

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**

**«ДОНСКОЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

**(ДГТУ)**

Факультет «Информатика и вычислительная техника»

КАФЕДРА «Кибербезопасность информационных систем»

**КУРСОВАЯ РАБОТА**

Тема: «Разработка WEB-приложения для информационного сопровождения задачи выполняемой группой работников»

Дисциплина: «WEB-программирование»

Направление подготовки: 09.03.01 Информатика и вычислительная техника

Направленность: Информатика и вычислительная техника

Обозначение курсового проекта WП.850000.000 Группа ВИ-41

Обучающийся \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Бойкова А.П.

(подпись, дата)

Курсовая работа защищена с оценкой \_\_\_\_\_\_\_\_\_\_\_\_\_\_

Руководитель работы \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ доцент, И.А. Сосновский

(подпись, дата)

Ростов-на-Дону

2024



МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**

**«ДОНСКОЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

**(ДГТУ)**

Факультет «Информатика и вычислительная техника»

КАФЕДРА «Кибербезопасность информационных систем»

**ЗАДАНИЕ**

на выполнение курсовой работы

Тема «Разработка WEB-приложения для информационного сопровождения задачи выполняемой группой работников»

Дисциплина: WEB-программирование

Обучающийся: Бойкова Арина Павловна

Обозначение курсовой работы WП.850000.000 Группа: ВИ-41

Срок представления проекта (работы) к защите «\_\_\_» \_\_\_\_\_\_\_\_ 20\_\_г.

Исходные данные для курсового проекта:

1. Учебные материалы, электронные ресурсы, периодические материалы.

2. Официальный сайт организации, для которой разрабатывается web-приложение, сайты аналогичных компаний.

3. Рекомендации по выполнению работы.

4. Рекомендации по оформлению работы.

**Содержание пояснительной записки**

|  |
| --- |
| Введение. |
| 1. Теоретические основы разработки сайтов. |
| 2. Сбор информации об объекте. |
| 3. Создание базы данных. |
| 3. Создание карты сайта или макета сайта. |
| 4. Создание шаблон дизайна страницы. |
| 5. Формирование контента. |
| 6. Вёрстка сайта. |
| 7. Тестирование и размещение сайта в сети. |
| Заключение. |
| Перечень использованных информационных ресурсов. |
| Приложение А. |

Руководитель проекта \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ И.А. Сосновский

(подпись, дата)

Задание принял к исполнению \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ А.П. Бойкова

(подпись, дата)

Содержание

|  |  |
| --- | --- |
| Введение……………………………………………………………………. | 4 |
| 1. Теоретические основы разработки сайтов…………………………….. | 5 |
| 1.1 Язык гипертекстовой разметки HTML………………………… | 5 |
| 1.2 Язык программирования JavaScript……………………………. | 6 |
| 1.3 Язык программирования PHP…………………………………... | 7 |
| 1.4 Сервер баз данных MySQL……………………………………...  1.5 Среда разработки VS Code ……………………………………... | 9  10 |
| 2. Сбор информации об объекте…………………………………………... | 11 |
| 3. Создание базы данных………………………………………………….. | 14 |
| 4. Создание карты сайта или макета сайта………………………………. | 17 |
| 5. Создание шаблон дизайна страницы………………………………….. | 20 |
| 6. Формирование контента………………………………………………… | 22 |
| 7. Вёрстка сайта……………………………………………………………. | 24 |
| 8. Тестирование и размещение сайта в сети …...………………………… | 29 |
| Заключение…………………………………………………………………. | 35 |
| Перечень использованных информационных ресурсов…………………. | 36 |
|  |  |

**Введение**

В современном информационном мире, где удаленная работа и совместное взаимодействие становятся все более распространенными, возникает необходимость в эффективных инструментах для организации коллективной работы и обмена идеями. В этом контексте WEB-приложение для информационного сопровождения задачи выполняемой группой работников, играет ключевую роль, предоставляя пользователю возможность создавать различные страницы, на которых можно совместно работать, обмениваться идеями и делиться ресурсами. Позволяя легко добавлять текст, изображения, видео и другие мультимедийные элементы WEB-приложение становится неотъемлемым инструментом для организации рабочего процесса.

В данном исследовании будут рассмотрены функциональные возможности WEB-приложения для информационного сопровождения задачи выполняемой группой работников и его потенциал в организации эффективного сотрудничества и обмена информацией в различных областях.

**1 Теоретические основы разработки сайтов**

* 1. **Язык гипертекстовой разметки HTML**

HTML (HyperText Markup Language) — это стандартизированный язык разметки, используемый для создания веб-страниц. HTML предоставляет структуру и семантику для содержимого веб-страницы, определяя, как содержимое должно быть организовано и отображено в браузере.

HTML состоит из серии элементов (тегов), которые определяют различные части содержимого страницы. Каждый элемент имеет свою собственную функцию и может содержать текст, изображения, ссылки, формы и другие элементы. Элементы HTML организованы в древовидную структуру, называемую DOM (Document Object Model), которая определяет иерархию элементов на странице.

Рассмотрим возможность применения данного языка:

* структурирование содержимого: HTML позволяет организовать содержимое веб-страницы с помощью различных элементов, таких как заголовки, абзацы, списки и таблицы, делая страницу более удобной для чтения и понимания;
* семантика: HTML предоставляет семантические элементы, которые помогают поисковым системам и ассистентам понимать структуру и содержание веб-страницы, что улучшает SEO и доступность;
* интерактивность: HTML поддерживает создание интерактивных элементов, таких как формы и кнопки, позволяя пользователям взаимодействовать с веб-страницей;
* визуальное форматирование: HTML используется для определения внешнего вида содержимого страницы с помощью CSS (Cascading Style Sheets) и добавления интерактивности с помощью JavaScript.

Вместе с тем, HTML является лишь одной из составляющих технологического стека веб-разработки, и взаимодействует с CSS и JavaScript для обеспечения визуального оформления и интерактивности веб-страниц.

**1.2 Язык программирования JavaScript**

JavaScript (JS) — это высокоуровневый, интерпретируемый язык программирования, который широко применяется для создания динамических веб-страниц. С течением времени JavaScript стал популярным языком программирования, который сейчас используется в различных областях, включая веб-разработку, разработку мобильных приложений, создание игр и многое другое. Основные особенности JavaScript:

* синтаксис и типы данных: JavaScript имеет синтаксис, подобный другим языкам программирования, таким как C++ и Java, что делает его относительно простым для изучения. Он поддерживает различные типы данных, включая числа, строки, массивы, объекты и булевы значения;
* динамическая типизация: В JavaScript не требуется явное указание типов данных переменных, так как он использует динамическую типизацию. Это означает, что тип данных переменной определяется автоматически во время выполнения программы;
* функции: JavaScript поддерживает функции как объекты первого класса, что означает, что функции могут быть присвоены переменным, переданы в качестве аргументов другим функциям и возвращены из других функций;
* объектно-ориентированное программирование (ООП): JavaScript поддерживает объектно-ориентированное программирование, позволяя создавать объекты, определять их свойства и методы, а также наследовать свойства и методы от других объектов;
* событийное программирование: JavaScript используется для обработки событий, таких как клики мыши, нажатия клавиш и загрузки страницы. С помощью обработчиков событий можно создавать интерактивные элементы на веб-страницах;
* асинхронное программирование: JavaScript поддерживает асинхронное выполнение кода, что позволяет выполнять операции ввода-вывода (например, запросы к серверу) без блокировки исполнения кода;
* интеграция с HTML и CSS: JavaScript используется для взаимодействия с HTML и CSS, позволяя изменять содержимое и стили веб-страницы динамически в зависимости от действий пользователя или других событий.

JavaScript является ключевым инструментом веб-разработки и играет важную роль в создании современных интерактивных и динамических веб-приложений. Он широко применяется как в браузере, так и на сервере.

**1.3 Язык программирования PHP**

PHP (Hypertext Preprocessor) — это широко используемый язык программирования, который применяется преимущественно для создания динамических веб-сайтов и веб-приложений. Он является одним из наиболее популярных языков программирования на вебе и широко применяется для разработки сайтов различного уровня сложности, от небольших блогов до крупных интернет-порталов и онлайн-магазинов. Основные особенности PHP:

* динамическая генерация контента, одним из ключевых преимуществ PHP является его способность генерировать динамический контент на веб-страницах. Это позволяет создавать интерактивные элементы, обрабатывать формы, работать с базами данных и создавать персонализированные страницы для пользователей;
* взаимодействие с базами данных, PHP обладает инструментами для работы с различными системами управления базами данных (СУБД), такими как MySQL, PostgreSQL, SQLite и другими. Он позволяет выполнять запросы к базам данных, получать, изменять и удалять данные, что делает его идеальным инструментом для создания сайтов с динамическим содержимым;
* универсальность и гибкость, PHP может быть использован для широкого спектра задач, начиная от создания скриптов обратной связи и заканчивая разработкой сложных веб-приложений;
* обширные библиотеки и фреймворки. Существует множество библиотек и фреймворков для PHP, которые упрощают разработку веб-приложений и предоставляют готовые решения для типичных задач, таких как аутентификация пользователей, маршрутизация URL-адресов, обработка форм и другие;
* PHP является кроссплатформенным языком программирования и может быть запущен на различных операционных системах, включая Windows, Linux, macOS и другие;
* большое сообщество и обширная документация, PHP имеет большое сообщество разработчиков и обширную документацию, что облегчает изучение языка и получение поддержки при разработке проектов;
* безопасность в PHP может быть слабым местом при неправильном использовании, существует множество методов и рекомендаций для обеспечения безопасности веб-приложений на этом языке, таких как защита от SQL-инъекций, кросс-сайтового скриптинга и других угроз.

