МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РЕСПУБЛИКИ КАЗАХСТАН

КАЗАХСКИЙ УНИВЕРСИТЕТ ТЕХНОЛОГИИ И БИЗНЕСА

Кафедра «Информационные технологии»

**Отчет**

**о научно-исследовательской работе**

**НИРМ.04**

Факультет Технологический

Образовательная программа: 7М06136 - Информационные системы

Тулегенов Алишер Канатович

Направление обучения: ғылыми және педагогикалық/научное и педагогическое

Научный руководитель Акишев Каршыга Максутович, к.т.н., асс. профессор

**Астана 2024**

**Содержание**

Введение

1. Автоматизация техники безопасности

1.1. Требования к СУОТ

1.2. Создание электронного журнала по технике безопасности

1.3. Что требуется в ПО для конкурентоспособности

2. Автоматизация управления журналом технической безопасности с использованием современных информационных технологий

2.1. Архитектура приложения

2.2. Структура программного обеспечения

2.3. Создание программного интерфейса

3. Тестирование и внедрение ПО

Заключение

Список использованной литературы

**Введение**

Автоматизация процессов ведения журнала техники безопасности открывает перед предприятиями новые возможности для оптимизации и повышения эффективности своей безопасностной культуры. Разработка программного обеспечения, способного интегрировать различные аспекты управления безопасностью, от мониторинга состояния оборудования до анализа данных по происшествиям, играет ключевую роль в этом процессе.

Онлайн ведение журнала техники безопасности не только сокращает временные затраты на рутинные операции по заполнению и обработке данных, но и повышает их достоверность и актуальность. Возможность оперативного обновления информации и мгновенного доступа к ней со стороны ответственных сотрудников существенно снижает риски возникновения чрезвычайных ситуаций и повышает уровень готовности предприятия к ним.

Кроме того, цифровизация журнала техники безопасности способствует улучшению системы аналитики и прогнозирования. Современные алгоритмы машинного обучения и анализа данных позволяют выявлять скрытые закономерности и тренды, на основе которых можно разрабатывать более эффективные стратегии предотвращения происшествий и управления рисками.

Более того, переход на онлайн формат ведения журнала техники безопасности открывает новые возможности для обучения и повышения квалификации персонала. Интерактивные материалы, тесты и тренировки, доступные через цифровую платформу, делают процесс обучения более удобным и доступным, что способствует формированию культуры безопасности на предприятии.

Таким образом, автоматизация журнала техники безопасности представляет собой не просто техническое обновление, а важный шаг в направлении улучшения безопасности на предприятии в целом. Это инвестиции в будущее, направленные на повышение эффективности и надежности производственных процессов и защиту интересов как сотрудников, так и предприятия в целом.

*Актуальность*. В современном Казахстане, безопасности и охране труда на производстве уделяется большое внимание. Вместе с тем, на предприятиях до сих пор работа инженера по технике безопасности организована в традиционном стиле. Принятый в 2022 году Закон о цифровизации, требует перевода документации в цифровой формат с целью снижения бюрократии и повышения производительности труда.

*Цель работы*. Повышение эффективности труда инженера по технике безопасности предприятия.

*Задачи исследования*:

1. Анализ функциональных обязанностей инженера по технике безопасности предприятия.
2. Исследование цифровых программ автоматизации деятельности инженера по технике безопасности предприятия.
3. Выбор и обоснование инструментов для разработки автоматизированного рабочего места инженера по технике безопасности предприятия.
4. Разработка программы «Автоматизированное рабочее места инженера по технике безопасности предприятия».
5. Апробация программы «Автоматизированное рабочее места инженера по технике безопасности предприятия» на базе действующего промышленного предприятия.

*Научная новизна* заключается в разработанной методике и программа для компьютера.

*Научная и практическая значимость* исследования является повышение эффективности и производительности персонала, внедрение современных программных продуктов.

1. **Автоматизация техники безопасности**

Автоматизация журнала по технике безопасности (ТБ) играет критическую роль в современных организациях и индустриях, где обеспечение безопасности персонала, оборудования и окружающей среды имеет высший приоритет. Вот некоторые ключевые причины, почему автоматизация журнала по ТБ является важным и неотъемлемым аспектом:

Эффективность и точность: Ручное ведение журнала ТБ может быть трудоемким и подверженным ошибкам человека. Автоматизация устраняет человеческий фактор, что приводит к повышению точности и эффективности сбора, анализа и документирования данных. Это особенно важно в условиях высокой загруженности персонала и больших объемов информации.

Соблюдение нормативов и стандартов: В различных отраслях существуют строгие нормативы и стандарты в области безопасности. Автоматизированный журнал ТБ способствует соблюдению этих стандартов, предоставляя надежный способ систематической регистрации и анализа данных безопасности. Это облегчает аудит и подготовку отчетов, что важно для соблюдения требований и нормативов.

Оптимизация ресурсов: Позволяет анализировать данные и выявлять области, где можно оптимизировать использование ресурсов. Анализ показателей безопасности и производственной эффективности может помочь выявить узкие места, проблемы и неэффективные практики. Это позволяет распределять ресурсы более эффективно и повышать производительность.

Улучшение культуры безопасности: Внедрение автоматизированных систем ТБ способствует формированию культуры безопасности в организации. Регулярный мониторинг и анализ данных создает более высокий стандарт безопасности, что образует осведомленный и ответственный подход персонала к соблюдению безопасных практик и процедур.

**1.1 Требования к СУОТ**

*Обязательность наличия штатного специалиста по охране труда*. Производственные предприятия со штатной численностью до 50 сотрудников могут самостоятельно определяться с введением cистемой управления охраны труда (СУОТ). Если штат компании превышает 50 человек, наличие собственного специалиста по охране труда становится обязательным.

*Обучение и инструктаж*. Все сотрудники, принимаемые на рабочие места, связанные с риском для здоровья, в обязательном порядке проходят первичный инструктаж по нормативам и правилам, обеспечивающим безопасность трудовой деятельности. Периодичность повторного инструктажа для персонала составляет шесть и три месяца в зависимости от степени опасности конкретного рабочего места.

*Внеплановый инструктаж* или переподготовка сотрудников в специализированных учебных центрах проводятся при установке нового технологического оборудования или изменениях, касающихся условий труда. Обеспечение проведения данных мероприятий возлагается на специалиста по охране труда, который составляет соответствующие списки работников, заверяющиеся подписью руководителя предприятия.

*Информационное обеспечение по охране труда*. Внедрение на производстве методов СУОТ сопровождается обязательным оповещением персонала об изменениях в политике охраны труда посредством собраний, внеплановых инструктажей и размещения информации на специальных стендах. Сотрудники, чьи рабочие места связаны с риском для жизни и здоровья должны быть проинформированы в отношении своих прав на материальную компенсацию, сокращенный рабочий день и увеличенное время отдыха (руководство организовывает ознакомление сотрудников с соответствующими документами под личную роспись). Работники могут также оформлять подписку на получение периодических изданий, посвященных изменениям и нововведениям в охрану труда.

*Документальное оформление*. О наличии на производстве СУОТ свидетельствуют нормативные документы, в которых излагается политика предприятия в сфере охраны труда и методы, обеспечивающие защиту здоровья персонала. От правильности документального оформления во многом зависит эффективность планирования и безопасность функционирования производственных процессов, являющихся основными задачами современного менеджмента.

**1.2 Создание электронного журнала по технике безопасности**

Многие организации имеют разветвленную филиальную структуру, со множеством удаленных сотрудников. В соответствии с законодательством все они должны проходить инструктажи по охране труда. Сведения обо всех инструктажах должны заноситься в журналы учета.

Возникает вопрос соблюдения этих требований для удаленных сотрудников организации, ведь не всегда представляется возможным приехать с бумажным журналом к сотруднику, расстояния у нас в стране большие. Проблемы налицо: ведение классических бумажных журналов представляет собой довольно утомительный и скучный процесс, усложненные процедуры проверки наличия и правильности заполнения журналов в филиале. В случае ошибок приходится вносить изменения, переписывать записи, часто нужно записывать в журнал одни и те же данные (например, фамилию, дату). Подпись в журнале легко подделать, при необходимости вообще можно переписать журнал "с нуля" за ночь. В результате бумажные журналы мало того, что не обеспечивают надежный учет, так еще и загружают сотрудников рутинной работой, требуют место для хранения, создают сложности в случае необходимости получения подписи сотрудника, находящегося в другом месте. Посещение офиса удаленным сотрудником часто бывает проблематично, а иногда и просто невозможно. Выход из этой ситуации есть, если организовать ведение журналов в электронном виде. В этом поможет закон "Об электронной подписи", описывающий условия равнозначности собственноручной и электронной подписей. Никаких запретов на ведение журналов в электронном виде нет. Но самое главное, в случае с электронными журналами и современными технологиями видеоконференцсвязи может быть легко реализована технология дистанционного инструктажа.

Для организации инструктажа и учета в электронном виде необходимо: выбрать способ формирования электронной подписи (это может быть и простая подпись на базе паролей, и графическая подпись, создаваемая с помощью специальных планшетов, и квалифицированная электронная подпись, полученная в аккредитованном удостоверяющем центре); выбрать компьютер для установки серверной части системы ведения журналов в электронном виде; назначить лицо, ответственное за ведение журналов в электронном виде, и контроль времени на сервере; установить требуемое количество клиентов системы, назначить ответственных за конкретные журналы, распределить роли пользователей (чтение / запись).

