

Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
Томский государственный университет систем управления и
радиоэлектроники (ТУСУР)
Кафедра экономической математики, информатики и статистики (ЭМИС)

К ЗАЩИТЕ ДОПУСТИТЬ
Заведующий кафедрой ЭМИС
доктор физ.- мат. наук, профессор
_____ И.Г. Боровской
« ____ » _____ 2024 г.

Бакалаврская работа
по направлению подготовки
09.03.02 «Информационные системы и технологии»
профиль «Аналитические информационные системы»

WEB-ПЛАТФОРМА АВТОМАТИЗАЦИИ ПРОЦЕССОВ ОХРАНЫ ТРУДА И ПРОМЫШЛЕННОЙ БЕЗОПАСНОСТИ

Студенты гр.590-1
_____ Г.К. Петров
_____ Б.Б. Джумабаев
« ____ » _____ 2024 г.

Руководители
ст. преподаватель ЭМИС ТУСУР
_____ И.Г. Афанасьева
ст. преподаватель ЭМИС ТУСУР
_____ А.А. Матолыгин
« ____ » _____ 2024 г.

Томск 2024

Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение
высшего образования

Томский государственный университет систем управления и
радиоэлектроники (ТУСУР)

Факультет вычислительных систем (ФВС)

Кафедра экономической математики, информатики и статистики (ЭМИС)

УТВЕРЖДАЮ

Заведующий кафедрой ЭМИС

доктор физ.-мат. наук, профессор

_____ И.Г. Боровской

«_____» _____ 2024 г.

**Индивидуальное задание
на выполнение выпускной квалификационной работы**

студенту гр. 590-1 Петрову Глебу Константиновичу

студенту гр. 590-1 Джумабаеву Богдану Бекполатовичу

1. Тема работы: Web-платформа автоматизации процессов охраны труда и промышленной безопасности
(утверждена приказом от _____ № _____).
2. Цель работы: разработка web-платформы по автоматизации процессов в сфере охраны труда и промышленной безопасности.
3. Содержание работы (перечень вопросов, подлежащих разработке):
 - 3.1. Анализ предметной области_____.
 - 3.2. Проектирование архитектуры приложения_____.
 - 3.3. Разработка серверной части приложения_____.
 - 3.4. Коммуникационные процессы в web-платформе_____.
 - 3.5. Разработка клиентской части приложения_____.
 - 3.6. Бизнес-модель_____.

4. Дата выдачи задания: «_____» _____ 2024 г.

5. Дата сдачи работы на кафедру: «_____» _____ 2024 г.

Руководитель

ст. преподаватель ЭМИС

ТУСУР, И.Г. Афанасьева

(Ф.И.О., должность)

_____ (подпись)

« _____ » _____ 2024 г.

ст. преподаватель ЭМИС

ТУСУР, А.А. Матолыгин

(Ф.И.О., должность)

_____ (подпись)

« _____ » _____ 2024 г.

Задание приняли к исполнению:

Г.К. Петров

(Ф.И.О.)

_____ (подпись)

« _____ » _____ 2024 г.

Б.Б. Джумабаев

(Ф.И.О.)

_____ (подпись)

« _____ » _____ 2024 г.

Реферат

Выпускная квалификационная работа (ВКР) содержит 71 страницу, 15 рисунков, 7 таблиц, 26 источников, 3 приложения.

WEB-ПЛАТФОРМА, ОХРАНА ТРУДА, ДОКУМЕНТЫ, АВТОМАТИЗАЦИЯ ДОКУМЕНТООБОРОТА, ФРЕЙМВОРК REACT, DOCKER, КОНТЕЙНЕР, GOLANG, POSTGRESQL.

Автоматизировать создание документов по охране труда;

Создана система, которая позволяет вести организацию и автоматически создавать документы по охране труда на основе шаблонов.

Объектом исследования является документооборот в сфере охраны труда.

В результате проделанной работы изучено документооборот в сфере охраны труда, реализована Web-Платформа по автоматизации документооборота в сфере охраны труда.

The Abstract

The final qualifying work (FQP) contains 71 pages, 13 figures, 7 tables, 26 sources, 7 applications.

WEB PLATFORM, WEB PLATFORM DEVELOPMENT,
OCCUPATIONAL SAFETY, DOCUMENTS, DOCUMENT FLOW
AUTOMATION.

The object of the study is document flow in the field of labor protection.

As a result of the work done, document flow in the field of labor protection was studied, and a Web platform was implemented to automate document flow in the field of labor protection.

Сокращения, обозначения, термины и определения

В настоящей работе применяют следующие сокращения с соответствующими определениями:

- МСП – малые и средние предприятия — наименование коммерческих предприятий, которые не превышают определенных показателей;
- СИЗ – средства индивидуальной защиты — средства, используемые работником для предотвращения или уменьшения воздействия вредных и опасных производственных факторов, а также для защиты от загрязнения;
- ДСИЗ – дерматологические средства индивидуальной защиты – это средства для защиты кожи от воздействия вредных и опасных производственных факторов;
- ОПр – оценка профессиональных рисков – совокупность мероприятий, которые позволяют выявить опасности и оценить сопутствующие им риски;
- СОТ – специалист по охране труда.
- СУОТ – система управления охраной труда – это комплекс взаимосвязанных правовых, организационных, технических, социально-экономических, санитарно-гигиенических, лечебно-профилактических и иных мер, направленных на обеспечение безопасных и здоровых условий труда;
- ПО – программное обеспечение – программа или множество программ, используемых для управления компьютером;
- СУБД – система управления базами данных – совокупность программных и лингвистических средств общего или специального назначения, обеспечивающих управление созданием и использованием баз данных;
- HTML – HyperText Markup Language – стандартизированный язык гипертекстовой разметки документов для просмотра веб-страниц в браузере;
- API – Application Programming Interface – описание способов взаимодействия одной компьютерной программы с другими;

- SPA – Single Page Application – одностраничное приложение;
- CLI – Command Line Interface – интерфейс командной строки;
- SDL – Simple DirectMedia Layer – кроссплатформенная мультимедийная библиотека, реализующая единый программный интерфейс к графической подсистеме, звуковым устройствам и средствам ввода для широкого спектра платформ;
- DOM – Document Object Model – независящий от платформы и языка программный интерфейс, позволяющий программам и скриптам получить доступ к содержимому HTML-, XHTML- и XML-документов, а также изменять содержимое, структуру и оформление таких документов;
- HTTP – HyperText Transfer Protocol – протокол прикладного уровня передачи данных;
- TTL – Time-to-Live – предельный период времени или число итераций или переходов, которые набор данных может осуществить до своего исчезновения;
- GORM – Go Object-Relational Mapping – библиотека, которая предоставляет удобный интерфейс для взаимодействия с различными базами данных, сохраняя при этом идиоматичность Go.

