведения

Документ представляет собой техническое задание на создание веб-приложения «Писательский клуб», которое предназначено для поддержки и развития творчества начинающих авторов.

1.2 Цели и задачи

Целью создания веб-приложения является создание платформы для обмена творческими работами, получения обратной связи и оценки опубликованных текстов. Задачи веб-приложения включают:

- Управление творческими проектами и публикациями.
- Обеспечение возможности получения отзывов и комментариев.
- Генерация отчетов о деятельности клуба.

2 Основания для разработки

2.1 Нормативные документы

Документ основывается на следующих нормативных документах:

- ГОСТ 34.602-2020 "Информационная технология. Комплекс стандартов на автоматизированные системы. Техническое задание на создание автоматизированной системы".
- ГОСТ Р 56477-2015 "Проектирование и внедрение информационных систем. Общие требования".
- Приказ №351-у от 07.02.2025 "Об утверждении тем дипломных проектов и назначении руководителей и консультантов".

					ДП.09.02.07-3.25.212.02. ПЗ	Лист
						59
Изм.	Лист	№ докум.	Подпись	Дата		

– Методические указания по выполнению дипломного проекта для специальности 09.02.07 Информационные системы и программирование, квалификация: разработчик веб и мультимедийных приложений.

2.2 Проектные документы

Проектные документы включают:

- Пояснительную записку.
- Руководство пользователя.

3 Назначение системы

3.1 Общее описание

Веб-приложение «Писательский клуб» предназначено для использования начинающими авторами и любителями литературы. Приложение будет поддерживать процессы обмена творческими работами, а также предоставлять инструменты для получения обратной связи.

3.2 Преимущества и новизна

Веб-приложение будет предоставлять:

- Интуитивно понятный интерфейс для публикации и комментирования работ.
- Интеграцию с социальной сетью Telegram.

4 Требования к системе

4.1 Функциональные требования

- Роль гость:

					ДП.09.02.07-3.25.212.02. ПЗ	Лист
						60
Изм.	Лист	№ докум.	Подпись	Дата		

1) Возможность поиска по названию, фильтрации результатов поиска по дате («сегодня», «за 7 дней», «за 30 дней», «за год») всех публикаций.

2) Возможность просматривать профили зарегистрированных пользователей и их публикации.

– Роль пользователь:

1) Функциональность гостя.

2) Регистрация и авторизация в веб-приложении с помощью Telegram API или номера телефона и пароля.

3) Редактирование информации о себе в личном кабинете пользователя.

4) Загрузка аватара пользователя в форматах .png и .jpg, разрешенный размер: не более 5Мб.

5) Возможность поиска по названию, фильтрации результатов поиска по дате («сегодня», «за 7 дней», «за 30 дней», «за год») своих публикаций в личном кабинете.

6) Создание, редактирование и удаление пользователем своих собственных публикаций, сохранение их со статусом «Черновик» или «Опубликовано».

7) Загрузка обложки к публикации в форматах .png и .jpg, размером не более 5Мб.

8) Возможность комментирования публикаций другими пользователями.

9) Возможность подписываться на других пользователей.

10) Уведомления о новых подписках и новых комментариях к публикациям.

11) Возможность просматривать информацию о предстоящих и начавшихся мероприятиях и их заданиях.

12) Возможность загружать решения на вышедшие задания.

– Роль администратор:

1) Функциональность пользователя.

					ДП.09.02.07-3.25.212.02. ПЗ	Лист
						61
Изм.	Лист	№ докум.	Подпись	Дата		

2) Управление списком мероприятий: создание, редактирование, удаление и просмотр списка мероприятий с функцией поиска по названию, а также сортировкой по дате начала, дате окончания и названию.

3) Возможность загружать изображение к мероприятию и иконку к заданию в форматах .png и .jpg, размером не более 5Мб.

4) Управление списком заданий к мероприятиям: создание, редактирование, удаление и просмотр списка заданий с фильтрацией по мероприятиям.

5) Управление ролью пользователей: просмотр базовой информации о пользователе, назначение новых администраторов или снятие их с должности, фильтрация по ролям и поиск по имени.