PHP продолжает оставаться одним из наиболее популярных и востребованных языков программирования для веб-разработки благодаря своей мощности, гибкости и широкому применению.

**1.4 Сервер баз данных MySQL**

MySQL — это одна из самых популярных открытых реляционных систем управления базами данных (СУБД), широко используемая для хранения и управления данными во многих веб-приложениях. Основные особенности MySQL:

* MySQL предназначен для хранения и организации данных в структурированном формате. Он обеспечивает механизмы для создания баз данных, таблиц и индексов для эффективного хранения и обработки информации;
* MySQL использует язык структурированных запросов (SQL) для взаимодействия с базой данных. С помощью SQL можно выполнять различные операции с данными, такие как вставка, выборка, обновление и удаление;
* MySQL предоставляет механизмы управления доступом к данным и базе данных. Администраторы могут настраивать права доступа пользователей к конкретным таблицам или операциям;
* MySQL обеспечивает механизмы безопасности для защиты данных от несанкционированного доступа. Это включает в себя установку паролей, шифрование данных и другие меры защиты;
* MySQL обладает хорошей масштабируемостью и может работать как на небольших, так и на крупных проектах. Он поддерживает горизонтальное и вертикальное масштабирование для увеличения производительности и обработки больших объемов данных.

Работа с MySQL через PHPMyAdmin:

* PHPMyAdmin предоставляет удобный веб-интерфейс для управления базами данных и таблицами MySQL. С его помощью можно создавать, изменять и удалять базы данных и таблицы, а также просматривать и редактировать структуру таблиц;
* PHPMyAdmin позволяет выполнять SQL-запросы прямо из веб-интерфейса. Это позволяет администраторам и разработчикам выполнять различные операции с данными и управлять базой данных;
* PHPMyAdmin поддерживает импорт и экспорт данных из и в базу данных MySQL из различных форматов файлов, таких как CSV, SQL и другие;
* PHPMyAdmin позволяет администраторам управлять пользователями и их привилегиями доступа к базам данных и таблицам. Это включает в себя создание новых пользователей, установку паролей и назначение различных уровней доступа.

В целом, MySQL предоставляет надежную и эффективную платформу для хранения и управления данными, а PHPMyAdmin облегчает администрирование баз данных MySQL через удобный веб-интерфейс. С их помощью разработчики и администраторы могут эффективно управлять базами данных и обеспечивать их безопасность и производительность.

**1.5 Среда разработки Visual Studio Code**

Visual Studio Code (VS Code) — это легкий и мощный текстовый редактор с открытым исходным кодом, разработанный Microsoft. Он широко используется разработчиками для создания различных типов приложений, включая веб-сайты, веб-приложения, мобильные приложения, а также приложения для облачных и настольных платформ. Вот основные особенности, которые следует учитывать при работе с Visual Studio Code:

* Visual Studio Code поддерживает большое количество языков программирования, включая JavaScript, TypeScript, Python, Java, C++, C#, PHP, HTML, CSS и многие другие. Он обеспечивает подсветку синтаксиса, форматирование кода и другие функции для работы с различными языками;
* VS Code имеет огромное сообщество разработчиков, которые создают расширения функциональности редактора. Расширения могут добавлять новые функции, интегрировать сторонние сервисы, оптимизировать рабочий процесс и многое другое;
* VS Code интегрируется с популярными системами контроля версий, такими как Git, что позволяет разработчикам легко управлять и отслеживать изменения в их проектах. Он предоставляет интерфейс для выполнения различных операций Git, таких как коммиты, слияния, ветвление и др., прямо из редактора;
* VS Code предоставляет встроенные инструменты для отладки кода, что упрощает процесс поиска и устранения ошибок. Он поддерживает отладку для различных языков программирования с использованием различных отладчиков и настроек. VS Code интегрируется с облачными сервисами, такими как Azure, AWS, Google Cloud и другие, что позволяет разработчикам легко разрабатывать, развертывать и управлять облачными приложениями;
* В VS Code встроен терминал, который позволяет выполнять команды внутри редактора без необходимости переключаться на внешний терминал. Терминал можно расширять с помощью различных интеграций, таких как PowerShell, Command Prompt, Bash и другие;
* VS Code предоставляет широкие возможности настройки, позволяя пользователю адаптировать редактор под свои нужды. Различные параметры, настройки тем оформления, расширения и другие инструменты помогают создать комфортное и продуктивное рабочее окружение;
* VS Code интегрируется с различными инструментами разработки.

**2 Сбор информации об объекте**

Платформа для информационного сопровождения задачи выполняемой группой работников может быть очень полезной и интересной для создания интерактивных онлайн-досок, где пользователи могут делиться идеями, заметками, изображениями и другим мультимедийным контентом. Такое WEB – приложение нужно для:

* коллаборация и обмен идеями. Приложение может быть направлено на облегчение совместной работы и обмена идеями между пользователями. Это может включать возможность создания виртуальных досок, на которых пользователи могут добавлять заметки, изображения, видео и другие мультимедийные элементы, а также комментировать их;
* учебные цели. Приложение может быть использовано в образовательных целях для создания интерактивных уроков, заданий и проектов, где учащиеся могут совместно работать, обмениваться материалами и обсуждать темы;
* организация событий и проектов. Подобное приложение может быть полезно для организации событий, проектов или встреч, где участники могут делиться идеями, планировать действия и обсуждать детали;
* управление процессом принятия решений. Приложение может помочь группе людей в принятии решений, путем совместного обсуждения вариантов, голосования и анализа результатов.

Важные детали в реализации WEB – приложения:

* приложение будет реализовано для администратора, руководителя и пользователей;
* регистрация и аутентификация пользователей, пользователи должны иметь возможность создавать учетные записи и входить в систему;
* создание досок, руководитель и администратор должны иметь возможность создавать доски или "панели" для различных проектов или тем;
* добавление контента: администратор и руководитель должны иметь возможность добавлять текст, изображения, и другие мультимедийные элементы на свои доски;
* комментарии и обсуждения, пользователи могут комментировать и обсуждать контент, добавленный на доску;
* управление доступом: администратор и руководитель могут управлять правами доступа к доскам, делиться ими с другими пользователями;
* должна быть обеспечена безопасность для данных пользователей и безопасность их учетных записей.

**3 Создание базы данных**

Для создания WEB-приложения, база данных играет важную роль, поскольку она служит для хранения всех данных, необходимых для функционирования приложения и обеспечения пользовательского опыта. Вот несколько причин, почему база данных необходима для такого приложения:

* хранение пользовательских данных. База данных позволяет хранить информацию о пользователях, включая их учетные записи, личную информацию и предпочтения;
* хранение данных досок. Для каждой созданной доски необходимо хранить информацию о её содержимом, такую как заметки, изображения, видео и комментарии. База данных позволяет структурированно хранить эту информацию и обеспечивать доступ к ней;
* управление правами доступа. База данных позволяет определять права доступа пользователей к различным функциям приложения и контенту. Например, определение администраторов досок или управление приватностью контента;
* отслеживание активности пользователей. База данных может использоваться для отслеживания активности пользователей, такой как создание, редактирование или удаление заметок, комментариев и досок, что позволяет улучшить аналитику и персонализировать пользовательский опыт;
* обеспечение целостности данных. Использование базы данных позволяет обеспечить целостность данных путем предотвращения потери или искажения информации.

Для создания базы данных для такого приложения необходимо определить её структуру и используемые таблицы, а также установить отношения между ними. Даталогическая модель для информационного сопровождения задачи выполняемой группой работников представлена на рисунке 1.

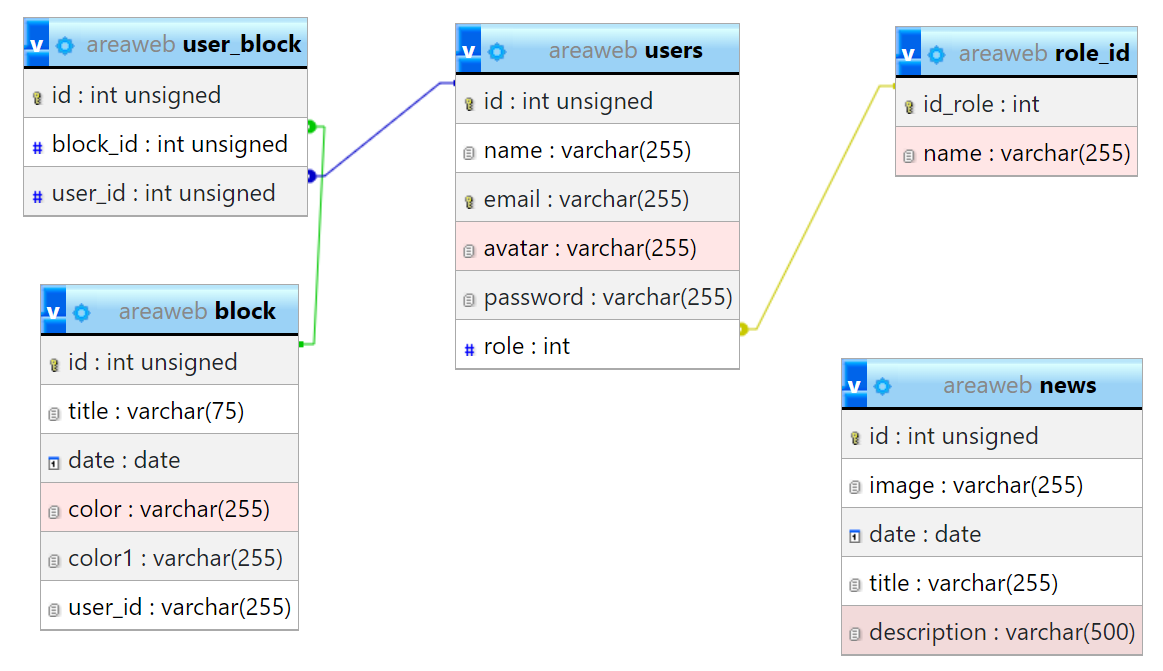


Рисунок 1 – Даталогическая модель

Рассмотрим каждую таблицу:

Таблица 1 - Пользователи

|  |  |  |
| --- | --- | --- |
| Столбец | Тип данных | Описание |
| id | int | Уникальный идентификатор пользователя |
| name | varchar (255) | Уникальный логин пользователя |
| email | varchar (255) | Уникальный email пользователя |
| avatar | varchar (255) | Фотография пользователя |
| password | varchar (255) | Хэшированный пароль пользователя |
| role | int | Роль пользователя |

Таблица 2 - Роль пользователя

|  |  |  |
| --- | --- | --- |
| Столбец | Тип данных | Описание |
| id\_role | int | Уникальный идентификатор роли |
| name | varchar (255) | Наименование роли (админ, пользователь) |

В таблице блоки содержится информация о созданных блоках доски, название, цвет блока, цвет текста, дата создания и распределение доступности для определенного пользователя, каждый из блоков будет отображаться на странице у администратора, руководителя и пользователя.

Таблица 3 - Блоки

|  |  |  |
| --- | --- | --- |
| Столбец | Тип данных | Описание |
| id | int | Уникальный идентификатор блока |
| title | varchar (255) | Тема блока |
| date | date | Дата создания блока |
| color | varchar (255) | Цвет блока |
| color1 | varchar (255) | Цвет текста блока |
| user\_id | int | Доступность блока пользователям |

В таблице пользовательские блоки, будет информация о том, какой блок с темой доступен определенному пользователю.

Таблица 4 - Блоки для пользователя

|  |  |  |
| --- | --- | --- |
| Столбец | Тип данных | Описание |
| id | int | Уникальный идентификатор |
| block\_id | int | Уникальный идентификатор блока |
| user\_id | int | Уникальный идентификатор пользователя |

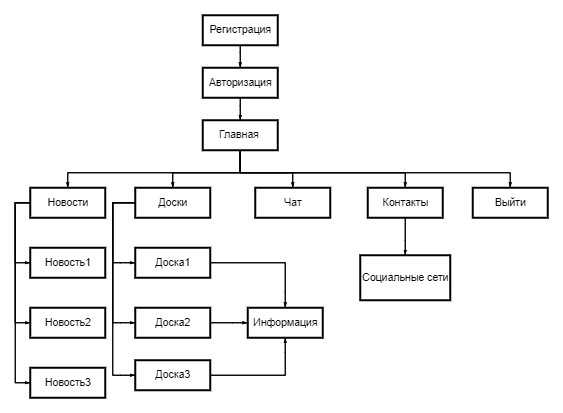
В таблице новости будет представлена информация о созданных новостях.

Таблица 5 - Новости

|  |  |  |
| --- | --- | --- |
| Столбец | Тип данных | Описание |
| id | int | Уникальный идентификатор новости |
| image | varchar (255) | Картинка новости |
| date | date | Дата создания новости |
| title | varchar (255) | Тема новости |
| description | varchar (255) | Описание новости |

**4 Создание карты сайта или макета сайта**

Карта сайта содержит список всех доступных на веб-сайте страниц или разделов, обычно представленных в иерархическом порядке. Карта сайта облегчает поиск информации по сайту для поисковых роботов и пользователей. Рассмотрим карту сайта на рисунке 2.

Рисунок 2 – Карта сайта

Основные разделы включают:

* главная страница: ссылки на основные разделы сайта, информационные блоки;
* блоки новостей, которые включают в себя новостную информацию;
* страница чата, содержит список пользователей и взаимодействие между ними;
* страница доски, которая включает в себя информацию о созданном блоке, содержит фотографии, документы, комментарии и реакции пользователей.

Каждый раздел сайта логически связан с другими, обеспечивая удобную навигацию и удобный интерфейс для пользователей. Рассмотрим макет страницы приложения на рисунке 3.

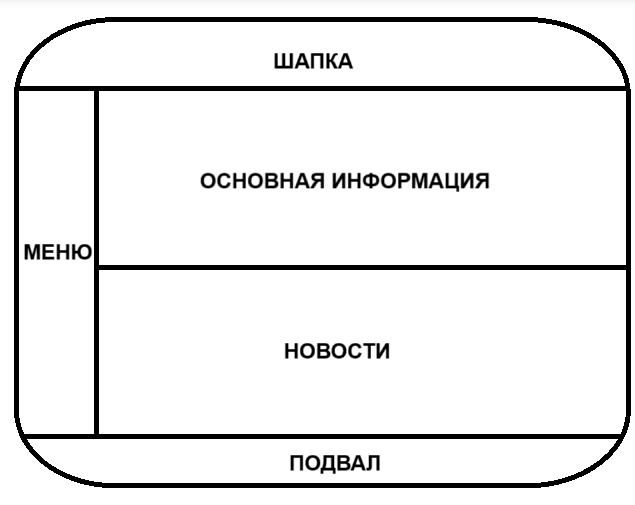


Рисунок 3 – Макет страницы

На данном макете представлена структура WEB-приложения, которая состоит из следующих разделов:

* шапка - верхняя часть страницы, обычно содержащая логотип, название сайта и основное меню навигации. Может также включать контактные данные, кнопки для входа в систему, поиск и другие элементы;
* навигационное меню - содержит ссылки на основные разделы сайта или страницы, облегчая пользователям навигацию по сайту. Обычно размещается в шапке страницы, но также может быть расположено в боковой панели или в подвале;
* основное содержимое - основная область страницы, содержащая основной контент, такой как текст, изображения, здесь размещается информация, которая предназначена для привлечения внимания пользователя и передачи основного сообщения страницы;
* подвал - нижняя часть страницы, содержащая дополнительные ссылки, контактную информацию и другие вспомогательные элементы. Обычно включает ссылки на разделы сайта, контактные данные, ссылки на социальные сети и другие;

Макет страницы определяет расположение и структуру контента на веб-сайте. Он позволяет разработчикам и дизайнерам организовать информацию на странице таким образом, чтобы пользователи могли легко ориентироваться и быстро находить нужную им информацию.

**5 Создание шаблон дизайна страницы**

Шаблон страницы определяет общий стиль и внешний вид веб-сайта. основные элементы, шаблона дизайна страницы:

* цветовая палитра WEB - приложения выполнена в белых, синих и голубых оттенках. Цвета должны гармонировать друг с другом, чтоб не портить внешний вид и не создавать диссонанс. Более яркие цвета будут использоваться для выделения важной информации на странице;
* для удобства восприятия информации рекомендуется использовать читаемые и современные шрифты;
* страницы будут иметь четкую и легко воспринимаемую структуру для удобной навигации пользователей. Ясное разделение различных элементов, таких как меню навигации, заголовки, контент и подвал, делает навигацию по сайту интуитивно понятной и удобной;
* использование высококачественных изображений, отражающих деятельность вашей организации или проектов, чтобы создать привлекательное визуальное впечатление.

Пример шаблона дизайна главной страницы сайта можно увидеть на рисунке 4 и рисунке 5.