В настоящей работе применяют следующие сокращения с соответствующими определениями:

- Web – распределённая система, предоставляющая доступ к связанным между собой документам, расположенным на различных компьютерах, подключённых к сети Интернет;
- Unix – семейство операционных систем для компьютеров;
- Unicode – стандарт кодирования символов, включающий в себя знаки почти всех письменных языков мира;
- Go – это компилируемый многопоточный язык программирования от Google с открытым исходным кодом;

- GraphQL – это язык запросов и серверная среда для API с открытым исходным кодом;
- React – JavaScript-библиотека с открытым исходным кодом для разработки пользовательских интерфейсов;
- PostgreSQL – свободная объектно-реляционная система управления базами данных свободная объектно-реляционная система управления базами данных;
- Cookie – небольшой набор данных, отправляемый веб-сервером и хранимый на компьютере пользователя без изменений и какой-либо обработки;
- DevOps – методология автоматизации технологических процессов сборки, настройки и развёртывания ПО;
- FrontEnd – создание структуры гипертекстового документа на основе HTML-разметки, как правило, при использовании таблиц стилей и клиентских сценариев, таким образом, чтобы элементы дизайна выглядели аналогично макету;
- BackEnd – создание внутреннего механизма веб-приложений, который обрабатывает данные, управляет базами данных и обеспечивает их работоспособность.

Оглавление

Введение	11
1 АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ	14
1.1 Цифровизация охраны труда	14
1.2 Документы и нормы, регламентирующие охрану труда.....	14
2 ПРОЕКТИРОВАНИЕ АРХИТЕКТУРЫ ПРИЛОЖЕНИЯ	17
2.1 Выбор архитектуры приложения.....	17
2.2 Разработка базы данных.....	18
2.3 Разработка серверной части приложения	22
2.4 Передача данных в клиентскую часть.....	23
2.5 Разработка архитектурной части приложения.....	24
3 РАЗРАБОТКА СЕРВЕРНОЙ ЧАСТИ ПРИЛОЖЕНИЯ	27
3.1 Разработка проекта.....	27
3.2 Создание документов по шаблону в web-платформе.....	28
3.3 Получение данных из базы данных.....	30
3.4 Логгирование	31
4 КОММУНИКАЦИОННЫЕ ПРОЦЕССЫ В WEB-ПЛАТФОРМЕ.....	32
4.1 GraphQL	32
4.2 Отправка данных с сервера клиенту	33
5 РАЗРАБОТКА КЛИЕНТСКОЙ ЧАСТИ ПРИЛОЖЕНИЯ.....	36
5.1 Фреймворк React	36
5.2 Хранение данных у клиента.....	37
5.3 СУБД Redis	38
5.4 Интерфейс клиентской части приложения	40
5.5 Создание документов в клиентской части	44
6 БИЗНЕС-МОДЕЛЬ.....	47
6.1 Концепция стартап-проекта.....	47
6.1.1 Описание продукта	47
6.1.2 Рынок охраны труда	47

6.2 Анализ рынка конкурентов	49
6.3 Целевые сегменты и ёмкость рынка	51
6.4 Процесс производства. Планирование объёмов производства	52
6.5 Планирование инвестиционных затрат	53
6.6 Расчёт операционной прибыли	55
6.7 Цифровизация охраны труда	57
Заключение	59
Список использованных источников	61
Приложение А (обязательное) Пример инициализации подключения к БД и запрос на получение данных	64
Приложение Б (обязательное) Пример инициализации логгера и его функции	66
Приложение В (обязательное) Результат расчётов прибыли и план работы подчинённых	68

Введение

Обязанности работодателя в сфере охраны труда в России являются ключевым аспектом обеспечения безопасных и здоровых условий труда для работников. По статье 214, трудового кодекса РФ [1] в первую очередь, работодатель обязан создать и поддерживать безопасные условия труда, соответствующие установленным законодательным требованиям. Это включает проведение аттестации рабочих мест, выявление опасных и вредных факторов производственной среды и принятие мер по их устранению или минимизации.

Работодателю требуется организовать и обеспечить систематическое обучение и инструктаж по охране труда для всех работников. Обучение проводится при приеме на работу, а также периодически в процессе трудовой деятельности, чтобы сотрудники были осведомлены о потенциальных рисках и мерах предосторожности.

Кроме того, работодатель обязан проводить обязательные медицинские осмотры работников, особенно тех, кто занят на работах с вредными или опасными условиями труда. Эти осмотры необходимы для своевременного выявления профессиональных заболеваний и предотвращения их развития.

К требованиям безопасности труда относится обеспечение работников средствами индивидуальной и коллективной защиты. Средства индивидуальной защиты (СИЗ) включают в себя специальную одежду, обувь, защитные маски, очки и другие средства, необходимые для защиты от вредных факторов. Средства коллективной защиты включают в себя системы вентиляции, ограждения, сигнализацию и другие технические устройства, направленные на снижение уровня профессиональных рисков.

Не менее важной обязанностью работодателя является проведение регулярного контроля за соблюдением норм и правил охраны труда. Это включает в себя организацию внутренних проверок и аудит состояния охраны труда, а также взаимодействие с государственными инспекциями труда,

которые осуществляют надзор за соблюдением законодательства в данной сфере.