6) Генерация статистических отчетов по участникам (общая статистика по количеству активных участников и их средний рейтинг за выбранный период, список из 10 самых активных участников за выбранный период) и по публикациям (общая статистика по просмотрам и среднему рейтингу среди всех публикаций за выбранный период, список из 10 самых популярный публикаций на платформе за выбранный период).

7) Возможность экспорта отчетов в форматах DOCX и PDF для дальнейшего анализа и печати.

4.2. Технические требования

– Производительность:

1) Обработка до 15 пользователей одновременно.

2) Время отклика системы не более 15 секунд при загрузке данных.

– Надежность:

1) Доступность системы не менее 99,5% в год.

– Безопасность:

1) Аутентификация пользователей через Telegram.

					ДП.09.02.07-3.25.212.02. ПЗ	Лист
						62
Изм.	Лист	№ докум.	Подпись	Дата		

- 2) Шифрование данных на уровне передачи и хранения.

4.3. Эксплуатационные требования

- Удобство использования:
 - 1) Дружественный пользовательский интерфейс.
- Интеграция:
 - 1) Возможность интеграции с социальными сетями для авторизации.

5. Требования к техническому обеспечению

5.1. Оборудование

- Сервер: Серверная платформа с процессором не менее 4 ядер, 8 ГБ ОЗУ, SSD объемом 512 ГБ.
- Клиентские рабочие станции: ПК с ОС Windows 10 или Linux, 4 ГБ ОЗУ, 2 ГБ свободного места на диске.

5.2. Сетевые требования

- Сеть: Доступ в Интернет со скоростью не менее 10 Мбит/с.
- Сетевые протоколы: Поддержка TCP/IP, HTTPS.

6. Требования к программному обеспечению

6.1. Программные компоненты

- Операционная система: Серверная версия ОС Linux (Ubuntu 20.04 LTS и выше) или Windows Server.
- Базы данных: MySQL версии не ниже 10.0.
- Веб-сервер: Nginx или Apache с настройкой проксирования запросов к Node.js.

					ДП.09.02.07-3.25.212.02. ПЗ	Лист
						63
Изм.	Лист	№ докум.	Подпись	Дата		

- Сервер приложений: Node.js версии 18.x или выше.

6.2. Интерфейсы

- Клиентский интерфейс: Веб-интерфейс, доступный через браузеры: Google Chrome, Opera GX, Yandex Browser.
- Программный интерфейс (API): REST API, реализованный на Express.js с поддержкой авторизации через JWT.
- Протоколы и порты: HTTP/HTTPS, порты: 80/443 (внешние), 3000 (внутренний Node.js-сервер)

7. Организационно-технические требования

7.1. Этапы разработки

В таблице 1 представлены сроки и этапы разработки приложения.

Таблица 1 – Сроки и этапы разработки приложения

№	Этап	Срок выполнения
1	Предпроектное исследование (выбор темы, постановка цели, задач, описание области применения, исследование предметной области)	22.02.2025
2	Разработка технического задания (выбор архитектуры программного обеспечения, выбор типа пользовательского интерфейса, выбор языка и среды программирования)	28.02.2025
3	Проектирование программного обеспечения (разработка структурной и функциональной схемы ПО, проектирование базы данных (инфологическое, ER-модель, физическая модель)	15.03.2025
4	Разработка (программирование) программного продукта	30.04.2025
5	Тестирование и отладка программного продукта (функциональное тестирование, составление тест-планов)	06.05.2025
6	Составление программной документации (оформление пояснительной записки, написание руководства пользователя, составление презентации и речи)	11.05.2025

Приложение Б Листинг главной страницы

HomeView.vue

```
<template>
<div class="home-container">
  <MainContentBlock />
  <AuthSection />
  <SearchSection />
</div>
</template>
```

```
<script setup>
import { onMounted } from 'vue'
import { useHomeStore } from '@stores/homeStore'
import MainContentBlock from '@components/home/MainContentBlock.vue'
import AuthSection from '@components/home/AuthSection.vue'
import SearchSection from '@components/home/SearchSection.vue'
```

```
const homeStore = useHomeStore()
```

```
onMounted(() => {
  homeStore.initData()
})
</script>
```

```
<style scoped>
.home-container {
  max-width: 1300px;
  margin: 0 auto;
  padding: 2rem 1rem;
}
</style>
```