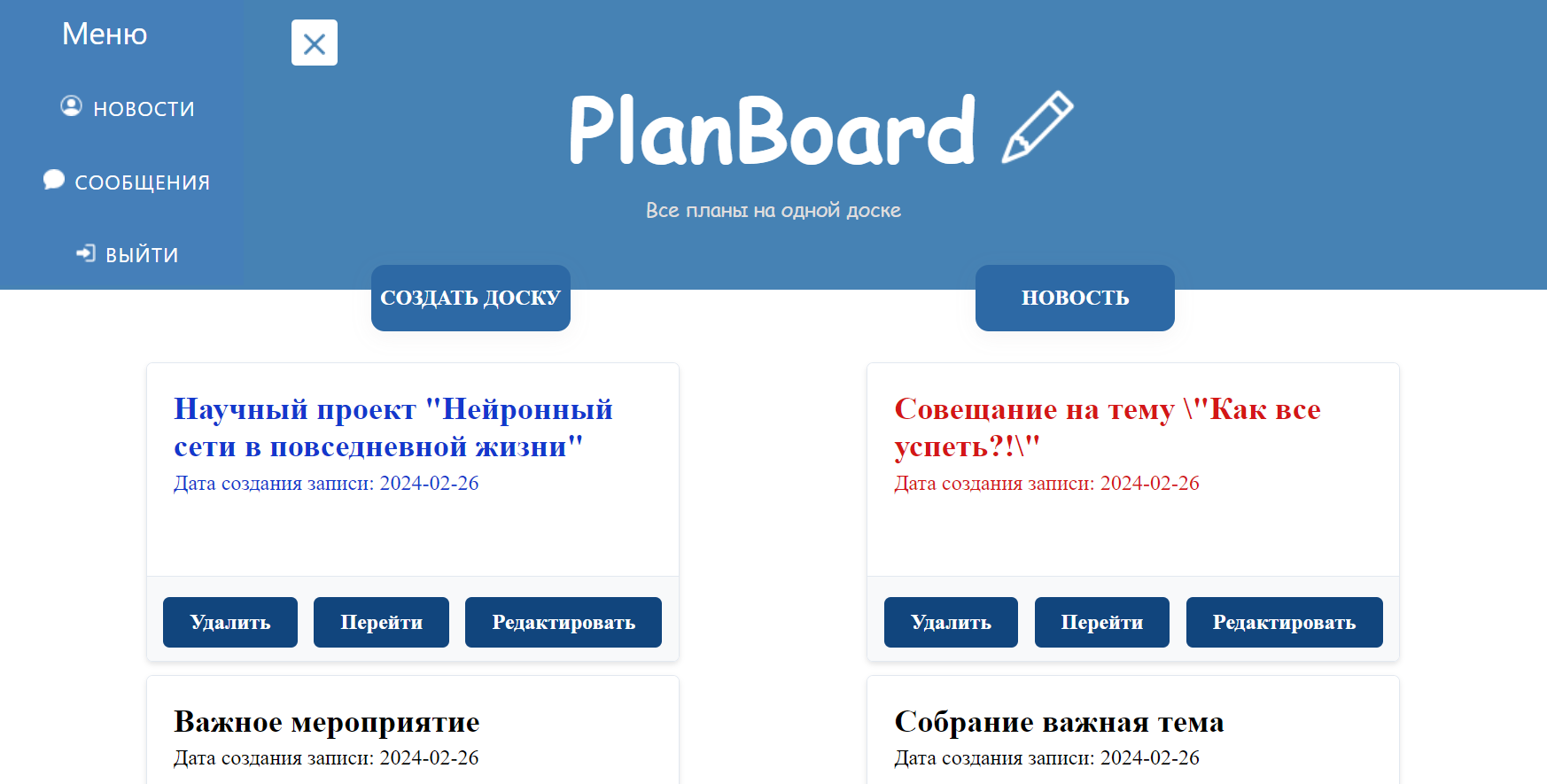


Рисунок 4 – Пример дизайна страницы

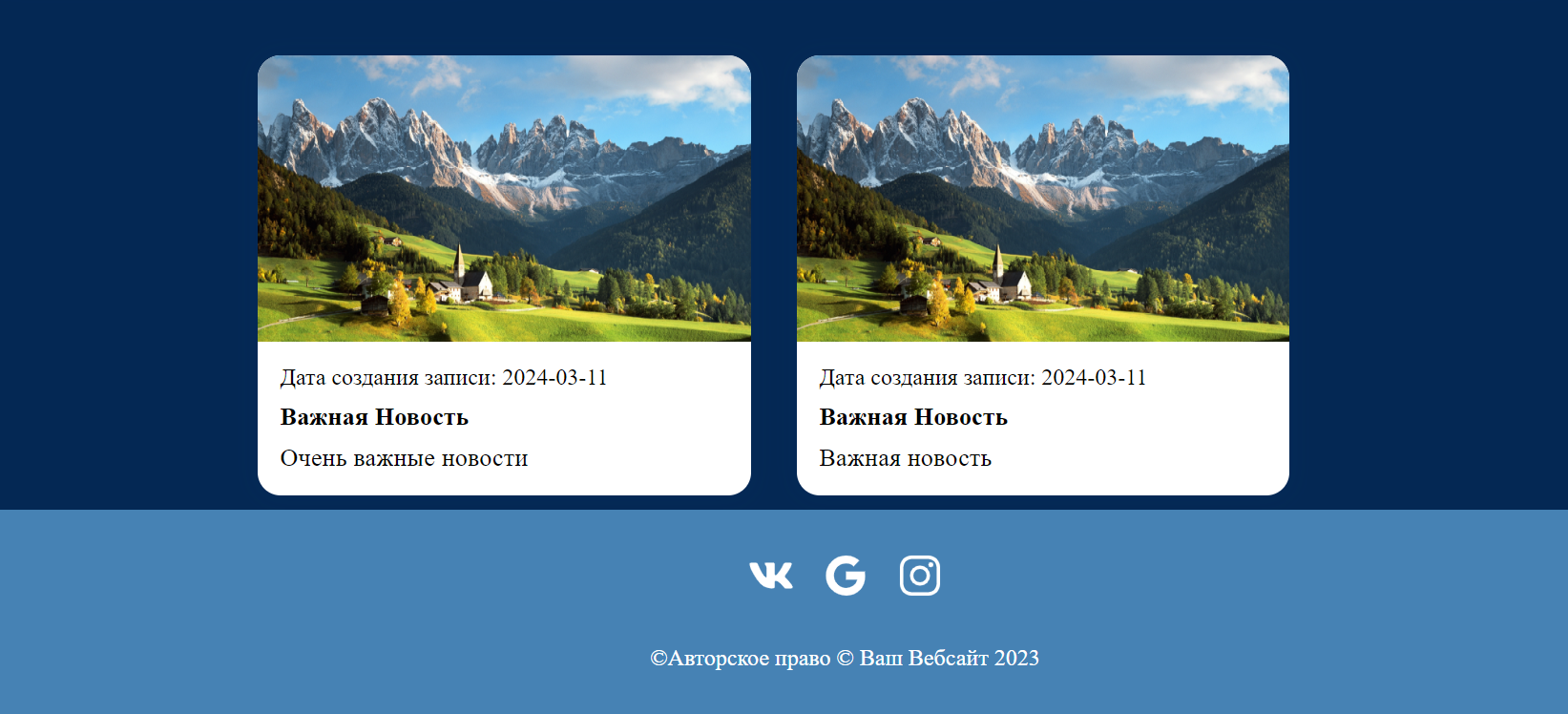


Рисунок 5 – Пример дизайна страницы

Дизайн страницы должен быть привлекательным, функциональным и удобным для пользователей.

**6 Формирование контента**

Страница регистрации:

* это страница, на которой пользователи могут создать учетную запись для доступа к WEB-приложению или сервису. На странице представлены поля для ввода имени пользователя, адреса электронной почты, изображение профиля и пароля. Пользователи должны заполнить все обязательные поля и нажать кнопку "Зарегистрироваться", чтобы завершить процесс регистрации. Также на странице может быть ссылка для перехода на страницу авторизации, если пользователь уже имеет учетную запись.

Страница авторизации:

* это страница, на которой пользователи могут войти в свою учетную запись, используя ранее созданные учетные данные. На странице присутствуют поля для ввода имени пользователя (или адреса электронной почты) и пароля. Пользователи должны правильно заполнить эти поля и нажать кнопку "Войти", чтобы получить доступ к своему аккаунту.

Главная страница:

* это основная страница WEB-приложения, на которой пользователи оказываются после успешной авторизации. На главной странице представлено навигационное меню, чат пользователей, основная информация из созданных досок и новостная лента, также представлен подвал, в котором располагается контактная информация и ссылки на сторонние сайты.

Страница информации по каждой доске:

* на данной странице представлена информация для каждого блока доски, документы, файлы, изображения, которые создаются администратором или руководителем WEB – приложения. Данные на этой странице пользователи не только видят, но также комментируют и отвечают реакциями.

Страница чата:

* на данной странице пользователи могут взаимодействовать друг с другом и так же с руководителем.

Страница администратора:

* на данной странице администратор видит всю информацию, но также может добавлять новую, добавлять новостные блоки, удалять элементы и редактировать их.

Страница руководителя:

* на данной странице руководитель может создавать новые доски и приглашать туда пользователей.

**7 Вёрстка сайта**

Для разработки сайта использовался локальный сервер Open Server, обеспечивающий среду для работы с PHP и базой данных MySQL.

Содержание папки «plan-board» с проектом включает следующие файлы:

* register.php: этот файл представляет собой HTML-страницу для регистрации пользователя на веб-сайте. В нем используется PHP для проверки статуса пользователя и отображения сообщений об ошибках валидации. Форма регистрации содержит поля для ввода имени, адреса электронной почты, пароля и изображения профиля. Пользователь должен заполнить обязательные поля и отправить форму, чтобы зарегистрироваться на сайте. Если введенные данные не проходят валидацию, пользователю отображаются сообщения об ошибках. В конце страницы есть ссылка для перехода на страницу авторизации, если у пользователя уже есть аккаунт;
* index.php: этот файл представляет собой HTML-страницу для авторизации пользователей на веб-сайте "PlanBoard". Она содержит форму входа, позволяющую пользователям ввести свой email и пароль для входа в свою учетную запись. В верхней части страницы используется PHP для проверки статуса пользователя: если пользователь уже авторизован, он будет перенаправлен на другую страницу;
* home.php: этот файл является главной страницей веб-приложения "PlanBoard" для администратора. В файле присутствуют формы для добавления новых досок и новостей, а также списки уже созданных досок и новостей. Страница также содержит шапку с логотипом и навигационным меню, а также подвал с ссылками на социальные сети и информацией об авторском праве. Для динамического обновления контента используются технологии JavaScript и библиотека Vue.js;
* homeruk.php: этот файл является главной страницей веб-приложения "PlanBoard" для руководителя. В файле также присутствуют формы для добавления новых досок;
* homeuser.php: этот файл является главной страницей веб-приложения "PlanBoard". Он предоставляет пользователю интерфейс для просмотра новостей и создания досок с информацией;
* helpers.php: этот файл содержит набор функций и настроек для работы с сессиями, базой данных и другими аспектами веб-приложения. Он предоставляет удобные методы для реализации функций, таких как перенаправление пользователя, установка и получение сообщений, обработка ошибок валидации, загрузка файлов на сервер и другие;
* getUsers.php: этот файл является обработчиком запросов к базе данных и возвращает данные в формате JSON для получения списка пользователей;
* getNews.php: этот файл отвечает за получение данных о новостных блоках из базы данных и их отправку в формате JSON;
* getBlock.php: этот файл отвечает за получение данных о блоках доски из базы данных и их отправку в формате JSON;
* deleteblock.php: этот PHP скрипт выполняет удаление записей из базы данных на основе полученных данных от клиента. Он начинает сессию и подключается к базе данных, используя данные из файла конфигурации. Затем он проверяет наличие соединения с базой данных. После выполнения операции удаления, скрипт формирует ответ в формате JSON, сообщая о результатах операции. Если обязательные данные отсутствуют, скрипт также формирует соответствующее сообщение об ошибке;
* config.php: этот файл устанавливает соединение с базой данных MySQL на локальном сервере. Если соединение не установлено, выводится сообщение об ошибке;
* blockuser.php: этот PHP скрипт отвечает за получение данных о блоках, доступных конкретному пользователю. Сначала он начинает сессию и проверяет, установлен ли идентификатор пользователя в сессии. Если не установлен, скрипт перенаправляет пользователя на страницу входа;
* addnews.php: этот файл PHP отвечает за обработку данных, полученных из формы на клиентской стороне и их добавление в базу данных. Он также устанавливает соединение с базой данных, проверяет наличие ошибок при подключении, получает данные из POST-запроса, обрабатывает изображение (если оно было загружено), генерирует SQL-запрос для добавления данных в базу данных, выполняет запрос и выводит сообщение об успехе или ошибке. После обработки данных он закрывает соединение с базой данных;
* addblock.php: этот файл PHP отвечает за обработку данных, полученных из формы на клиентской стороне, добавление новой записи в базу данных и связанных с ней записей в других таблицах. Он также проверяет наличие обязательных данных в запросе, подготавливает и выполняет SQL-запросы для добавления записей в базу данных, обрабатывает возможные ошибки и возвращает JSON-ответ с сообщением об успешном создании записи или с информацией об ошибке. После обработки данных он закрывает соединение с базой данных;
* register.php: Этот PHP-скрипт отвечает за обработку данных, полученных из формы регистрации на клиентской стороне. Он осуществляет проверку введенных пользователем данных, таких как имя, адрес электронной почты, пароль и подтверждение пароля. Далее он выполняет различные проверки на корректность данных, такие как уникальность имени пользователя и адреса электронной почты, минимальная длина пароля, формат адреса электронной почты и тип загружаемого аватара. В случае наличия ошибок в данных, он возвращает пользователя на страницу регистрации с сообщениями об ошибках;
* logout.php: этот PHP-скрипт обрабатывает выход пользователя из системы. Он проверяет метод HTTP-запроса: если он является POST, то вызывается функция для выхода пользователя из сессии. После чего независимо от результата запроса, пользователь перенаправляется на главную страницу приложения;
* login.php: этот PHP-скрипт обрабатывает вход пользователя в систему. Он получает адрес электронной почты и пароль из POST-запроса, затем выполняет проверки валидности данных. Если данные валидны, скрипт ищет пользователя в базе данных по адресу электронной почты. Если пользователь найден и пароль совпадает, он аутентифицируется путем установки идентификатора пользователя и его роли в сессию. Затем происходит перенаправление на соответствующую страницу в зависимости от роли пользователя;
* styles.css: этот файл для основной страницы, обеспечивающий её внешний вид и структуру;
* login.css: этот файл для страницы авторизации, обеспечивающий её внешний вид и структуру;
* reg.css: этот файл для страницы регистрации, обеспечивающий её внешний вид и структуру;
* news.js: этот файл является клиентским скриптом Vue.js, который используется для создания интерактивного интерфейса на веб-странице. Он загружает список новостей с сервера при загрузке страницы и отображает их на странице;
* code.js: этот скрипт файл отвечает за остановку анимации загрузки и скрытие загрузочного блока после полной загрузки содержимого страницы;
* block.js: этот скрипт файл отвечает за запрос данных о блоках с сервера и их отображение на странице. При загрузке страницы он отправляет запрос на сервер, ожидает ответ и затем обновляет массив блоков в объекте Vue. Если блоков нет, устанавливается соответствующее сообщение;
* app2.js: этот скрипт файл используется для управления интерфейсом, связанным с блоками на веб-странице. Он обеспечивает получение данных о блоках с сервера, их отображение и возможность удаления. Файл также отвечает за открытие и закрытие модального окна для подтверждения удаления блока;
* app1.js: этот скрипт файл отвечает за управление формой добавления новостей на веб-странице. Он обрабатывает ввод данных пользователем, отправляет их на сервер для обработки и отображает сообщения об успешном или неудачном выполнении операции. Файл также обеспечивает функциональность для выбора изображения и его загрузки на сервер;
* app.js: этот скрипт файл отвечает за управление формой добавления блоков на веб-странице. Он обрабатывает ввод данных пользователем, отправляет их на сервер для обработки и отображает сообщения об успешном или неудачном выполнении операции. Файл также обеспечивает функциональность для выбора пользователей из списка и передачи их идентификаторов на сервер.

Рассмотрим реализацию WEB – приложения и его взаимодействие с базой данных. На рисунке 6 представлена форма регистрации. В форме регистрации пользователь вводит свои уникальные данные. Для отправки в базу данных, они все должны быть заполнены. Когда пользователь нажимает на кнопку продолжить, происходит проверка на заполнение данных и запрос отправляется на сервер, в базе данных появляется информация о пользователе, если запрос был корректно сформирован.

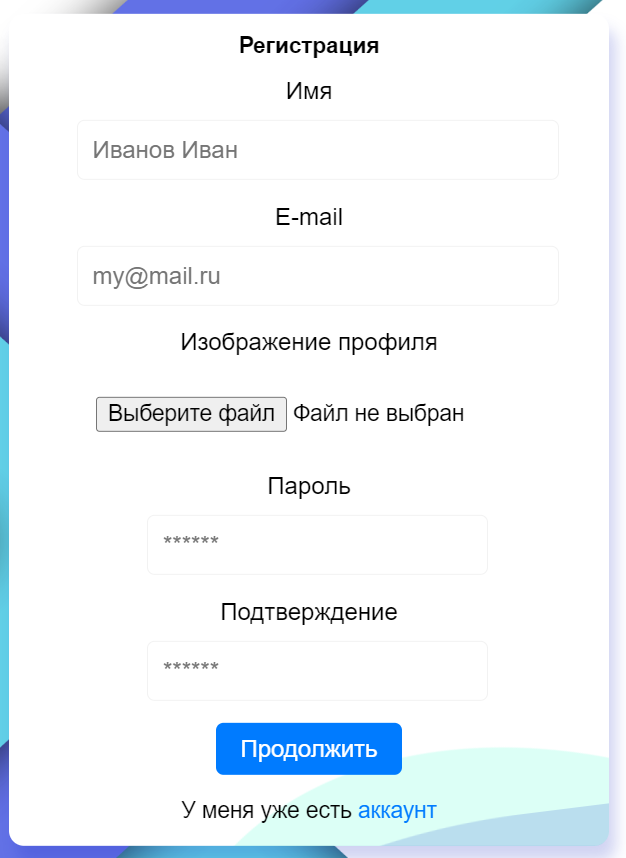


Рисунок 6 – Форма регистрации

Теперь пользователь может пройти авторизацию, представлена на рисунке 7.

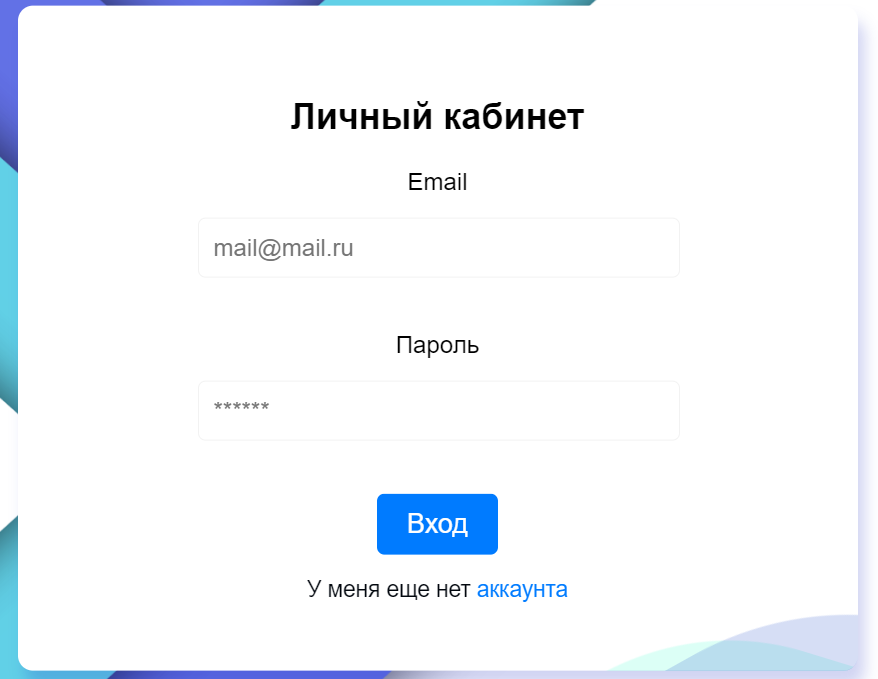


Рисунок 7 – Форма авторизации

В форму авторизации пользователь, администратор или руководитель вводят данные, которые были указаны при регистрации. Они отправляются на сервер, где происходит проверка данных, если данных не будет в базе данных, то вернется ошибка. Во время отправки данных устанавливается идентификатор пользователя, по сессии происходит проверка роли пользователя и открывается страница в зависимости от роли. Рассмотрим WEB – приложение со стороны работы администратора на рисунке 9.