Согласно законодательству, требуется вести учет и расследование несчастных случаев на производстве и профессиональных заболеваний. Все случаи травматизма должны быть тщательно расследованы, причины установлены, а результаты расследований оформлены документально. По итогам расследований должны быть разработаны и реализованы мероприятия, направленные на предотвращение подобных инцидентов в будущем.

Обязанности работодателя в сфере охраны труда охватывают широкий спектр мер и действий, направленных на обеспечение безопасности и здоровья работников, что способствует не только соблюдению законодательных требований, но и улучшению общей производственной культуры и эффективности труда.

Введение автоматизации рутинной работы становится все более востребованным в различных сферах деятельности. Это подтверждается быстрым ростом технологического прогресса по всему миру, а также активным переходом различных государственных и частных предприятий на автоматизированные системы [2]. Ручная обработка документов требует значительных ресурсов и времени, что снижает общую эффективность работы. Внедрение автоматизированных систем обработки документов является необходимым шагом для оптимизации бизнес-процессов предприятия.

Сотруднику охраны труда следует оценивать и проверять рабочие места сотрудников, поэтому приложение, автоматизирующее создание документов по охране труда должно быть мобильным и быть в доступе из любой точки страны.

Целью данной работы является разработка web-платформы по автоматизации процессов в сфере охраны труда и промышленной безопасности. Для достижения этой цели следует выполнить ряд задач:

- провести анализ предметной области цифровизации охраны труда;

- провести сравнительный анализ готовых решений автоматизации системы охраны труда;
- сформировать требования к разрабатываемой web-платформе;
- разработка клиентской и серверной частей web-платформы;
- оценка рентабельности разрабатываемой web-платформе.

1 АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

1.1 Цифровизация охраны труда

В охране труда и организации безопасных условий труда в настоящее время существует множество проблем, требующих решения. Одной из таких проблем является цифровизация охраны труда.

Она заключается в том, что многие организации всё ещё используют устаревшие методы и инструменты для управления и контроля условиями труда на своём производстве. Это вызывает ряд проблем, как, например, неэффективное использование времени и ресурсов, недостаточный контроль за соблюдением правил и стандартов безопасности.

С развитием технологий информационной безопасности важно иметь автоматизированные системы в области охраны труда. Это позволяет эффективно управлять всеми процессами, связанными с охраной труда, что подтверждается различными исследованиями [3].

1.2 Документы и нормы, регламентирующие охрану труда

Для того, чтобы эффективно вести своё предприятие, соблюдать условия промышленного и технологического развития, следует также обратить внимание на документы отчётности. Они играют важную роль в обеспечении безопасности работников, а значит помогают в развитии предприятия.

В перечень обязанностей работодателя по соблюдению правил и норм охраны труда при работе входят следующие пункты [4, 5]:

- обеспечение безопасности при эксплуатации зданий и сооружений, технического оборудования, а также при осуществлении технологических процессов и применении инструментов или ресурсов;

- выдача средств индивидуальной защиты – работодатель обязан обеспечить персонал всеми средствами защиты, приобретать их за счет организации и выдавать, заменять в соответствии с нормативами;

- создание безопасных условий деятельности. Это значит, что рабочие места должны быть оборудованы в соответствии со всеми требованиями, сертифицированы в общем порядке;

- обучение безопасным приемам и методам работы – на рабочем месте проводят стажировку с последующей проверкой знаний техники безопасности;

- контролировать безопасные условия – работодатель проводит контроль условий труда с определённой периодичностью. При этом проверяется не только уровень безопасности, но также правильность применения средств индивидуальной защиты;

- организация медицинского контроля – медицинские обследования проводятся для всех работников за счет работодателя с определённой периодичностью или же по мере необходимости.

- соблюдение действующего законодательства – работодатель обязан соблюдать все требования и нормы действующего законодательства.

По мере усложнения и развития труда возрастает опасность работы на производстве. Это может быть обусловлено различными факторами, например:

- увеличение количества и сложности используемого оборудования;
- повышение требований к квалификации работников;
- изменения в производственных процессах.

Эти опасности могут иметь серьезные последствия на здоровье работников, а также для самого производства.

Статья 214 ТК РФ посвящена обязанностям работодателя [6]. Согласно ей работодатель обязан обеспечить безопасность условий и охраны труда, а также должен создавать безопасные условия труда исходя как из оценки

технического и организационного уровня рабочего места, так и из оценки факторов производственной среды и трудового процесса.

Помимо прочего работодатели также должны выполнять следующие задачи – систематическое выявление опасностей и профессиональных рисков, их анализ и оценка. В соответствии с проанализированными данными работодателю следует обновлять условия труда, чтобы обеспечить безопасные условия труда на предприятии.

Из-за масштабов предприятия и большого количества сотрудников возникает необходимость в обработке больших данных и обработке большого количества документов – так на каждого сотрудника следует составить значительное количество документов, которые, к тому же, должны со временем обновляться. Одним из важных аспектов здесь является обеспечение безопасности труда сотрудников. В рамках нормативных документов и трудового кодекса определены процедуры по определению и управлению рисками, связанными с рабочей деятельностью.

С ростом количества информации, которая нужна для поддержания безопасности и соблюдения нормативных требований, возникает необходимость в какой-либо системе для упрощения и ускорения процесса создания и управления документами, в том числе документами, связанными с охраной труда. Такая система должна вести и обрабатывать данные сотрудников, составлять отчёты, а также автоматизировать процессы составления документов, предоставляя шаблоны и инструменты для их заполнения. Также такая система должна обеспечивать централизованное хранение и управление информацией, что может сделать процесс управления документами более удобным, а также эффективным.

2 ПРОЕКТИРОВАНИЕ АРХИТЕКТУРЫ ПРИЛОЖЕНИЯ

2.1 Выбор архитектуры приложения

Решением проблемы упрощения и ускорения работы с нормативными документами может быть создание клиент-серверного приложения для управления документацией, связанной с охраной труда. Такое приложение предоставляло бы доступ к документам, а также автоматизацию множества процессов, связанных с охраной труда.

Помимо прочего, приложение обеспечило бы улучшенную безопасность данных, а также предоставило бы большой функционал, адаптированный для нужд этой сферы деятельности. Исходя из этого можно сделать вывод, что создание и последующее использование такого приложения – верный шаг для оптимизации и усовершенствования управления информацией в области охраны труда.