Преимуществами реализации проекта по созданию электронного журнала регистрации инструктажей, стажировки и проверки знаний по охране труда: карточка по процессам и результатам (HR); автоматическое оповещение, в случае если необходимый инструктаж просрочен; отображение списка работников, которым не проведен (просрочен) необходимый инструктаж; возможность определения периодичности проведения инструктажей; распоряжение об отстранении; протокол проверки знаний; отчет по работнику / профессии.

Нормативно не установлено требование, в соответствии с которым журналы проведения инструктажей по охране труда должны вестись в бумажном виде. Следовательно, журналы проведения инструктажей по охране труда можно вести в электронном виде. Также нужно обеспечить сохранность электронных журналов проведения инструктажей по охране труда, защиту от случайного удаления или некорректных правок.

Что касается приложения для автоматизации техники безопасности, оно должно выполнять ряд ключевых функций, чтобы эффективно поддерживать безопасность и обеспечивать надлежащее управление рисками в организации. Вот некоторые из основных функций, которые такое приложение может предоставлять:

Документирование событий - приложение должно предоставлять возможность документировать все события, инциденты и меры, предпринятые для реагирования. Каждое событие должно быть четко описано, включая дату, время, тип события и сопутствующие детали. Эти записи могут служить исторической базой для анализа, отчетности и аудита, а также для обучения и разработки лучших практик.

Соблюдение стандартов и нормативов - приложение может быть настроено для автоматического соблюдения стандартов безопасности и нормативных требований. Это включает в себя создание шаблонов отчетов, документации и процедур в соответствии с требованиями безопасности и регулятивами.

**1.3 Что требуется в ПО для конкурентоспособности**

Функциональность приложения объединяющее мониторинг, анализ и управление активами и безопасностью. Функции мониторинга состояния оборудования, оперативного реагирования на угрозы, а также анализа данных для оптимизации операций.

Интеграция и гибкость - обеспечение возможности интеграции с другими системами, что позволит пользователям создавать комплексные решения. Гибкость настройки и расширения приложения для адаптации к разнообразным отраслям и требованиям клиентов также важна.

Удобный интерфейс - интуитивно понятный, который позволит пользователям легко осваивать и использовать приложение, даже без глубоких технических знаний.

Облачная инфраструктура. Возможность хранения данных и обработки в облачной среде, что обеспечит доступность и масштабируемость приложения.

Безопасность данных. Высокий уровень безопасности данных, используя современные методы шифрования, аутентификации и защиты от несанкционированного доступа.

Производительность и масштабируемость. Гарантировать высокую производительность даже при обработке больших объемов данных, а также возможность масштабирования в зависимости от потребностей.

1. **Автоматизация управления журналом технической безопасности с использованием современных информационных технологий**

Разработанное приложение представляет собой клиент-серверную систему, целью которой является автоматизация процессов управления журналом технической безопасности. В основе приложения лежит возможность хранения, ознакомления и редактирования статей по технике безопасности, обеспечивая удобный доступ к информации для пользователей. Администратор системы имеет возможность управлять базой данных, добавлять новых пользователей, вносить статьи и следить за прогрессом пользователей при прохождении материалов, включая тесты с ответами. Разработка приложения направлена на повышение эффективности работы специалистов в области технической безопасности и обеспечение более удобного и прозрачного взаимодействия между администраторами и пользователями.

Во время создания программного обеспечения для автоматизации журнала техники безопасности нужно обратить внимание, что существует множество технологий и инструментов. Выбор технологий зависит от целей и требований к программному обеспечению.

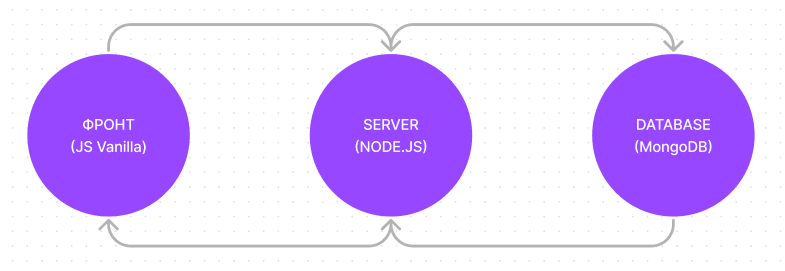
Создание веб-приложения имеет несколько значительных преимуществ, которые делают его предпочтительным выбором для конкуренции с другими программными обеспечениями

* Доступность: Веб-приложения доступны через веб-браузер из любого места с доступом в интернет. Это позволяет пользователям работать с приложением на различных устройствах, включая компьютеры, планшеты и смартфоны, без необходимости установки дополнительного программного обеспечения.
* Масштабируемость: Веб-приложения легче масштабируются, так как серверная инфраструктура может быть настроена для обработки большого числа пользователей без значительных изменений в клиентской части приложения.
* Удобство обновлений: Внесение изменений и обновлений в веб-приложение проще и быстрее. Новые версии могут быть развернуты на сервере, и пользователи сразу получают доступ к обновленным функциям.
* Интеграция: Веб-приложения легче интегрировать с другими онлайн-сервисами и системами, что позволяет создавать комплексные решения и максимизировать функциональность.
* Кроссплатформенность: Веб-приложения могут быть использованы на различных операционных системах (Windows, macOS, Linux) без необходимости разработки отдельных версий для каждой платформы.
* Безопасность: Доступ к данным и функциональности контролируется на серверной стороне, что может обеспечить более высокий уровень безопасности.
* Сервисная модель: Веб-приложения могут использовать подписочную или облачную модель лицензирования, что облегчает клиентам начало работы и управление расходами.
* Опыт пользователя: Современные веб-приложения могут обладать богатым интерфейсом, близким к десктопным приложениям, обеспечивая при этом простоту в использовании.

В целом, веб-приложение обеспечивает более гибкий, доступный и обновляемый способ предоставления функциональности пользователям, что делает его хорошим выбором для создания конкурентоспособного программного решения.

**2.1 Архитектура приложения**

Архитектура "клиент-сервер" предполагает, что процессы предоставления услуг и отправки запросов разделены между разными компьютерами в сети. Каждый из этих компьютеров выполняет свои задачи независимо. В этой концепции несколько клиентских компьютеров, находящихся в удаленных местах, отправляют запросы и получают услуги от централизованной серверной машины, которую также можно назвать хост-системой.



Архитектура «клиент-сервер»

Клиент-серверная архитектура была выбрана в связи с ее способностью эффективно разделять обязанности между клиентской и серверной частями системы. Преимущества такого подхода включают в себя легкость масштабирования, централизованное управление данными и возможность обеспечения высокой степени гибкости при разработке.

Функциональность клиента: Клиентская часть приложения ответственна за взаимодействие с пользователями. Это включает в себя предоставление пользовательского интерфейса, обработку пользовательских запросов и передачу данных на сервер для обработки.

Функциональность сервера: Серверная часть занимается выполнением бизнес-логики приложения, управлением базой данных, аутентификацией и авторизацией. Она обеспечивает централизованное хранение данных и координацию работы множества клиентов.

Преимущества клиент-серверной архитектуры:

- Разделение обязанностей: Клиент и сервер выполняют разные функции, что облегчает разработку, тестирование и обслуживание системы.

- Централизация данных: Централизованное хранение данных на сервере обеспечивает единый источник правды, облегчает управление данными и поддерживает их целостность.

- Масштабируемость: Гибкость масштабирования позволяет системе эффективно адаптироваться к изменяющимся нагрузкам путем добавления новых клиентов или серверов.

- Прозрачность: Взаимодействие между клиентом и сервером может быть скрыто от пользователя, обеспечивая прозрачность внутренних механизмов работы системы.

- Безопасность: Механизмы безопасности, такие как шифрование данных и аутентификация, могут быть легко реализованы и контролируются централизованно на сервере.

- Гибкость и обновляемость: Независимость клиентов и серверов позволяет легко вносить изменения в одну из частей системы без необходимости изменения другой.

Недостатки клиент-серверной архитектуры:

- Точка отказа: Если сервер выходит из строя, вся система может стать недоступной для клиентов, что делает архитектуру уязвимой к отказам сервера.

- Зависимость от сети: Работоспособность клиентов может зависеть от стабильности и пропускной способности сети, что может стать проблемой в условиях низкой связанности.

- Сложность обеспечения согласованности данных: Централизованное хранение данных может привести к сложностям в обеспечении согласованности данных при параллельных изменениях.

- Сложность масштабирования баз данных: При увеличении числа клиентов и объема данных может возникнуть необходимость в сложных мероприятиях для масштабирования базы данных.

- Производительность: Загруженность сервера может быть узким местом, особенно в случае большого числа активных клиентов, что влияет на производительность системы.

- Сложность обновлений на клиентской стороне: Изменения в логике приложения на клиентской стороне могут потребовать обновлений на каждом устройстве клиента.

Клиент-серверная архитектура предоставляет эффективный способ построения распределенных систем, однако требует внимательного управления и адаптации к конкретным требованиям и ограничениям проекта. Эффективное использование этой архитектуры требует сбалансированности между ее преимуществами и недостатками в соответствии с конкретными потребностями и целями разрабатываемой системы.

В будущем сервер-клиент архитектура продолжит играть ключевую роль в разработке программного обеспечения, поскольку она обеспечивает гибкость, масштабируемость и удобство взаимодействия между компонентами системы. С постоянным развитием технологий и изменением требований рынка, можно ожидать следующих тенденций в развитии этой архитектуры:

Рост микросервисной архитектуры: С развитием микросервисной архитектуры, сервер-клиент взаимодействие станет более модульным и децентрализованным. Компании будут предпочитать разбивать свои системы на небольшие, автономные сервисы, что способствует более гибкой разработке, развертыванию и масштабированию приложений.