MainContentBlock.vue

```
<template>
<div class="main-content">
  <LeftColumn />
  <CenterColumn />
```

					ДП.09.02.07-3.25.212.02. ПЗ	Лист
						65
Изм.	Лист	№ докум.	Подпись	Дата		

```
<RightColumn />
```

```
</div>
```

```
</template>
```

```
<script setup>
```

```
import LeftColumn from '@components/home/LeftColumn.vue'
```

```
import CenterColumn from '@components/home/CenterColumn.vue'
```

```
import RightColumn from '@components/home/RightColumn.vue'
```

```
</script>
```

```
<style scoped>
```

```
.main-content {
```

```
display: flex; justify-content: space-between; align-items: flex-start; margin-top: 4rem;
```

```
margin-bottom: 2rem; gap: 2rem; flex-wrap: wrap;
```

```
}
```

```
@media (max-width: 992px) {
```

```
.main-content {
```

```
flex-direction: column; align-items: center;
```

```
}
```

```
}
```

```
</style>
```

```
AuthSection.vue
```

```
<template>
```

```
<div class="auth-section">
```

```
<span v-if="!user" class="auth-text" @click="$router.push('/login')">Авторизация
```

```
</span>
```

```
<span v-else class="auth-text" @click="$router.push('/profile')">
```

```
Привет, {{ user.username }}</span>
```

```
</div>
```

```
</template>
```

```
<script setup>
```

```
import { storeToRefs } from 'pinia'
```

```
import { useHomeStore } from '@stores/homeStore'
```

					ДП.09.02.07-3.25.212.02. ПЗ	Лист
						66
Изм.	Лист	№ докум.	Подпись	Дата		

```
const homeStore = useHomeStore()
const { user } = storeToRefs(homeStore)
</script>
```

```
<style scoped>
.auth-section {
margin-top: 2rem; margin-left: 2rem; text-align: left;
}
```

```
.auth-text {
font-size: 1rem; font-weight: 500; cursor: pointer; color: #333;
}
</style>
```

SearchSection.vue

```
<template>
<div class="search-section">
<div class="search-input-container">
<input type="text" v-model="searchQuery" class="form-control search-input"
placeholder="Начните вводить свой запрос..." @keyup.enter="searchPosts"/>
<button class="search-button" @click="searchPosts">
<svg xmlns="http://www.w3.org/2000/svg" width="16" height="16" fill="currentColor"
viewBox="0 0 16 16">
<path
d="M11.742 10.344a6.5 6.5 0 1 0-1.397 1.398h-.001c.03.04.062.078.098.115l3.85
3.85a1 1 0 0 0 1.415-1.414l-3.85-3.85a1.007 1.007 0 0 0-.115-.115zM12 6.5a5.5 5.5 0 1 1-
11 0 5.5 5.5 0 0 1 11 0z" />
</svg>
</button>
</div>
</div>
</template>
```

```
<script setup>
import { ref } from 'vue'
import { useRouter } from 'vue-router'
```

```
const router = useRouter()
const searchQuery = ref("")
```

```
function searchPosts() {
  if (searchQuery.value.trim()) {
    router.push(`/posts?query=${searchQuery.value}`)
  }
}
</script>
```

```
<style scoped>
```

```
.search-input-container {
  position: relative;
}
```

```
.search-input {
  width: 100%; padding: 12px 50px 12px 20px; border-radius: 30px; border: 1px solid
  #ced4da; font-size: 1rem; background: rgba(255, 255, 255, 0.8);
}
```

```
.search-input:focus {
  background: #fff; outline: none;
}
```

```
.search-button {
  position: absolute; right: 16px; top: 50%; transform: translateY(-50%); background:
  none; border: none; color: #666; cursor: pointer;
}
</style>
```

					ДП.09.02.07-3.25.212.02. ПЗ	Лист
						68
Изм.	Лист	№ докум.	Подпись	Дата		