Клиент-серверное приложение для управления документацией по охране труда может предоставить следующий функционал:

1. Централизованный доступ – это значит, что пользователи могут получать доступ к документам при наличии интернета, что делает их более доступными и удобными для использования;

2. Автоматизация процессов – это значит, что приложение может автоматизировать множество процессов, например генерацию отчетов или даже анализ данных;

3. Обеспечение безопасности – это значит, что приложение может обеспечивать более надежную защиту данных путем использования системы аутентификации и авторизации, шифровании данных;

4. Масштабирование функциональности – приложение будет разработано специально для нужд охраны труда.

Исходя из этого можно сделать вывод, что использование клиент-серверного приложения для ведения документации по охране труда может сделать этот процесс более эффективным, удобным и безопасным.

Серверная часть веб-приложения представляет собой программное обеспечение (ПО), которое исполняется на сервере и обрабатывает запросы от клиентской части. При это оно выполняет различные функции и взаимодействует с базой данных (БД) или другими внешними сервисами. Серверная часть приложения выполняет функции аутентификации и авторизации пользователя, обрабатывает бизнес-логику, заполняет документы.

Клиентская часть веб-приложения обеспечивает взаимодействие пользователя с приложением. Она предоставляет интерфейс для взаимодействия с пользователем и выполнение разнообразных операций, что делает её неотъемлемой частью современных веб-проектов. Особенность клиентской части является её способность к отображению данных, реагированию на действия пользователя, отправке запросов к серверу и получению обновлений. Клиентская часть работает непосредственно на устройстве пользователя. Это обеспечивает быстрый доступ к функциональности приложения без необходимости постоянного взаимодействия с сервером. Таким образом можно сделать вывод, что клиентская часть веб-приложения – это ключевой элемент в создании приятного и продуктивного пользовательского опыта.

2.2 Разработка базы данных

Учитывая необходимость хранения информации о пользователях, их организациях и сотрудниках, а также следует учитывать, что данные имеют четкую структуру, и ожидаемый большой объем информации, было принято решение использовать реляционную базу данных.

Для удобного использования баз данных существуют готовые решения – системы управления базами данных (СУБД). СУБД разделяются на следующие виды:

- реляционные СУБД – системы, представляющие данные из баз данных в виде таблиц;
- документные – системы, хранящие информацию в виде документов;
- графовые – системы, в которых информация представлена в виде графа, где элементы имеют взаимосвязи друг с другом.

Для проекта было решено использовать реляционный вид СУБД, поскольку информацию в такой системе можно без проблем извлекать, а также производить с ними какие-либо манипуляции. Документные и графовые СУБД, в свою очередь, не предоставляют такой возможности.

Наиболее популярными реляционными СУБД являются такие системы, как PostgreSQL, MySQL, а также SQLite [7]. PostgreSQL же, в свою очередь является идеальным выбором для решения представленной задачи, поскольку из ранее упомянутых СУБД, она имеет большую функциональность и производительность. Эта система управления базами данных (СУБД) обладает рядом преимуществ, включая возможность хранения текста размером до 2 ГБ в одной ячейке и бесплатную доступность для использования. Отмечается, что PostgreSQL широко признана в качестве надежной и проверенной базы данных, и её используют такие крупные компании, как Sony, Huawei, Alibaba, Tripadvisor, Hitachi, Яндекс, Skype, Reddit, Amazon Redshift, FlightAware и многие другие [8].

Необходимо разработать структуру базы данных для хранения разнообразных данных, включая информацию о пользователях (логин, пароль), организациях (название) и сотрудниках организации (фамилия, имя, отчество, должность, рабочее место, класс условий труда, объект исследования), а также данные, представленные в выпадающих списках.

Требуется также установить связь между оценкой вероятности происшествия и ущербом, а также между оценкой вероятности и риском.

Кроме того, для хранения информации, необходимой для карточки охраны труда, следует создать отдельную таблицу. Для выявления потенциальных опасностей, с которыми могут столкнуться работники предприятия. Предполагается подготовить документ, в котором будут указаны обнаруженные риски и опасности.

Поскольку нет нормативно утвержденного формата карты оценки профессиональных рисков (ОПР), специалист по охране труда (СОТ) компании ТЕСАРТ разработал свою карту ОПР.

На первой странице карты ОПР указаны данные сотрудника:

- “Наименование объекта исследования” – остается статичным;
- “Обобщенные трудовые функции работника” – пользователь вводит сам;
- “Перечень рабочих мест и иных объектов исследования, которые подвергаются воздействию опасностей” – пользователь вводит сам, никаких выпадающих списков в этом поле не предусмотрено;
- “Класс условий труда” – Пользователь может ввести сам, но есть база с уже заранее готовыми вариантами ответа, нужно добавить выпадающий список;
- “Приказ о формировании комиссии по оценке профессиональных рисков” – пользователь вводит сам.

Первая таблица в карте ОПР представляет собой набор наименований объектов возникновения опасностей. Пользователь при заполнении карточки будет выбирать, на какой территории работает сотрудник, в каких зданиях, с какими машинами и оборудованием она взаимодействует, какие инструменты использует, с каким сырьем и материалами работает и с какими биологическими объектами взаимодействует. Чтобы пользователь мог быстрее заполнить все эти поля, у нас есть уже подготовленные варианты заполнения этих полей. Далее все эти поля по типу будут заполняться и перечисляться в данной таблице.

Поля следующих таблиц имеют следующую структуру:

- “Наименование выявленных опасностей” – пользователь может выбрать из заранее подготовленных вариантов или ввести сам;
- “Наименование идентифицированной опасности (источника риска)” – пользователь может выбрать из заранее подготовленных вариантов или ввести сам;
- “Опасное событие” – пользователь может выбрать из заранее подготовленных вариантов или ввести сам;
- “Оценка вероятности (с учетом длительности)” – пользователь может выбрать из заранее подготовленных вариантов;
- “Ущерб, баллов” – Выставляется автоматически, в зависимости от введенного поля
- “Оценка вероятности (с учетом длительности)”;
- “Риск” – Выставляется автоматически, в зависимости от введенного поля;
- “Оценка вероятности (с учетом длительности)”;
- “Мероприятия” – Пользователь вводит сам.