Развитие протоколов и стандартов: Стандарты и протоколы взаимодействия между сервером и клиентом будут постоянно совершенствоваться. Это включает в себя улучшение протоколов передачи данных, таких как HTTP/2 и HTTP/3, а также стандартизацию протоколов авторизации и аутентификации, обеспечивая повышенную безопасность и производительность.

Увеличение использования облачных вычислений: С ростом облачных технологий и услуг, сервер-клиент архитектура будет все больше ориентироваться на облачные вычисления. Это обеспечит более высокую отказоустойчивость, масштабируемость и доступность приложений за счет использования облачных платформ для размещения серверных компонентов.

Использование новых технологий: С развитием технологий, таких как искусственный интеллект, интернет вещей (IoT) и блокчейн, сервер-клиент архитектура будет интегрировать эти инновации для создания более интеллектуальных и ресурсоэффективных приложений. Например, использование алгоритмов машинного обучения на сервере для анализа данных клиентов или интеграция с устройствами IoT для сбора и обработки данных в реальном времени.

Улучшение пользовательского опыта: С развитием веб-технологий, таких как веб-сокеты и серверные события, а также ростом мобильных приложений, сервер-клиент архитектура будет все более фокусироваться на обеспечении высокопроизводительного и реактивного пользовательского опыта. Это включает в себя использование техник кэширования данных на клиентской стороне, предварительной загрузки и отложенной загрузки ресурсов для ускорения работы приложений.

В целом, сервер-клиент архитектура будет продолжать эволюционировать, чтобы соответствовать требованиям современного программного обеспечения, обеспечивая высокую производительность, безопасность и масштабируемость приложений в различных сферах применения.

**2.2 Структура программного обеспечения**

Взаимодействие между клиентом и сервером в клиент-серверной архитектуре осуществляется с использованием различных протоколов и технологий, которые играют ключевую роль в обмене данными и обеспечивают эффективную коммуникацию между компонентами системы.

**HTTP**, как протокол прикладного уровня, служит стандартным средством обмена данными в веб-приложениях. Он основан на концепции "запрос-ответ", где клиент инициирует запрос на сервер, а сервер отвечает передачей данных обратно клиенту. Этот протокол широко применяется для загрузки веб-страниц, изображений и других ресурсов, а также для создания RESTful API, предоставляя стандартизированный интерфейс для взаимодействия.

**WebSocket** представляет собой технологию, обеспечивающую полнодуплексное взаимодействие между клиентом и сервером. Его особенность заключается в возможности эффективного обмена данными в режиме реального времени. Одно установленное соединение позволяет отправлять данные как от клиента к серверу, так и наоборот. WebSocket часто используется в разработке чатов, мессенджеров и других приложений, где требуется мгновенная передача информации.

**RESTful API** представляет собой подход к взаимодействию между клиентом и сервером, основанный на представлении данных в виде ресурсов с уникальными идентификаторами (URI). Этот подход поддерживает ограниченный интерфейс и четко определенные методы запросов, такие как GET, POST, PUT, DELETE. RESTful API широко применяется для создания веб-сервисов и взаимодействия с данными в веб-приложениях.

**Node.js** - это среда выполнения JavaScript, основанная на движке V8 Chrome, которая позволяет выполнять JavaScript на серверной стороне приложения. Одним из главных преимуществ Node.js является его асинхронная и событийно-ориентированная архитектура, которая позволяет эффективно обрабатывать большие объемы запросов без блокировки потоков, что обеспечивает высокую производительность и отзывчивость приложения.

В сочетании с фреймворком Express.js, Node.js становится мощным инструментом для разработки веб-приложений. Express.js предоставляет удобный и гибкий способ создания веб-серверов и обработки HTTP запросов. С его помощью можно легко определять маршруты, обрабатывать запросы, управлять сессиями и многое другое. Это упрощает разработку веб-приложений и позволяет сосредоточиться на бизнес-логике приложения, минимизируя издержки на инфраструктурные задачи.

Одним из ключевых преимуществ использования Node.js и Express.js является возможность разработки приложений на JavaScript как на клиентской, так и на серверной стороне. Это упрощает обмен кодом и легко масштабируется, что ускоряет разработку и упрощает поддержку приложения.

Кроме того, использование Node.js и Express.js обеспечивает нам большую гибкость и возможность интеграции с различными сторонними сервисами и библиотеками благодаря обширной экосистеме Node.js. Мы можем легко использовать тысячи пакетов npm для добавления дополнительной функциональности, такой как аутентификация, авторизация, работа с базами данных и многое другое, что позволяет нам быстро расширять возможности нашего приложения и следовать лучшим практикам разработки.

Таким образом, выбор Node.js и Express.js для серверной стороны нашего приложения обеспечивает нам высокую производительность, гибкость, простоту разработки и расширения, что позволяет нам эффективно создавать и поддерживать мощные веб-приложения.

**MongoDB** - это NoSQL база данных, которую мы выбрали для нашего приложения, интегрированную через Mongoose. Она отличается гибкостью в хранении и обработке данных. В отличие от традиционных реляционных баз данных, MongoDB использует гибкие JSON-подобные документы для хранения данных, что позволяет нам эффективно моделировать сложные структуры данных и легко изменять их по мере развития приложения.

Интеграция с MongoDB через Mongoose обеспечивает нам удобный и выразительный способ взаимодействия с базой данных. Mongoose - это объектно-документный отображатель (ODM) для MongoDB, который предоставляет нам возможность определять схемы данных, создавать модели и выполнять запросы к базе данных с помощью простого и понятного API. Это упрощает разработку и поддержку базы данных, а также обеспечивает защиту от ошибок и нежелательных изменений данных.

Выбор MongoDB и Mongoose обеспечивает нам гибкость в работе с данными. Мы можем легко изменять структуру данных и добавлять новые поля или коллекции по мере необходимости, что позволяет нам быстро реагировать на изменения требований бизнеса или пользовательских запросов.

Кроме того, MongoDB хорошо масштабируется горизонтально, что означает, что мы можем легко увеличивать производительность и доступность базы данных путем добавления новых узлов кластера MongoDB. Это позволяет нам обеспечивать высокую производительность и надежность нашего приложения даже при увеличении его нагрузки или объема данных.

Таким образом, выбор MongoDB и Mongoose для нашего приложения обеспечивает нам гибкость, простоту разработки и масштабируемость базы данных, что позволяет нам эффективно управлять данными и обеспечивать высокую производительность нашего приложения.

На клиентской стороне приложения мы используем ванильный **JavaScript**, **HTML** (включая EJS для серверного рендеринга) и **CSS**. Этот стек технологий обеспечивает удобное взаимодействие пользователя с приложением, а также обеспечивает легкость разработки и поддержки интерфейса. JavaScript является основным языком программирования для взаимодействия с пользователем и обработки динамического контента на стороне клиента. HTML используется для структурирования содержимого страницы, а CSS - для оформления и стилизации элементов интерфейса.

Использование EJS для серверного рендеринга HTML позволяет нам создавать динамические страницы на стороне сервера, что улучшает производительность и обеспечивает более быстрое отображение контента для пользователей. Это также упрощает поддержку и обновление интерфейса, так как изменения могут быть внесены централизованно на сервере без необходимости внесения изменений в каждую страницу вручную.

В результате мы получаем мощный и эффективный стек технологий, который позволяет нам создавать пользовательский интерфейс с высокой степенью интерактивности и отзывчивости. Это способствует повышению удовлетворенности пользователей и обеспечивает более плавный и приятный опыт использования нашего приложения.

Проект включает использование **GitHub** для эффективного управления версиями кода и обеспечивает удобное отслеживание изменений. GitHub - это популярная платформа для хостинга и совместной работы над проектами с использованием системы контроля версий Git. Наше использование GitHub позволяет нам:

Управлять версиями кода: Мы можем легко создавать новые версии кода, фиксировать изменения и отслеживать историю изменений с помощью системы контроля версий Git. Это обеспечивает нам контроль над эволюцией нашего приложения и упрощает возврат к предыдущим версиям кода в случае необходимости.

Совместную разработку: GitHub предоставляет инструменты для совместной работы над проектом несколькими разработчиками. Мы можем легко создавать ветки для новых функций или исправлений ошибок, проводить обзоры кода и слияние изменений, что способствует эффективной и организованной разработке.

Отслеживание задач и ошибок: Мы можем использовать GitHub Issues для управления задачами, отслеживания ошибок и запросов на улучшение функционала. Это обеспечивает централизованное место для общения и совместной работы по улучшению нашего приложения.

Документация и вики-страницы: GitHub позволяет нам создавать и поддерживать документацию и вики-страницы для нашего проекта. Это помогает нам документировать процессы, инструкции по установке и использованию приложения, а также предоставлять полезную информацию для участников проекта и конечных пользователей.

Интеграция с CI/CD: Мы можем интегрировать GitHub с системами непрерывной интеграции и доставки (CI/CD), чтобы автоматизировать процессы сборки, тестирования и развертывания приложения. Это помогает нам обеспечивать высокое качество кода и быструю поставку изменений в продуктивную среду.

Таким образом, использование GitHub обеспечивает нам удобный и эффективный способ управления версиями кода, совместной работы над проектом и отслеживания изменений, что способствует повышению производительности и качества нашего приложения.

**Роли в системе:**

Администратор (admin): Обладает полным доступом к системе. Может создавать новых пользователей, добавлять статьи по технике безопасности, а также управлять базой данных. Авторизация администратора происходит через страницу входа.