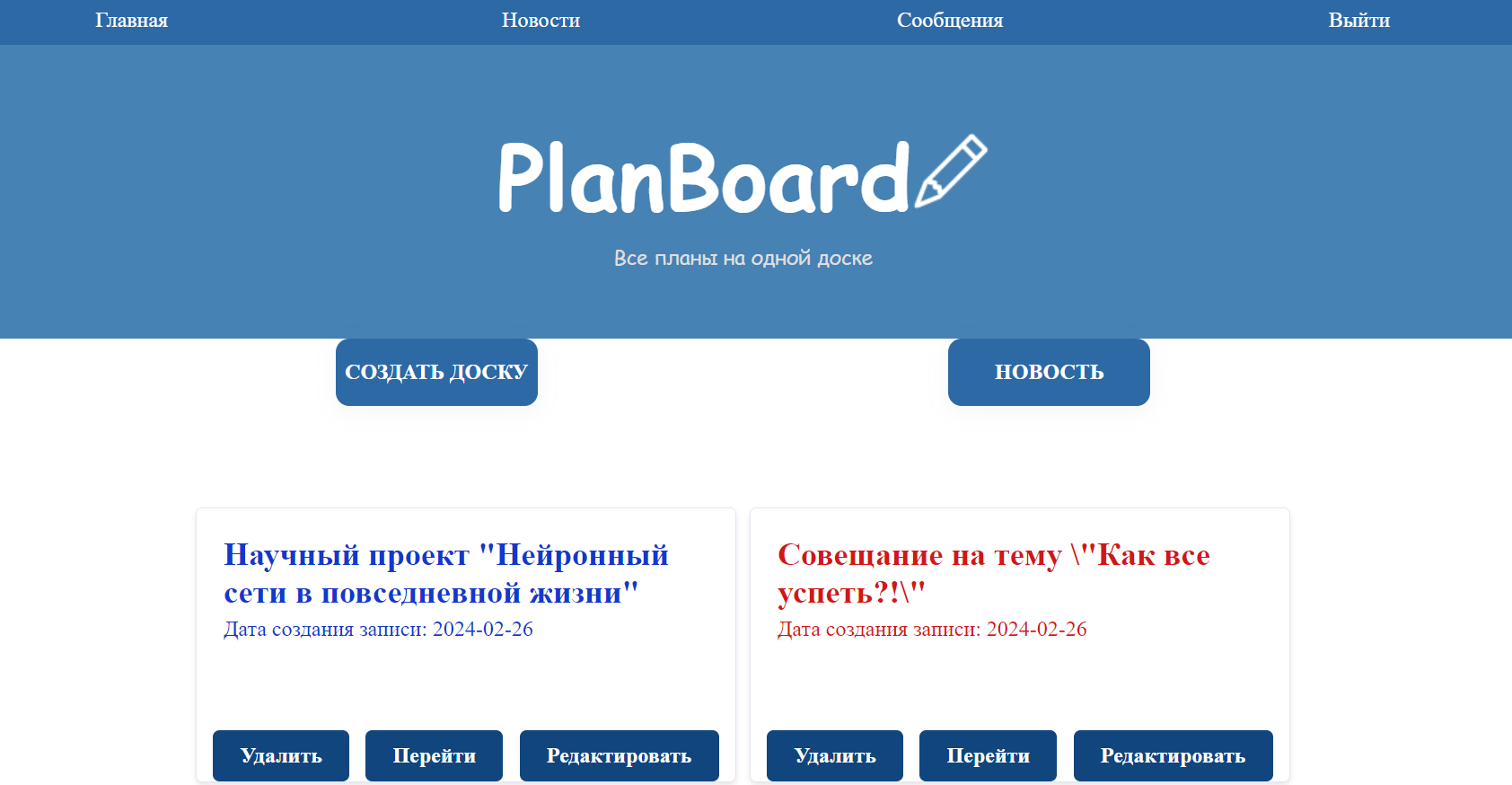


Рисунок 9 – Страница администратора

На этой странице администратор с помощью динамических форм «Создать доску» и «Новость» может добавить данные на страницу приложения, работа данных форм представлена на рисунке 10.

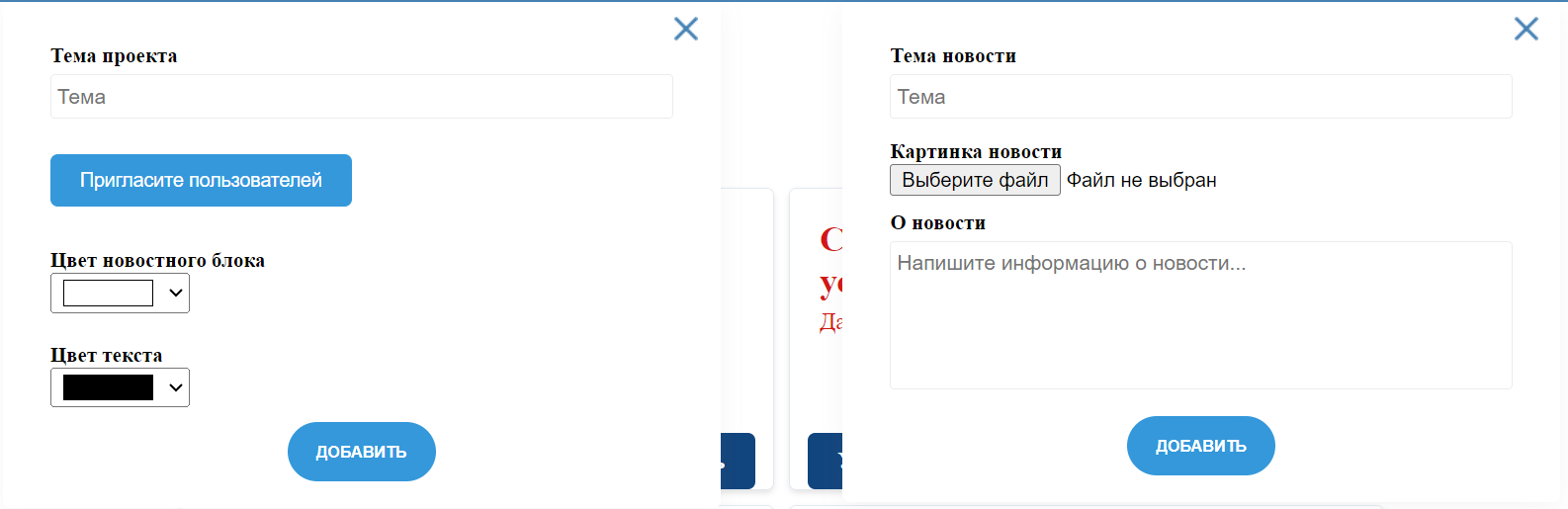


Рисунок 10 – Динамические формы отправки

Администратор заполняет форму данными и отправляет их на сервер, если данные указаны корректно они добавляются в базу данных и затем отображаются на странице,отображение блоков на рисунке 11.

Администратор может удалять и редактировать данные в блоке, выбирается блок на удаление и происходит запрос по идентификатору блока, если он корректный, блок удаляется. Так же формируется запрос на редактирование, данные отправляются на сервер и перезаписываются в базе данных.