Последняя, таблица – “Перечень мер контроля рисков (защиты от опасности), созданный на основании идентификации опасностей”. В данной таблице идет объединение всех данных из таблиц 2-6 карты ОПР. Таблицы здесь имеют структуру, приведённую выше, с добавлением нескольких новых полей:

- “Ответственный” – Пользователь выбирает из сотрудников, введенных в его организации;
- “Срок исполнения мероприятия до” – Пользователь вводит дату;

Готовая ER диаграмма на рисунке 2.1.

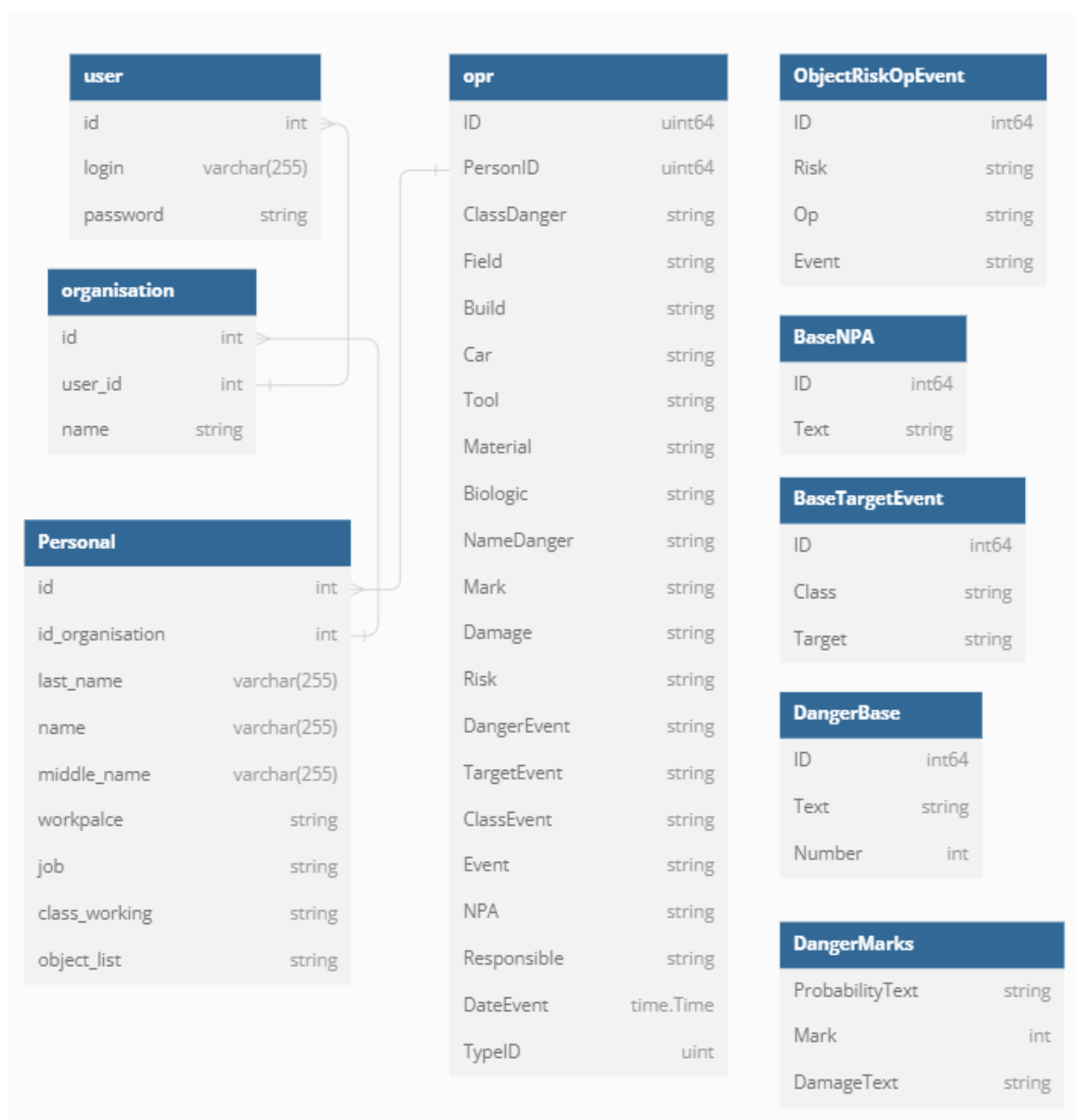


Рисунок 2.1 – Физическая модель базы данных

2.3 Разработка архитектуры серверной части приложения

Для разработки серверной части приложения принято решение использовать высокоуровневый язык программирования Golang [9].

У языка Go много преимуществ. Некоторые из них уникальны для Go, а другие свойственны и иным языкам программирования. Среди наиболее значимых преимуществ и возможностей Go можно отметить следующие:

Код Go является переносимым, особенно между UNIX-машинами;

Go поддерживает сборку мусора, поэтому не придется заниматься выделением и освобождением памяти;

По умолчанию в Go используется статическая компоновка — это значит, что создаваемые двоичные файлы легко переносятся на другие компьютеры с той же ОС.

Как следствие, после успешной компиляции Go-программы и создания исполняемого файла не приходится беспокоиться о библиотеках, зависимостях и разных версиях этих библиотек;

Go поддерживает Unicode, а следовательно, вам не понадобится дополнительная обработка для вывода символов на разных языках;

Идеального языка программирования не существует, и Go не исключение.

Есть языки программирования, которые эффективнее в некоторых других областях программирования. Вот некоторые из недостатков Go:

- В Go нет встроенной поддержки объектно-ориентированного программирования. Это может стать проблемой для тех программистов, которые привыкли писать объектно-ориентированный код;
- С все еще остается более быстрым, чем любой другой язык системного программирования главным образом потому, что UNIX написана на C.