Пользователь (user): Имеет доступ к чтению статей, прохождению тестов и взаимодействию с материалами в системе. Для безопасности и предотвращения несанкционированных аккаунтов, создание новых пользователей осуществляется только администратором.

**Структура базы данных:**

Таблица "articles": Хранит статьи по технике безопасности, их содержание и дополнительные атрибуты.

Таблица "users": Содержит информацию о пользователях, их учетные данные и статус активации.

**Процесс авторизации и безопасности:**

* Пользователи вводят свои учетные данные на странице авторизации (логин и пароль).
* Создание новых аккаунтов осуществляется исключительно администратором для обеспечения безопасности.
* После создания аккаунта пользователю необходимо активировать его, следуя инструкциям, отправленным на указанный им адрес электронной почты.
* Это обеспечивает защиту от нежелательных аккаунтов и уменьшает риск DDoS-атак.

**2.3 Создание программного интерфейса**

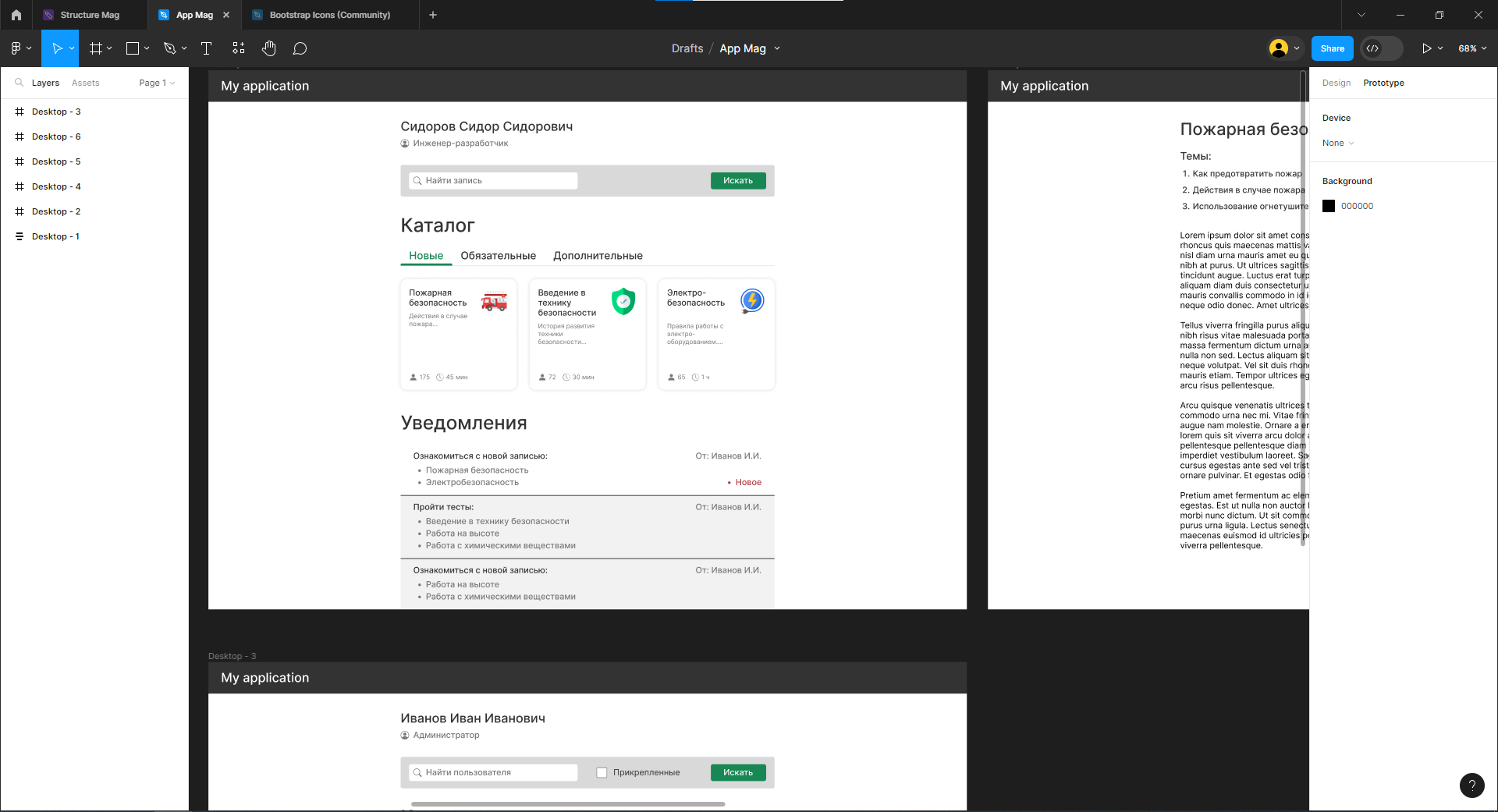
В процессе создания первоначального дизайна программы был применён инструмент - Figma. На нем были разработаны интерфейс и проработано удобство использования программного обеспечения.

Процесс разработки макета включал в себя 3 этапа:

- Дизайн модели - идеи того как будет выглядеть программа

- Навигация - переходы между страницами

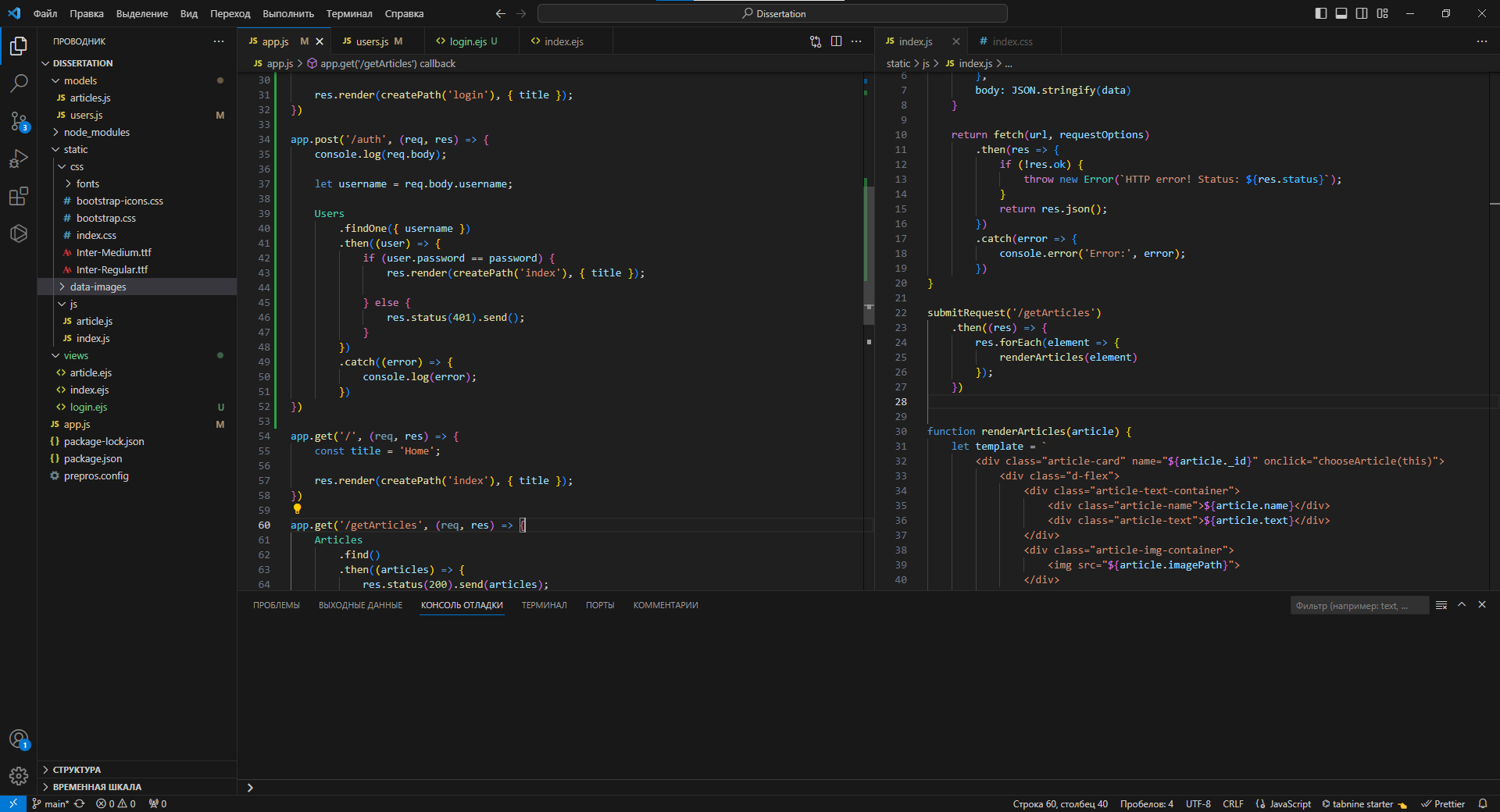
- Работа над эргономикой - поиск лучшего размещения элементов управления и прочего контента