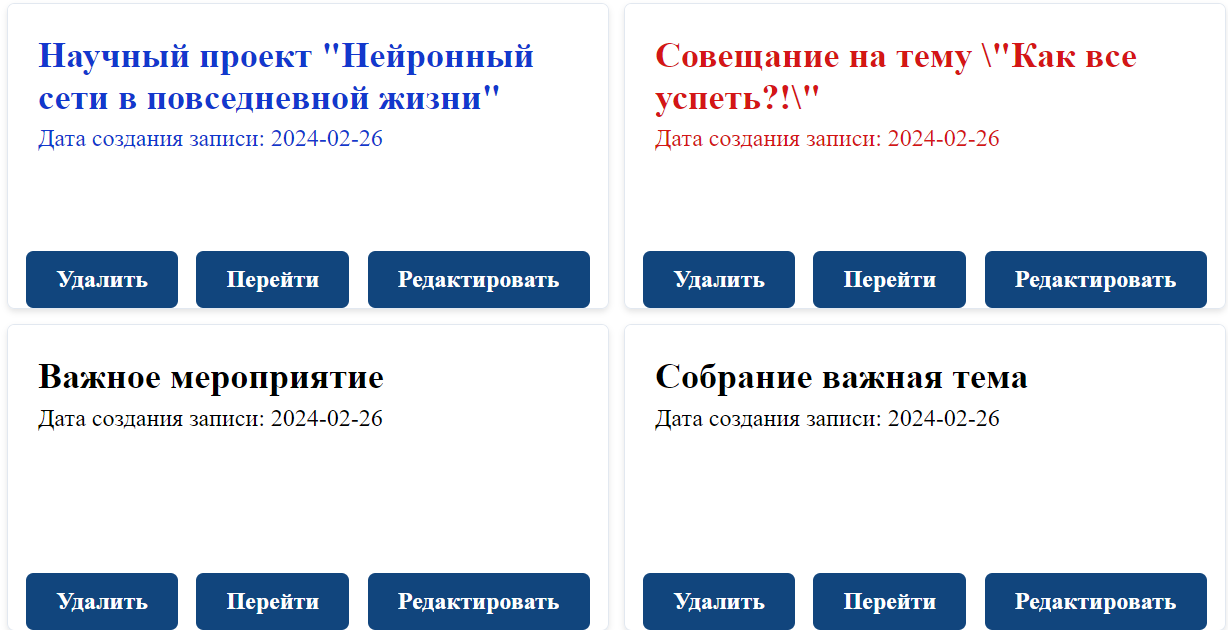


Рисунок 11 – Блоки досок

Если рассматривать работу приложения со стороны руководителя,то руководитель может добавлять только новостные блоки.

Данные заполняются в форме добавления новостных блоков и отправляются на сервер.Корректные данные добавляются в базу данных и отображаются на странице, рисунок 12.

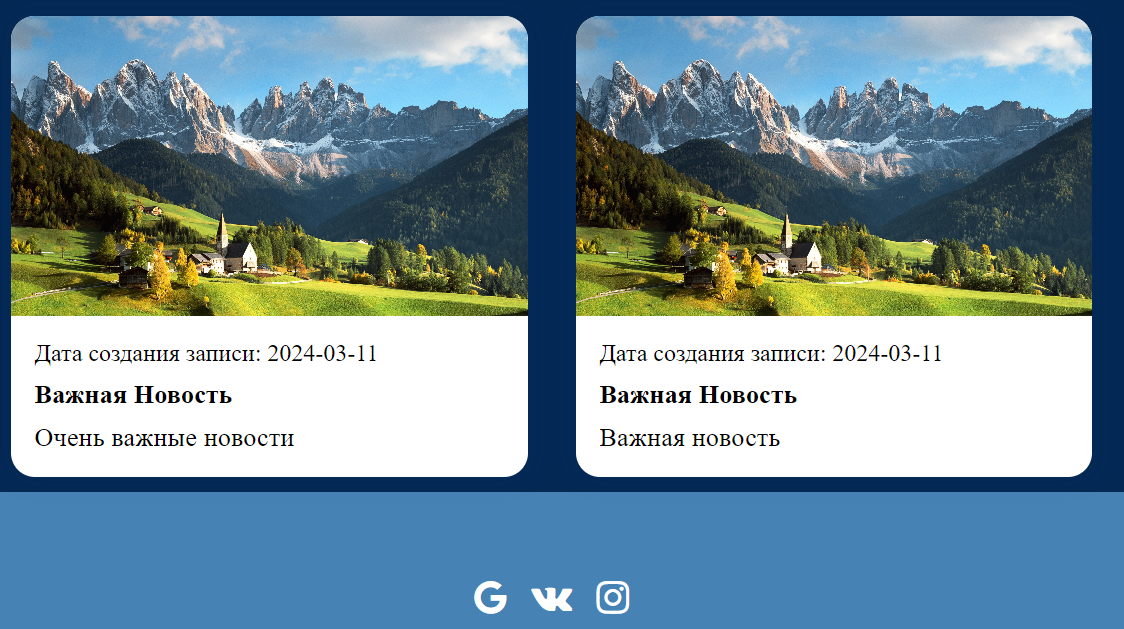


Рисунок 12 – Новостные блоки

Работа WEB – приложения со стороны пользователя представлена на рисунке 13.

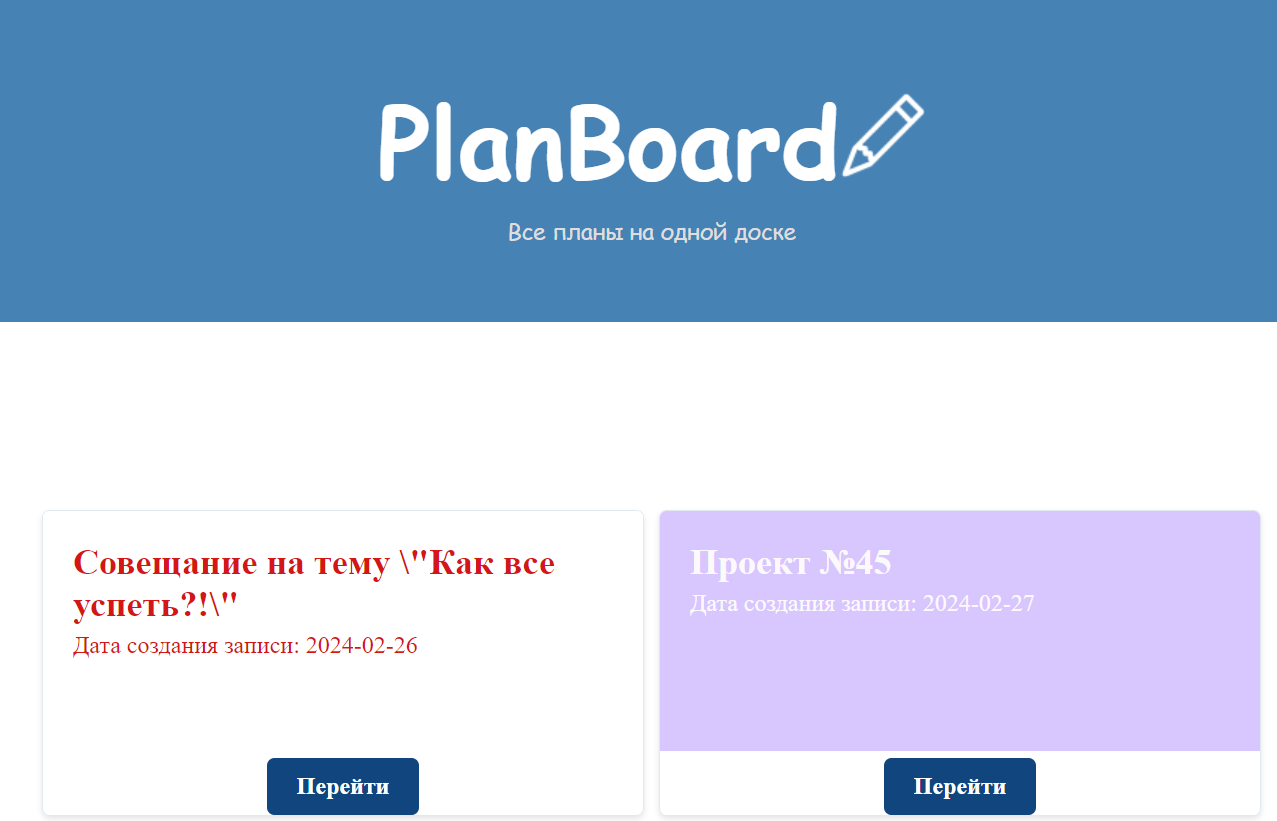


Рисунок 13 – Страница пользователя

Пользователь может просматривать только те блоки, которые ему доступны.

Все описание работы серверной части можно более детально рассмотреть с помощью алгоритма на рисунке 13.

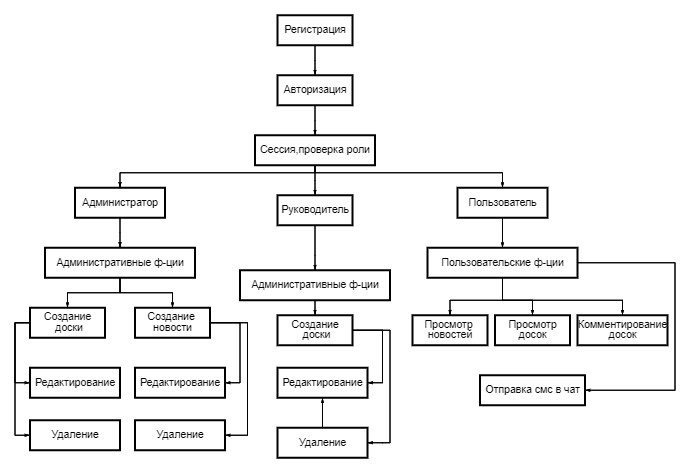


Рисунок 13 – Алгоритм работы серверной части

**8 Тестирование и размещение сайта в сети**

Для обеспечения работоспособности сайта необходимо пройти через целый ряд тестовых сценариев, чтобы убедиться, что все функции работают правильно. Вот общий список тестовых сценариев для проверки работоспособности сайта:

Регистрация нового пользователя:

* ввести некорректные данные нового пользователя (например, неуникальный логин, или неподходящее по формату изображение) в форму регистрации;
* убедиться, что при попытке регистрации с некорректными данными выдается соответствующая ошибка и данные не добавляются в базу данных;
* ввести корректные данные нового пользователя в форму регистрации. Нажать кнопку "Зарегистрироваться" и убедиться, что пользователь успешно добавлен в базу данных. Проверить таблицу пользователей в базе данных, чтобы убедиться, что новая запись была добавлена с корректными данными.