Интерес к этому языку продолжает расти. С 2018 по 2020 год Go был языком номер один, который разработчики хотели изучать. Такие компании, как Uber, Twitch, Dropbox и Google используют Go в своем технологическом стеке [10, 11, 12].

2.4 Передача данных в клиентскую часть

В разрабатываемом приложении наблюдается значительный объем данных, что приводит к частым и сложным запросам к базе данных. Этот факт обуславливает необходимость для разработчиков создавать структуры данных как на серверной, так и на клиентской стороне приложения для каждого запроса. Подобный подход может повлечь за собой ряд потенциальных

ошибок. Для минимизации рисков было принято решение о внедрении GraphQL [13].

GraphQL представляет собой язык запросов для пользовательского API, обеспечивающий возможность клиентам запрашивать только необходимые им данные, независимо от структуры сервера. Этот протокол был разработан в 2012 году и впоследствии стал открытым стандартом в 2015 году.

Преимущества GraphQL включают в себя гибкость запросов, сокращение объема передаваемых данных, способность обеспечивать эволюцию API без нарушения совместимости и предоставление инструментов для автоматической генерации документации и отладки. Тем не менее использование GraphQL может потребовать дополнительных усилий по настройке и освоению соответствующих концепций.

2.5 Разработка архитектуры клиентской части приложения

Современные веб-интерфейсы отличаются от предшествующих поколений своей динамичностью и интерактивностью. Они стали не просто информационными площадками, а мощными инструментами, позволяющими осуществлять различные операции, такие как совершение покупок, общение с другими пользователями, и написание комментариев.

С изменением целей и функционала веб-сайтов изменилась и их структура. Ранее сайты представляли собой набор отдельных HTML-страниц, тогда как современные веб-приложения часто реализуются как единый HTML-документ, изменяемый динамически при помощи JavaScript.

Такой подход получил название SPA (Single-Page Application), или одностраничное приложение. Оно трансформирует веб-интерфейсы в полноценные приложения, которые не требуют перезагрузки страницы при переходах между различными частями приложения. Вместо этого, для перехода между разделами приложения происходит динамическая

перерисовка содержимого страницы с помощью JavaScript и полученных данных [14].

Данный подход обусловлен несколькими ключевыми факторами, важными для современных веб-приложений. В первую очередь, целевые пользователи работают в основном табличных препроцессорах, и добавляют и изменяют данные непосредственно в таблицах, в такой логике работы интерфейса пользователь не должен ждать обновления всей страницы, а страницы должны обновляться быстро и точно для комфортной работы. Кроме того, современные пользователи часто не обращают внимания на то, где произошла генерация интерфейса - на сервере или на клиенте. Они ценят скорость и отзывчивость приложения, а также его функциональность. Именно поэтому одностраничные приложения, стали широко распространены и популярны в веб-разработке.

Исходя из этого разработка приложения на React TypeScript Vite Apollo обосновывается несколькими ключевыми факторами, важными для современных веб-приложений. Прежде всего, выбор React как основной библиотеки обусловлен её популярностью, активным сообществом разработчиков и широким набором инструментов для создания пользовательских интерфейсов [14]. TypeScript, являющийся строго типизированной реализацией JavaScript, что придает проекту дополнительную стабильность и улучшает читаемость кода благодаря статической типизации.

Использование Vite в качестве инструмента сборки обусловлено его реализацией упрощенной сборки и отладки проекта. Vite предлагает инновационный подход к разработке веб-приложений, поддерживая горячую перезагрузку и инкрементальную сборку, что способствует более быстрой разработке и обновлению кода.

Интеграция с Apollo Client обеспечивает возможность эффективной работы с графовыми данными и обеспечивает клиент-серверное взаимодействие посредством GraphQL. Apollo Client предоставляет

инструменты для управления состоянием приложения и запросов к серверу, обеспечивая высокую гибкость и производительность.

Таким образом, выбор стека React TypeScript Vite Apollo для разработки клиентской части приложения обусловлен стремлением к стабильности и эффективности в разработке и поддержке современных веб-приложений.

3 РАЗРАБОТКА СЕРВЕРНОЙ ЧАСТИ ПРИЛОЖЕНИЯ

3.1 Структура проекта

Эффективная организация кода играет важную роль в разработке программного обеспечения. Она обеспечивает читаемость, поддерживаемость и масштабируемость кодовой базы, что является важными факторами успешного развития приложения. Принято решение разделить отдельные части кода, отвечающие за разные модули приложения.

Важно понимать, что не существует одного правильного или даже рекомендуемого способа структурирования веб-приложений в Go. Принято решение о проекте на использовании подхода с официального примера организации кода клиент-серверного приложения. На рисунке 3.1 представлена структура серверной части проекта.

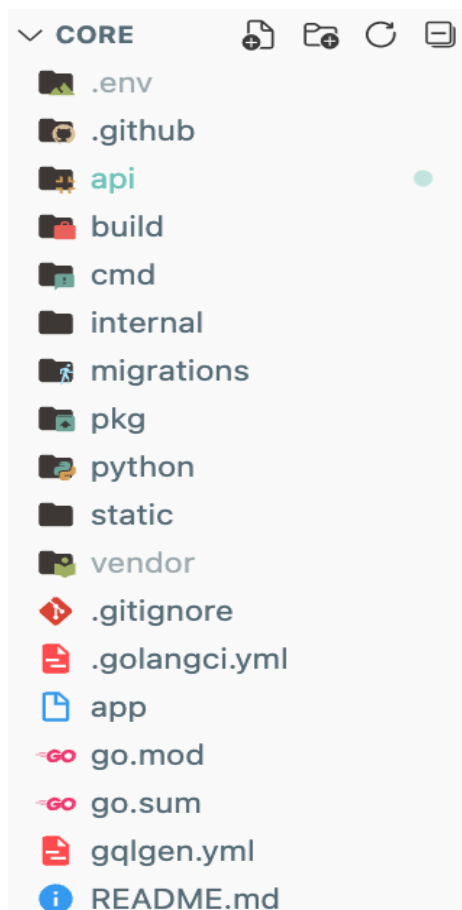


Рисунок 3.1 – Структура проекта

Папка `cmd` содержит папки разных приложений в проекте. На данный момент у нас будет только одно исполняемое приложение и это наше веб-приложение.