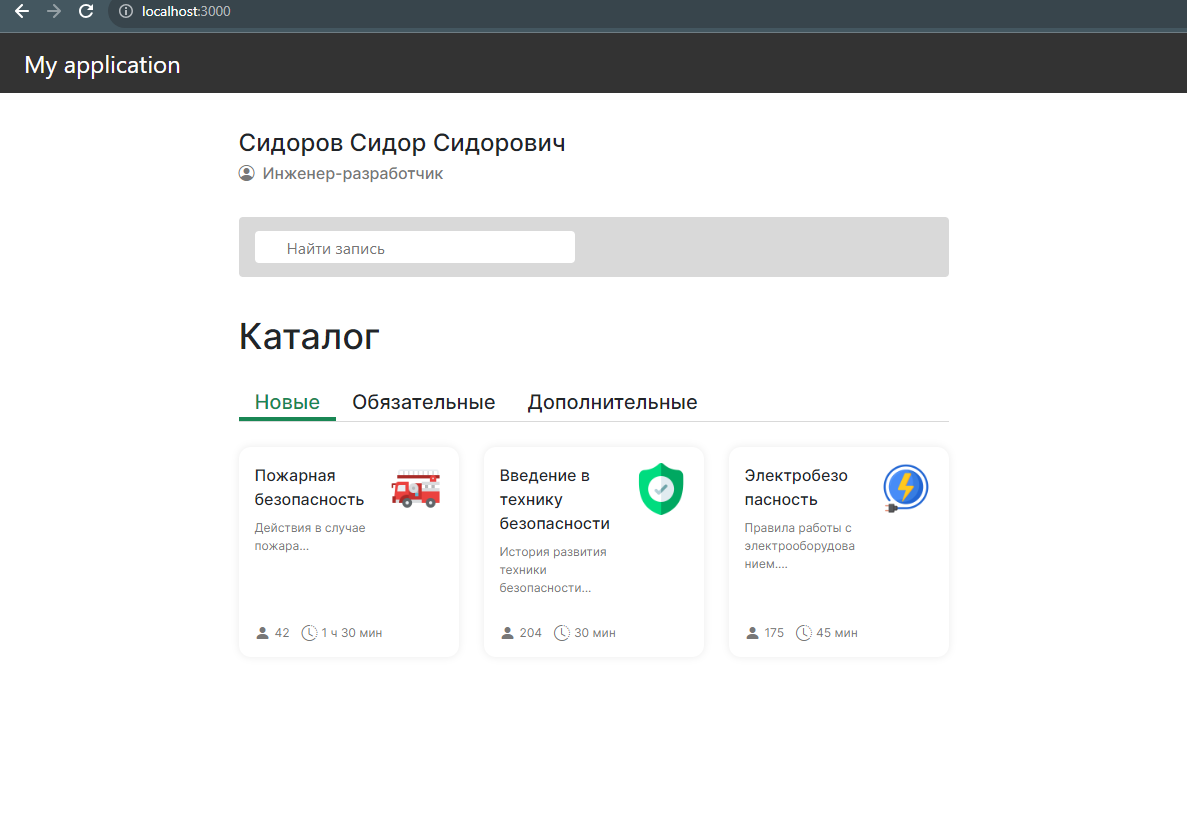


Проектирование пользовательского интерфейса в Figma предоставило обширные возможности для визуализации и оптимизации пользовательского опыта. После завершения этапа дизайна, перешел к разработке программного обеспечения в интегрированной среде разработки Visual Studio Code.

Разработка на языках программирования HTML, CSS, и JavaScript:

Созданный пользовательский интерфейс был воплощен с использованием технологий веб-разработки. Я выбрал использование языков программирования HTML, CSS и JavaScript для построения динамического и отзывчивого интерфейса. HTML обеспечивает структуру страницы, CSS - стилизацию, а JavaScript - взаимодействие с пользователем.



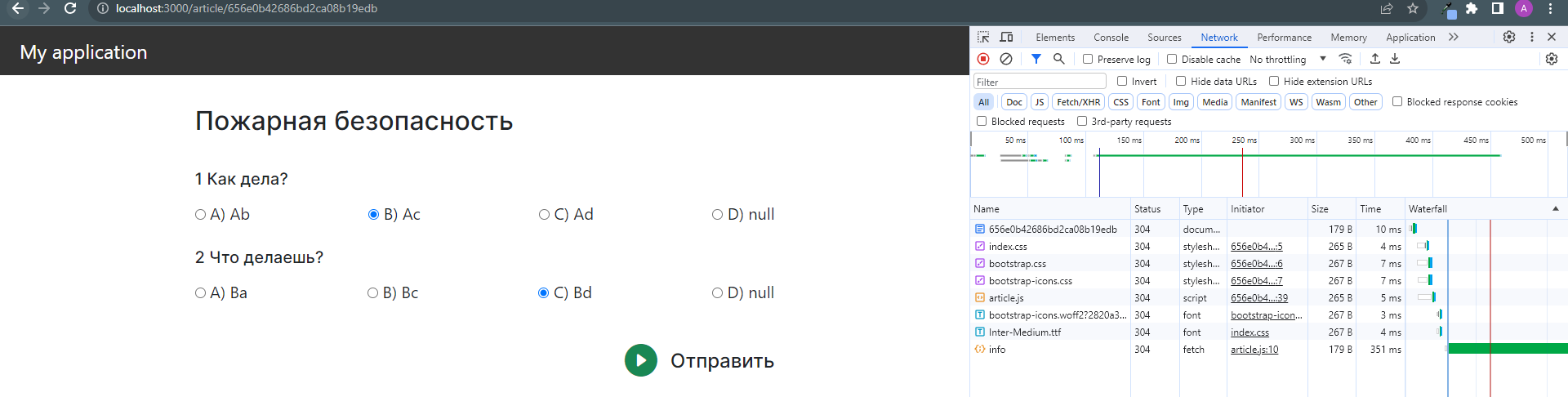


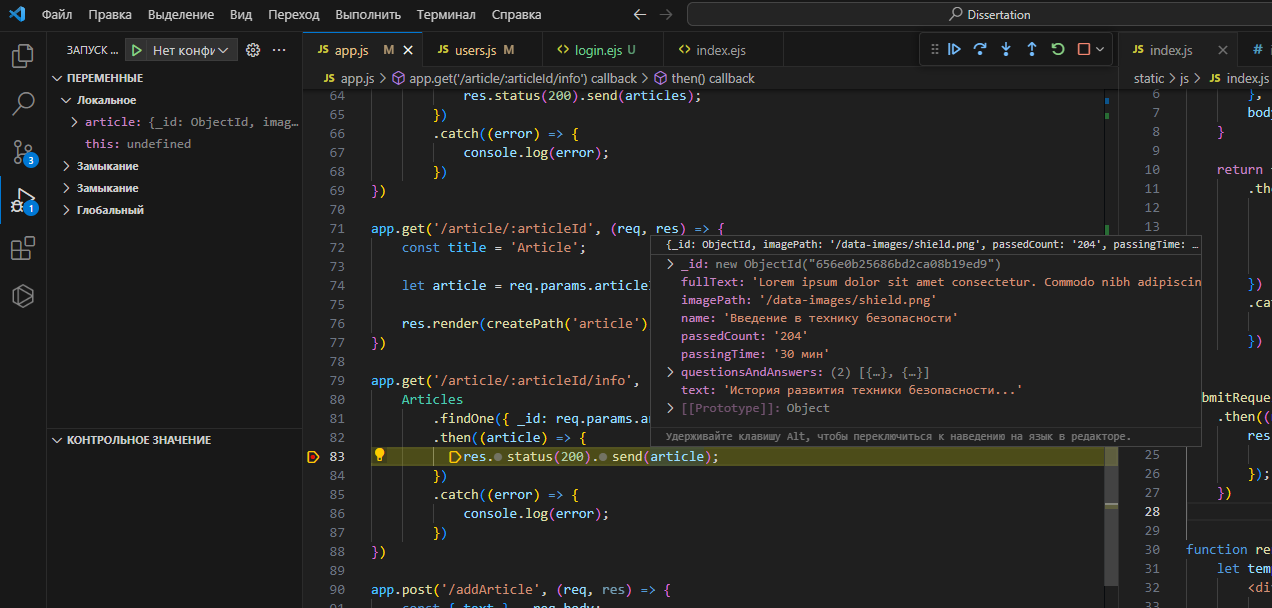
Вид главной страницы

Реализация логики приложения с использованием JavaScript и Node.js:

JavaScript (Vanilla) был выбран для реализации логики приложения на стороне клиента. Это обеспечило взаимодействие с элементами интерфейса и обработку пользовательских действий. При необходимости были использованы современные подходы и возможности языка JavaScript, обеспечивая высокую производительность.

На стороне сервера был использован Node.js, что позволило создать масштабируемый и эффективный сервер для обработки запросов от клиента. Node.js также обеспечил возможность использования JavaScript на обеих сторонах приложения, что способствует унификации и упрощению процесса разработки.