Авторизация пользователя:

* ввести учетные данные незарегистрированного пользователя (логин и пароль) в форму авторизации. Нажать кнопку "Войти" и убедиться, что система выдает ошибку входа на сайт;
* ввести учетные данные зарегистрированного пользователя (логин и пароль) в форму авторизации. Нажать кнопку "Войти" и проверить корректность входа на сайт;
* убедиться, что после успешной авторизации пользователь перенаправляется на свою страницу профиля или главную страницу доски. Проверить, что доступные функции, такие как создание, редактирование или удаление записей, доступны только авторизованным пользователям.

Эти шаги помогут убедиться в корректной работе функций регистрации и авторизации на виртуальной доске. В случае возникновения ошибок необходимо проанализировать причины и внести соответствующие исправления.

Создание нового блока:

* ввести некорректные данные для создания нового блока (например, пустой текст). Нажать кнопку "Добавить блок" и убедиться, что приложение выдает ошибку и не добавляет блок с некорректными данными;
* ввести корректные данные для создания нового блока (например, заполните текст и добавьте изображение). Нажать кнопку "Добавить блок" и убедитесь, что новый блок успешно добавлен на доску.

Редактирование блока:

* выбрать блок для редактирования;
* нажать кнопку "Редактировать";
* изменить текст блока;
* попытаться сохранить изменения;
* возникла ошибка при сохранении изменений;
* уведомить пользователя об ошибке и предложить повторить попытку;
* проверить, что данные блока остались неизменными на доске;
* записать сообщение об ошибке для администратора или разработчика.

Удаление блока:

* выбрать блок для удаления;
* нажать кнопку "Удалить";
* появилось подтверждение удаления блока;
* подтвердить удаление;
* возникла ошибка при удалении блока;
* уведомить пользователя об ошибке и предложить повторить попытку;
* проверить, что блок все еще присутствует на доске;
* записать сообщение об ошибке для администратора или разработчика.

В обоих случаях важно обеспечить корректное информирование пользователя о возникших проблемах и предложить понятные инструкции о дальнейших действиях. Также важно регистрировать и сохранять информацию об ошибках для последующего анализа и устранения проблемы.

Редактирование и удаление блока, успешная работа:

* выбрать блок, который необходимо отредактировать, и нажать кнопку "Редактировать». Изменить текст блока и сохранить изменения. Убедиться, что данные блока успешно изменены и отображаются корректно на доске;
* выбрать блок, который необходимо удалить, и нажать кнопку "Удалить». Убедиться, что блок был успешно удален и больше не отображается на доске.

Создание новости:

* ввести некорректные данные для создания новости (например, пустой текст или недопустимый формат изображения). Нажать кнопку "Добавить новость" и убедиться, что приложение выдает ошибку и не добавляет новость с некорректными данными;
* ввести корректные данные для создания новости (например, заполните текст и добавьте изображение). Нажать кнопку "Добавить блок" и убедиться, что новый блок успешно добавлен на доску.

Редактирование новости:

* выбрать блок новости для редактирования;
* нажать кнопку "Редактировать";
* внести изменения в текст или изображение новости;
* попытаться сохранить изменения;
* возникла ошибка при сохранении изменений;
* уведомить пользователя об ошибке и предложить повторить попытку;
* проверить, что данные блока новости остались неизменными на доске;
* записать сообщение об ошибке для администратора или разработчика.

Удаление новости:

* выбрать блок новости для удаления;
* нажать кнопку "Удалить";
* появилось подтверждение удаления новости;
* подтвердить удаление;
* возникла ошибка при удалении новости;
* уведомить пользователя об ошибке и предложить повторить попытку;
* проверить, что блок новости все еще присутствует на доске;
* записать сообщение об ошибке для администратора или разработчика.

Редактирование и удаление новости, успешная работа:

* выбрать блок новости, который необходимо отредактировать, и нажмите кнопку "Редактировать». Изменить текст или изображение новости и сохранить изменения. Убедиться, что данные блока успешно изменены и отображаются корректно на доске;
* выбрать блок новости, который необходимо удалить, и нажать кнопку "Удалить». Убедиться, что блок был успешно удален и больше не отображается на доске.

Отображение блоков:

* проверить, что все блоки корректно отображаются на доске и сохраняют свои положения и данные после перезагрузки страницы.

Обработка ошибок и отображение сообщений:

* ввести некорректные данные или выполните действие, которое может привести к ошибке;
* убедиться, что приложение корректно обрабатывает ошибки, выводит информативные сообщения об ошибках и предлагает решения или подсказки для пользователя.

Провести тестирование производительности, чтобы убедиться, что WEB-приложение работает стабильно и быстро при добавлении большого количества блоков или при многопользовательском использовании.

Нефункциональное тестирование:

* производительность: оценить скорость загрузки страниц и выполнения запросов к базе данных. Убедиться, что сайт работает быстро и отзывчиво;
* безопасность: проверить наличие уязвимостей безопасности. Обеспечить защиту данных пользователей и аутентификацию;
* совместимость: проверить работоспособность сайта на различных браузерах (Chrome, Firefox, Safari, Edge) и устройствах (компьютеры, планшеты, мобильные телефоны);
* отзывчивый дизайн: убедиться, что сайт корректно отображается и работает на разных разрешениях экрана и устройствах;
* проверка корректного отображения сайта и его функций на различных мобильных устройствах с разными размерами экранов;
* убедиться, что элементы управления и контент легко доступны и пригодны для использования на мобильных устройствах;
* проверка корректности управления сеансами пользователей, чтобы убедиться, что авторизованные пользователи не могут получить доступ к защищенным страницам или функциям без входа в систему;
* проверка производительности базы данных при обработке запросов на чтение и запись;
* оценка времени ответа базы данных при выполнении различных операций, таких как создание, обновление и удаление записей.

После проведения тестирования необходимо внимательно проанализировать результаты, исправить обнаруженные ошибки и улучшить функциональность и удобство использования виртуальной доски.

Подготовка базы данных на Handy Host:

* cоздание базы данных PostgreSQL через панель управления Handy Host;
* cоздание пользователя базы данных с необходимыми правами доступа.

Тестирование соединения:

* написание тестового скрипта для подключения к базе данных PostgreSQL на Handy Host и выполнения простого запроса;
* запуск тестового скрипта для проверки успешного подключения и выполнения запросов.

Импорт данных:

* подготовка данных для импорта, если они есть;
* использование инструментов Handy Host или командной строки для импорта данных в созданную базу данных.

Настройка безопасности:

* проверка и обновление настроек безопасности базы данных PostgreSQL на Handy Host, включая права доступа к таблицам и пользователям;
* регулярное обновление паролей и других настроек безопасности.

Настройка резервного копирования:

* настройка регулярного резервного копирования базы данных PostgreSQL на Handy Host через инструменты, предоставленные хостинг-провайдером.

Масштабирование и оптимизация:

* оценка производительности и масштабирование базы данных на Handy Host, при необходимости увеличение выделенных ресурсов;
* оптимизация запросов и индексов для повышения производительности.

Мониторинг и обслуживание:

* регулярное мониторинг состояния и производительности базы данных на Handy Host;
* реагирование на любые проблемы или сбои в работе базы данных и веб-приложения;
* обновление веб-приложения и базы данных для поддержания их работоспособности и безопасности.

**Заключение**

При выполнении работы над приложением для информационного сопровождения задачи выполняемой группой работников были пройдены следующие этапы:

* планирование и проектирование: Определение функциональных и нефункциональных требований к приложению, создание архитектуры приложения и выбор технологий;
* разработка функциональности: Создание основных функций приложения, таких как добавление, редактирование и удаление блоков, работа с пользователями, загрузка изображений и другие;
* тестирование и отладка: Проведение тестирования для обнаружения и исправления ошибок, а также улучшение производительности и безопасности приложения;
* доработка и оптимизация: Улучшение интерфейса, добавление дополнительных функций, оптимизация кода и базы данных для повышения производительности и улучшения пользовательского опыта.

В ходе выполнения данной работы было создано приложение для информационного сопровождения задачи выполняемой группой работников, которое позволяет просматривать, создавать, редактировать и удалять блоки информации на виртуальной доске. Реализация приложения включает в себя использование современных технологий веб-разработки, таких как Vue.js для создания динамического интерфейса, PHP для обработки серверных запросов и взаимодействия с базой данных, а также AJAX для обмена данными между клиентом и сервером без перезагрузки страницы. Приложение обладает базовым набором функций, необходимых для работы с блоками информации, и может быть доработано и расширено в будущем в соответствии с потребностями пользователей.

**Перечень использованных информационных ресурсов**

1. Официальная документация языка разметки HTML [Электронный ресурс], URL: https://www.w3.org/2019/04/WHATWG-W3C-MOU.html.

2. Официальная документация каскадных таблиц стилей [Электронный ресурс], URL: https://www.w3.org/Style/CSS/Overview.en.html.

3. Официальная документация языка программирования JS [Электронный ресурс], URL: https://www.javascript.com.

4. Официальная документация языка программирования РНР [Электронный ресурс], URL: <https://www.php.net/manual/ru/index.php>

5. Официальная документация MySQL [Электронный ресурс], URL: <https://www.opennet.ru/docs/RUS/mysqlrus/>