Папка `pkg` содержит вспомогательный код, не зависящий от приложения в проекте. Было решено использовать данную папку для хранения вспомогательного кода, который потенциально может быть повторно использован. Это могут быть вспомогательные средства проверки данных и модели базы данных для проекта.

Папка `static` содержит статичные файлы, шаблоны документов, аватарки пользователей.

Папка `build` содержит файлы, для автоматического сбора приложения на сервере.

Папка `api` будет содержать файлы, для реализации `graphql`.

Такая структура отлично масштабируется, если требуется добавить в проект еще одно исполняемое приложение. Например, если требуется добавить CLI (Command Line Interface) для автоматизации некоторых административных задач в будущем. С такой структурой можно создать это CLI приложение в `cmd/cli`, и оно сможет импортировать и повторно использовать весь код, который была написан в папке `pkg`.

3.2 Создание документов по шаблону в web-платформе

Основной функционал приложения заключается в автоматическом создании документов, на основе заранее подготовленных шаблонов.

Для шаблонов документов принято решение использовать формат хранения данных `docx`, так как является одним из самых популярных форматом хранения текстовых документов, широко используется в организациях и легко переводится в другие форматы.

Шаблоны хранятся в формате docx и для работы с ними используется библиотека go-docx, а также библиотека docx на языке программирования python.

В шаблонах имеются специальные метки, которые находит программа и заменяет их в зависимости от требуемых данных.

В шаблоне существуют таблицы. Заранее неизвестно сколько строк будет в документе, поэтому увеличивать строки в таблице приходится уже на стадии создания документа. К сожалению, в библиотеке go-docx на языке программирования golang, которую используем для взаимодействия с шаблоном нет функции создания новых строк для таблиц, поэтому было принято решение написать скрипт на языке программирования Python с помощью библиотеки docx увеличивается количество строк в документе.

После создания документа на основе шаблона, готовый документ отправляется на клиентскую часть приложения и скачивается на устройство пользователя.

На рисунке 3.2 представлен скриншот шаблона документа «ОПР (КИО, КОР, КУР)».

ОЦЕНКА ПРОФЕССИОНАЛЬНЫХ РИСКОВ

КАРТА № {1}

{организация}

(наименование организации)

Наименование объекта исследования: **Рабочее место, с учетом наличия вредных (опасных) производственных факторов**

Обобщенные трудовые функции работника: {работа с ПЭВМ}

Перечень рабочих мест и иных объектов исследования, которые подвергаются воздействию опасностей: {инженер, специалист, программист, техник} → →

Классе условий труда по СОУТ: {2 класс, допустимые условия труда}

Приказ о формировании комиссии по оценке профессиональных рисков: {Приказ № 32 от 13.02.2022г}

(наименование, номер, дата)

I. → Идентификация опасностей и оценка рисков

Наименование объектов возникновения опасностей:

Территория:	{Территория цеха/подразделения}
Здания и сооружения:	{Производственные, административные}
Машины и оборудование:	{Автомобильный транспорт, оборудование}
Инструменты и приспособления:	{Строительные инструменты, слесарные инструменты, электроинструменты}
Сырье и материалы:	{Заготовки деталей}
Биологические объекты:	{Посторонние лица, животные}

Рисунок 3.2 – Скриншот шаблона документа «ОПР (КИО, КОР, КУР)»

На рисунке 3.3 представлен пример кода части функции, которая создает готовый документ на основе шаблона.

```

func CreateCardKioKorKur(orgID int64, internalID int64) (string, string, error)
// Открытие docx файла
doc, err := docx.Open("../static/documents/test.docx")
if err != nil {
    log.Printf("Ошибка открытия документа: %s", err)
}

// Ключ-значение для шаблона
replaceMap := docx.PlaceholderMap{
    "fio":          `${person.LastName} ${person.FirstName}
${person.MiddleName}`,
    "job":          person.Position,
    "unit":         person.Unit,
    "personel_number": person.PersonnelNumber,
    "organisation": organisation.Name,
    "class_working": person.ClassWorking,
    "workplace":    person.Workplace,
    "number":        person.InternalID,
    "object_study": person.ObjectStudy,
}

////////////////////////////////////
//Замена параметров основных 5 таблиц
num := 1.1
increment := 0.1
precision := 1
numInCard := 1
lastTable := 1
for tableNum := 1; tableNum <= 5; tableNum++ {

```

Рисунок 3.3 – Пример функции создания документа по шаблону

3.3 Получение данных из базы данных

При запуске серверной части приложения происходит инициализация подключения к базе данных PostgreSQL. При подключении из переменных окружения берутся адрес, порт, логин и пароль юзера для базы данных. После успешного подключения программа выводит в логи сообщение об успешном подключении.

Структура таблиц из базы данных хранится в файле generated.go, который находится в api/graph. Таблицы из базы данных хранятся в виде структур, которые повторяют названия таблицы. В структурах хранятся поля, идентичные полям в таблице в БД и типы данных для этих полей.

Запрос на получение данных из БД происходят при помощи библиотеки gorm, и его методов, таких как Select(), Find(), Where(). Такие методы значительно упрощают написание запросов в БД. Вместо длинных,

классических запросов в БД, при помощи нескольких методов можно получить нужные данные.

Пример инициализации подключения к БД приведен в приложении Б.

3.4 Логгирование

Логгирование – это процесс записи сообщений и событий, происходящих в программном обеспечении, в некоторый файл или другое хранилище данных, называемое журналом (лог-файлом). Целью логгирования является регистрация различных событий, таких как ошибки, предупреждения, информационные сообщения, а также информация о ходе выполнения программы.

Для логгирования в приложениях обычно используются специальные библиотеки или инструменты, которые позволяют программистам записывать сообщения в журнал с различными уровнями подробности (например, отладочные, информационные, предупреждения и ошибки).

Пример инициализации логгера и его функции приведены в приложении В.