Добавление различных функций и обработка запросов

Интеграция стилей и адаптация с использованием CSS:

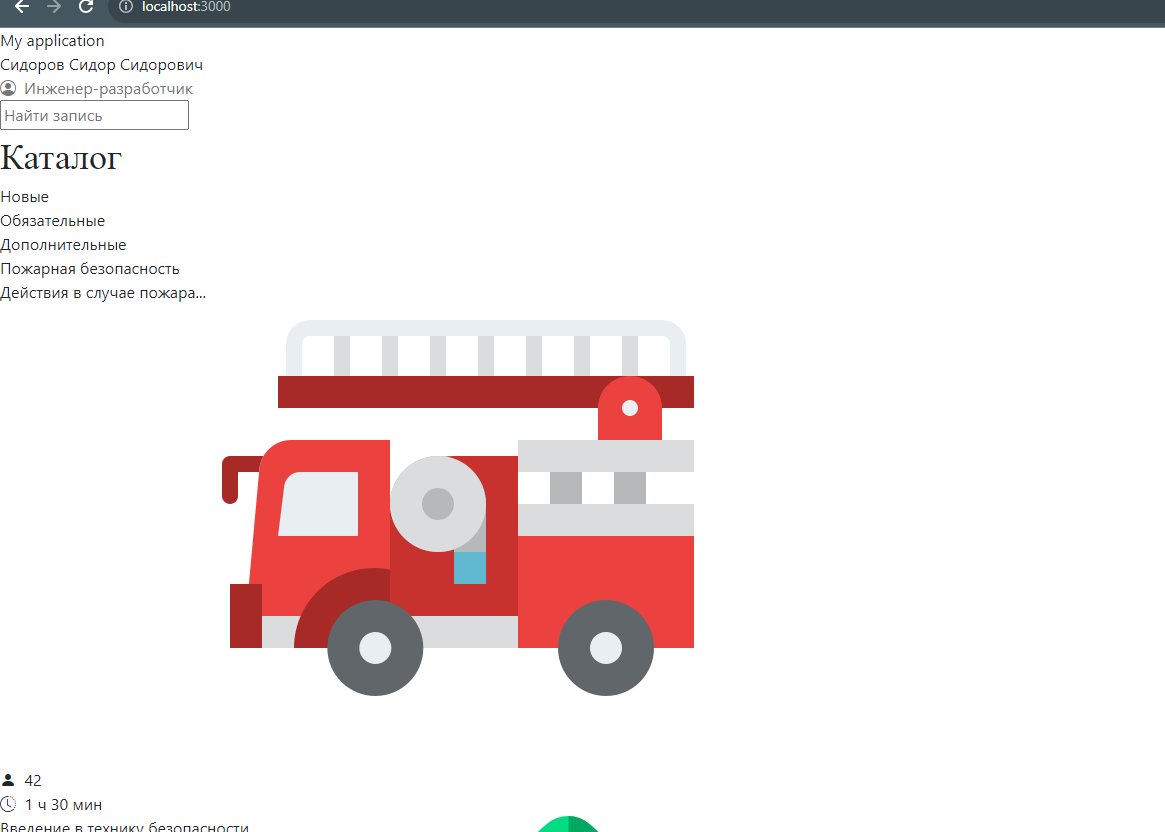
CSS использовался для стилизации элементов интерфейса, обеспечивая их визуальную привлекательность и соответствие дизайну, созданному в Figma. При этом уделялось внимание адаптивности, чтобы приложение выглядело красиво и функционировало корректно на различных устройствах.



Стили элементов

Использование Bootstrap для повышения отзывчивости:

Bootstrap, как фреймворк для веб-разработки, был включен в процесс разработки для улучшения отзывчивости и обеспечения единообразного дизайна на различных устройствах. Компоненты Bootstrap использовались для быстрой и удобной стилизации элементов интерфейса, что сократило время разработки.

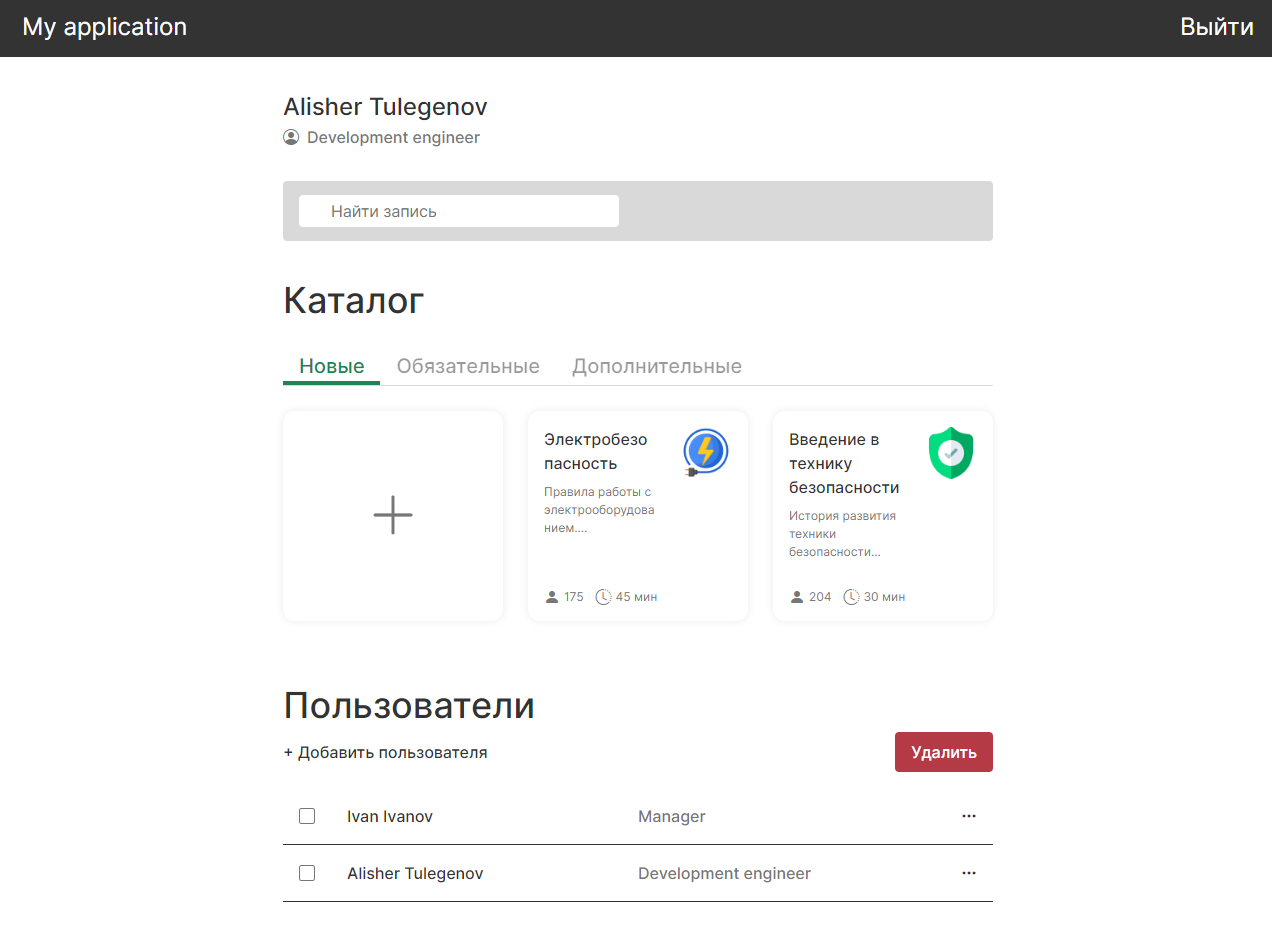


Страница без CSS и Bootstrap

**3 Тестирование и внедрение ПО**

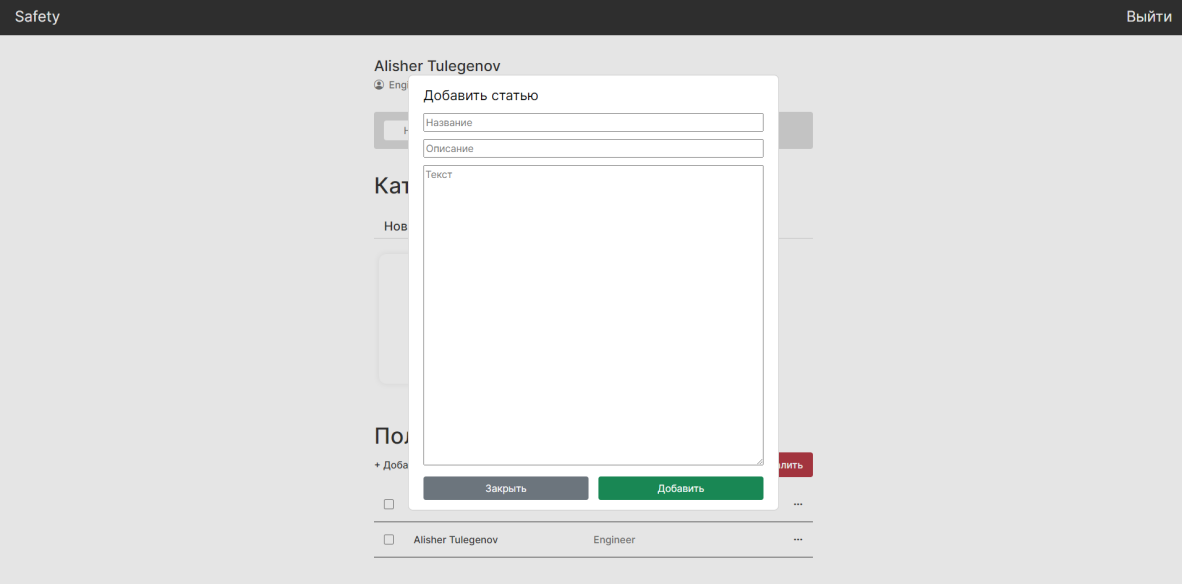
Главная страница приложения:

На главной странице пользователи встречаются с информацией о своем профиле, включая имя и должность. Главным элементом является каталог статей, представленных в виде удобных плиток. Каждая плитка содержит заголовок статьи, краткое описание, количество прочитавших, и оценку времени на прочтение. Пользователи имеют возможность сортировать статьи по разделам: "Новые", "Обязательные" (для пользователей с определенной должностью) и "Дополнительные" (необязательные).



Для администратора доступны дополнительные функции, такие как:

Добавление новых статей: Администратор может создавать новые статьи, используя специальную плитку на главной странице. При этом открывается модальное окно с полями для ввода необходимой информации.

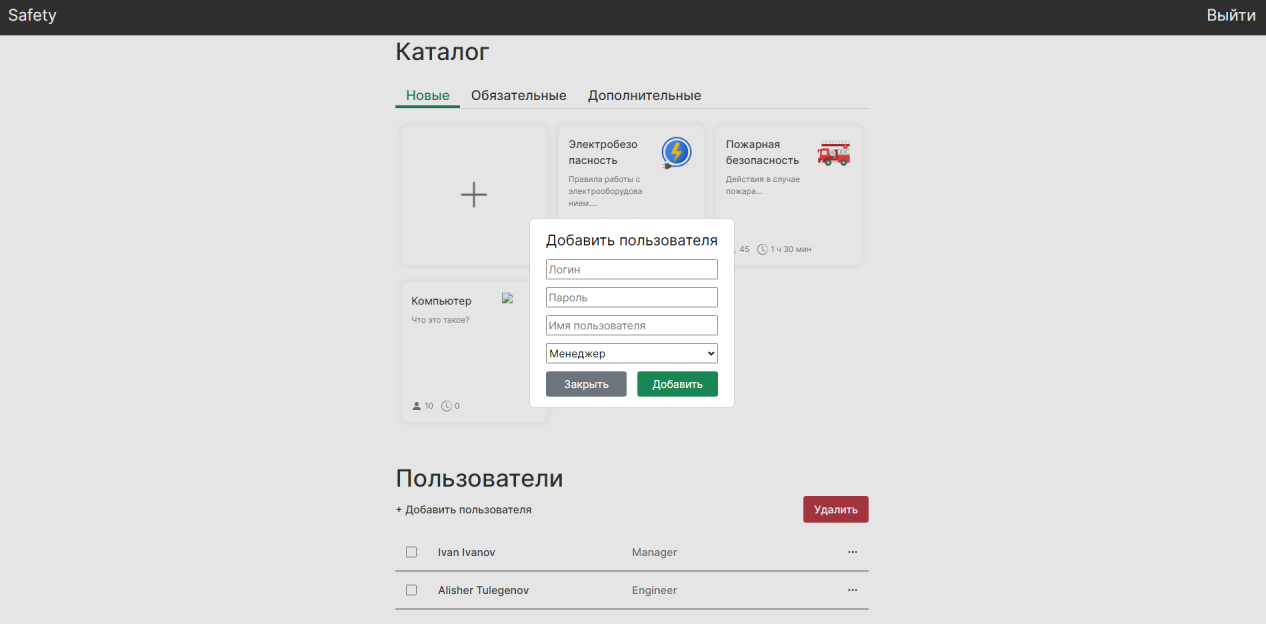


Модальное окно добавления новой статьи

Созданные статьи сразу можно увидеть в «Новые». Внутри карточек можно увидеть их название, краткое описания, количество вошедших в статью пользователей и приблизительное время прохождения.

Управление пользователями: Администратор видит список всех пользователей, включая их имена, должности и, возможно, адреса электронной почты. Имеются кнопки для добавления новых пользователей и удаления существующих.

Ниже секция показывающая всех пользователей. Здесь можно добавить\удалить пользователя.

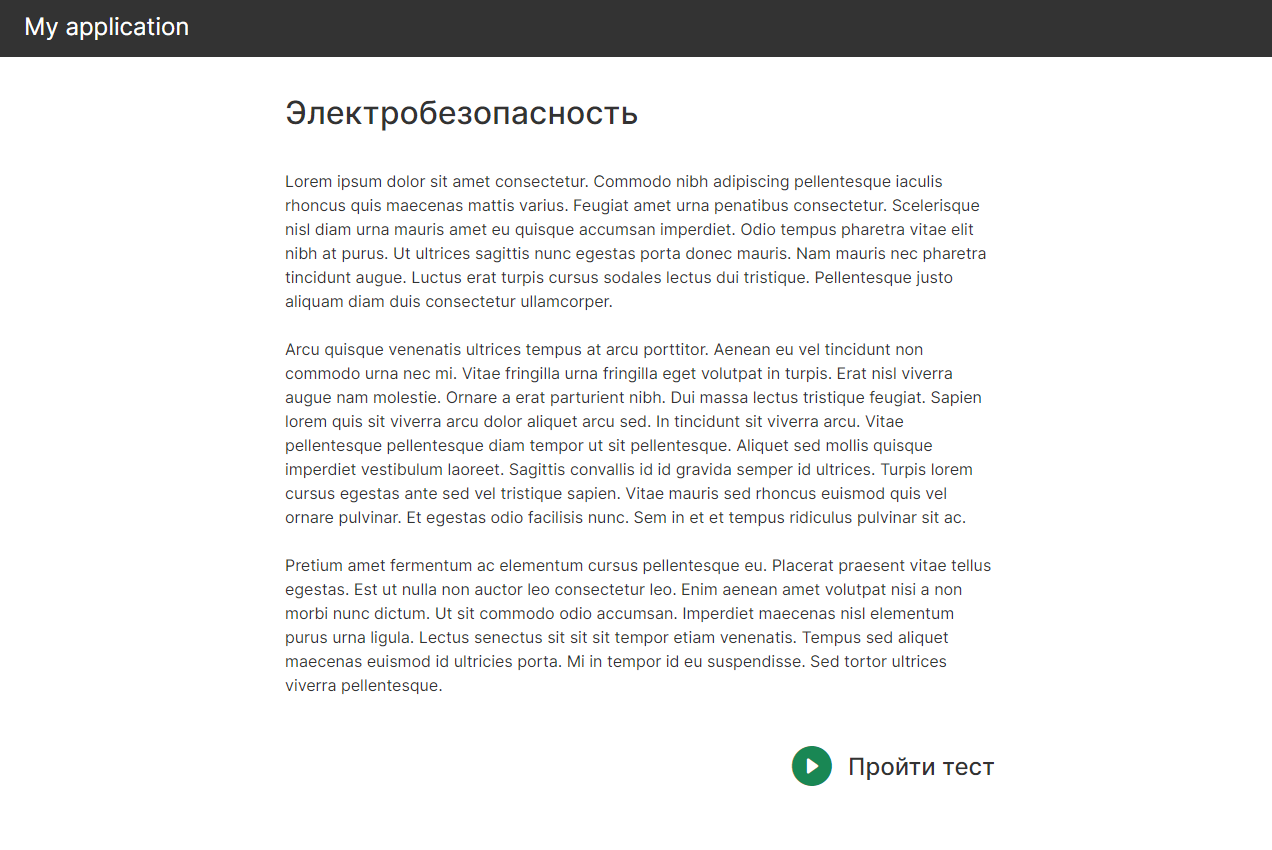


Модальное окно создания новых пользователей

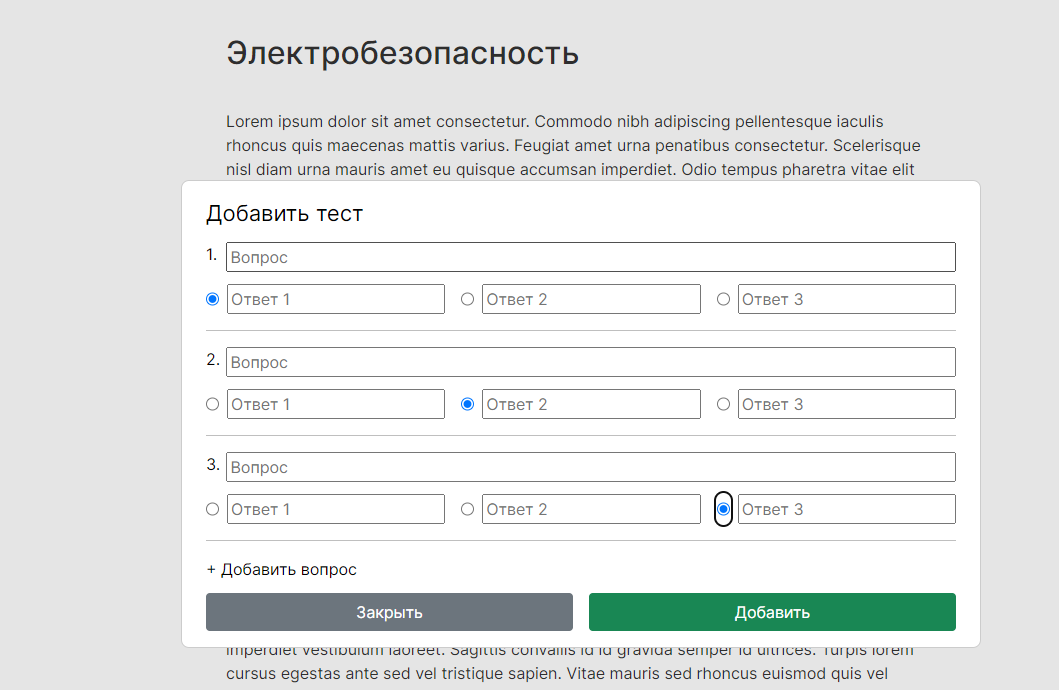
Так же если пользователь проходил тесты по статьям, их можно увидеть нажав на три точки справа.

**Переход на страницы статьи и прохождение тестов:**

После выбора статьи из каталога, пользователь попадает на страницу с развернутым текстовым содержанием статьи. Здесь ему предоставляется возможность ознакомиться с материалом более подробно. В нижней части страницы расположена кнопка, при нажатии на которую открывается доступ к тесту для проверки усвоенных знаний.



Также присутствует кнопка для добавления теста (только у Админа)



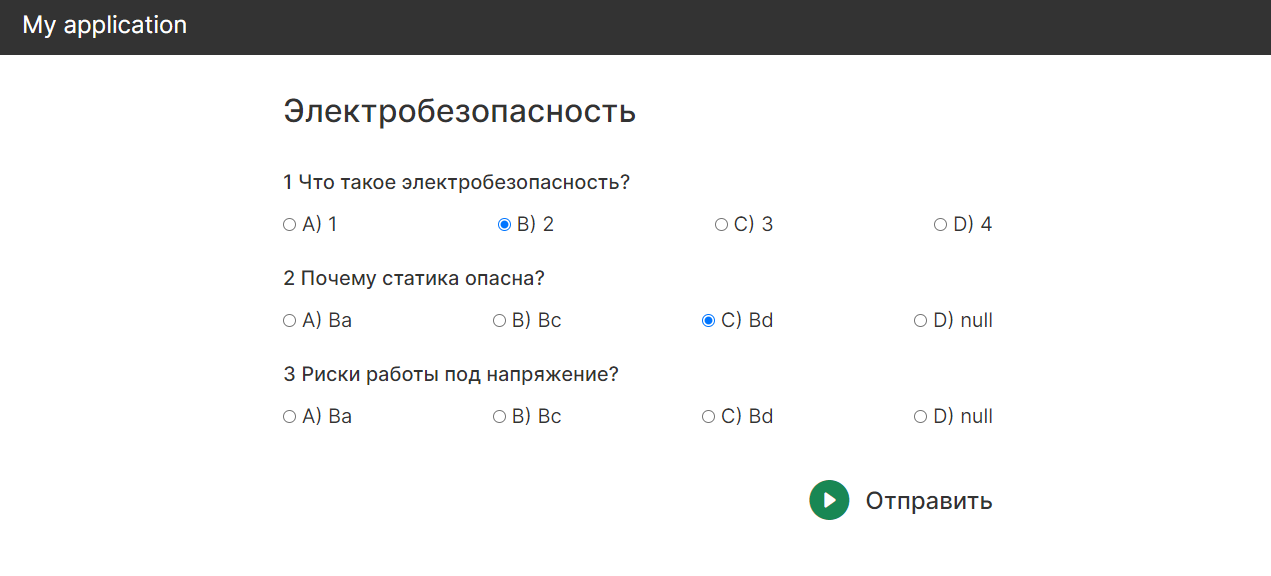
Создания теста для статьи

**Прохождение теста:**

В случае, если к данной статье предусмотрен тест, пользователь перенаправляется на страницу с вопросами.

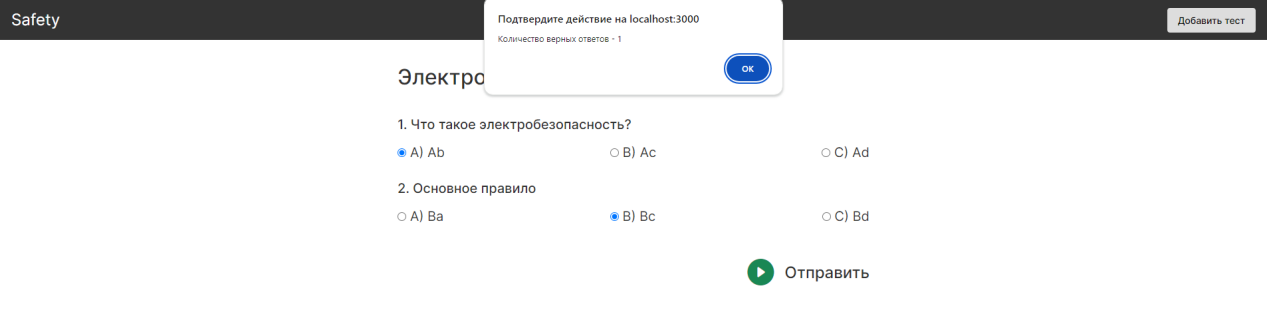
Тест может включать как закрытые, так и открытые вопросы, которые формулируются самими работниками ТБ.

После завершения теста, результаты (количество баллов) записываются в базу данных, а пользователь видит свой отчет.

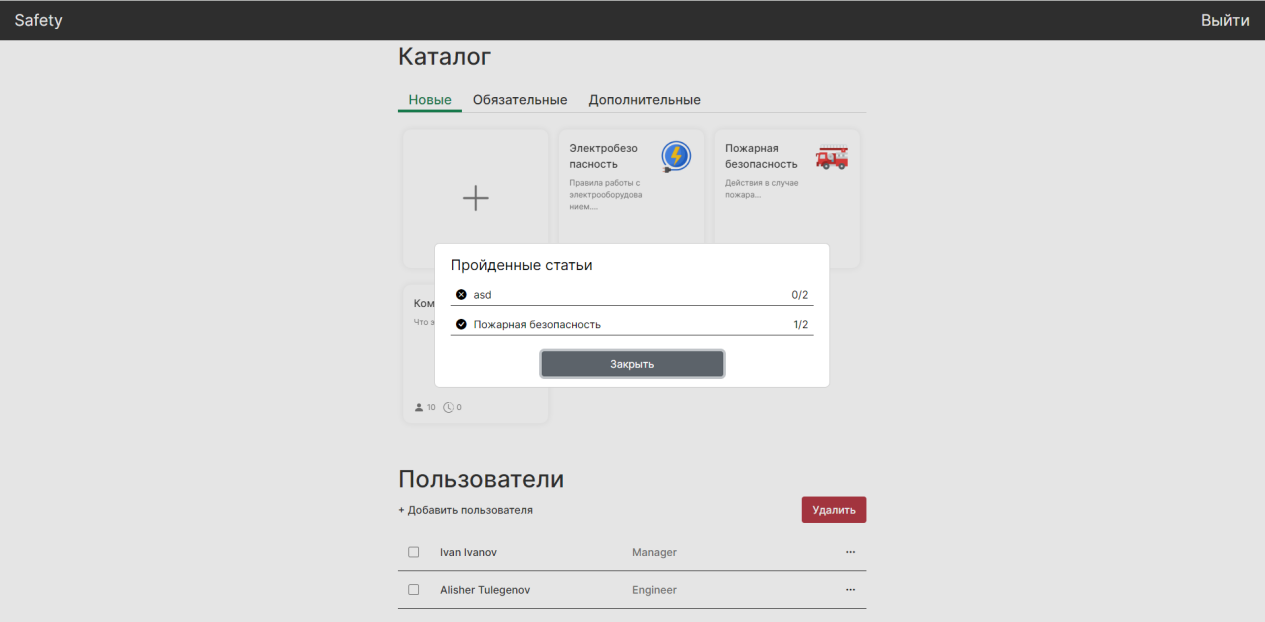


Отправка ответов (если все вопросы закрытые сразу получаем число верных ответов)

**Отчет и просмотр результатов:**



Администратор имеет возможность просматривать результаты тестирования в специальном разделе приложения. Отчет включает в себя количество набранных и общих баллов.



Модальное окно пройденных тестов пользователя

Это обеспечивает прозрачность и отслеживание процесса обучения сотрудников по технике безопасности.

Такой функционал позволяет пользователям эффективно проверять свои знания и предоставляет администратору инструменты для мониторинга успеваемости и обучения персонала.

Заключение

В век развития технологий травматизм и профессиональные заболевания не редкость, поэтому охрана труда должна быть одной из основных задач руководства промышленного предприятия.

Разработка ПО для автоматизации ТБ является неотъемлемой частью современных промышленных процессов. Оно позволяет не только обеспечить безопасность и снизить риски, но и повысить эффективность, оптимизировать производственные процессы и улучшить общий стандарт безопасности в рабочей среде. При правильном подходе к разработке, внедрению и поддержке, такое ПО становится незаменимым инструментом, способствующим процветанию современных промышленных предприятий.

По этой причине стоит обратить внимание на инновационные решения в данной сфере, а именно на автоматизацию задач охраны труда.

Список использованной литературы

1. Федоров А. В., Иванова Е. В. Автоматизация учета технических средств обеспечения безопасности: современные подходы и методы // Материалы X международной научно-практической конференции "Инновационные технологии в науке и образовании". - 2020.
2. Технологии автоматизации журнала техники безопасности: практический опыт и применение / Под ред. Иванова В. Н. - М.: Наука, 2021.
3. Джонсон Дж. Автоматизация процессов управления технической безопасностью на предприятии // Технологии безопасности. - 2019.
4. Техническая безопасность и автоматизация: сборник научных трудов / Под ред. Смирнова В. В. - СПб.: Издательство Политехнического университета, 2017.
5. https://blog.intalev.ru/post/avtomatizirovannyy-kontrol-tekhniki-bezopasnosti-v-stroitelstve/
6. Гринченко Ю. А., Соколов М. В. Применение информационных технологий для автоматизации учета и анализа технической безопасности // Вестник Костромского государственного университета им. Н. А. Некрасова. - 2018.
7. Алексеев А. В. Автоматизированные системы учета технической безопасности на производстве // Труды Международной научной конференции "Информационные технологии в науке, управлении, социальной сфере и медицине". - 2020.
8. https://www.ibm.com/products/maximo
9. https://www.honeywellforge.ai/
10. Automated Safety Management Systems: A Literature Review // Center for Chemical Process Safety. - 2017.
11. Шабанов С. Г., Иванов В. Н., Ломовцев А. А. Применение программных систем для автоматизации учета технической безопасности // Технологии безопасности. - 2022.

https://cyberleninka.ru/article/n/avtomatizatsiya-kompleksa-zadach-sistemy-ohrany-truda-i-tehniki-bezopasnosti-na-promyshlennom-predpriyatii