4 КОММУНИКАЦИОННЫЕ ПРОЦЕССЫ В WEB-ПЛАТФОРМЕ

4.1 GraphQL

GraphQL – это язык запросов для API и среда выполнения запросов с открытым исходным кодом для выполнения запросов с использованием типизированных данных. Он помогает упростить передачу данных между серверной частью и клиентской [15].

Запросы GraphQL позволяют клиентам запрашивать только те данные, которые им нужны, и получать эти данные в одном запросе. GraphQL использует систему типов для определения данных, которые можно запросить и получить от сервера. Каждый запрос имеет строго определенную структуру, что обеспечивает ясность и однозначность в том, какие данные будут получены.

В серверной части приложения определяются GraphQL схемы и типы данных с помощью спецификации GraphQL SDL (Schema Definition Language). В этих схемах описываются доступные запросы (Query), мутации (Mutation) и подписки (Subscription), а также соответствующие типы данных, которые можно запросить или изменить через GraphQL API. Для каждого типа данных в GraphQL-схеме обычно создаются соответствующие классы или структуры в коде серверной части приложения. Эти классы обычно содержат логику для получения данных из источников (например, базы данных или внешних API) и преобразования этих данных в формат, понятный для клиента.

В клиентской части приложения используется GraphQL-клиент для отправки запросов к серверу GraphQL и получения данных в ответ. Обычно в клиентской части приложения определяются GraphQL-запросы, которые нужно выполнить для получения необходимых данных. Эти запросы могут быть написаны с использованием GraphQL-запросов внутри кода клиентской части или с использованием инструментов, таких как Apollo Client или Relay,

которые позволяют определять запросы в более декларативном стиле. Полученные данные обычно используются для обновления пользовательского интерфейса приложения, например, для отображения списка элементов или обновления отдельных компонентов.

Серверная часть приложения определяет структуры данных и доступные операции с ними с помощью GraphQL-схем. Клиентская часть, в свою очередь, приложения отправляет запросы к серверу GraphQL и обрабатывает полученные данные для обновления пользовательского интерфейса.

4.2 Отправка данных с сервера клиенту

Для отправки данных с сервера на клиент с помощью GraphQL, клиент формирует запрос на языке GraphQL. Запрос содержит тип операции, например `query`, а также набор полей и аргументов, которые клиент хочет получить. Запрос на получение данных с сервера представлен на рисунке 4.1.

```
const { loading, error, data } = useQuery(gql`
  query {
    organization(orgID: ${organisationID}) {
      name
      persons {
        id
        firstName
        lastName
        middleName
        department
        position
        workstation
      }
    }
  }
`);
```

Рисунок 4.1 – Запрос GraphQL

Запросы, сформированные на клиенте, отправляются на сервер через HTTP POST или GET запросы. На сервере, в свою очередь, этот запрос принимается и обрабатывается. Он проверяется на соответствие схеме GraphQL, чтобы убедиться, что все запрашиваемые поля и аргументы корректны. Далее сервер выполняет запрос, вызывая соответствующие резолверы для каждого поля. Резолверы могут обращаться к базе данных или другим источникам данных для получения нужной информации. После выполнения запроса сервер формирует ответ в формате JSON и отправляет его обратно клиенту.

Чтобы между сервером и клиентом передавались данные с нужными ключами в формате JSON, необходимо в серверной части создать структуру данных для каждой таблицы из базы данных. Структура данных организации представлена на рисунке 4.2.

```
Organization struct {  
    City          func(childComplexity int) int  
    DirectorFullName func(childComplexity int) int  
    Email         func(childComplexity int) int  
    ID            func(childComplexity int) int  
    Inn           func(childComplexity int) int  
    Kpp           func(childComplexity int) int  
    LegalAddress  func(childComplexity int) int  
    Name          func(childComplexity int) int  
    Persons       func(childComplexity int) int  
    PhoneNumber   func(childComplexity int) int  
    PostalAddress  func(childComplexity int) int  
    PostalCode     func(childComplexity int) int  
}
```

Рисунок 4.2 – Структура данных организации

На рисунке 4.3 представлен пример использования резолверов в GraphQL.

```
43 // Person is the resolver for the person field.
44 Codeium: Refactor | Explain | X
45 func (r *queryResolver) Person(ctx context.Context, orgID *string, id string)
46 (*model.Person, error) {
47     panic(fmt.Errorf("not implemented: Person – person"))
48 }
49
50 // Mutation returns MutationResolver implementation.
51 Codeium: Refactor | Explain | X
52 func (r *Resolver) Mutation() MutationResolver { return &mutationResolver{r} }
53
54 // Query returns QueryResolver implementation.
55 Codeium: Refactor | Explain | X
56 func (r *Resolver) Query() QueryResolver { return &queryResolver{r} }
57
58 type mutationResolver struct{ *Resolver }
59 type queryResolver struct{ *Resolver }
```

Рисунок 4.3 – Пример использования резолверов

5 РАЗРАБОТКА КЛИЕНТСКОЙ ЧАСТИ ПРИЛОЖЕНИЯ

5.1 Фреймворк React

Подход в использовании веб-технологий в разработке платформы по автоматизации процессов охраны труда и промышленной безопасности обусловлен её доступностью. Она обеспечивается кроссплатформенностью. Это позволяет приложениям работать на любом устройстве с доступом к интернету, будь то стационарный компьютер или телефон. Это может расширить аудиторию приложения, а также снизить необходимость разработки и поддержки отдельных версий для разных платформ. Быстрое обновление и развертывание веб-приложений позволяет разработчикам в любой момент предоставлять пользователям новые функции и исправления без необходимости загрузки и установки обновлений. Кроссплатформенность – полезное свойство веб-приложения, особенно после выхода нового постановления правительства РФ “О порядке проведения требований организации охраны труда”, где упоминается о том, что пользователи должны постоянно использовать последнюю версию того или иного приложения.

В качестве технологии для разработки клиентского приложения принято решение использовать React. Это технология, разработанная компанией Meta Platforms, Inc. Она стала одним из самых популярных инструментов для создания пользовательских интерфейсов благодаря своей компонентной архитектуре. Она позволяет создавать многоразовые и изолированные компоненты, что упрощает разработку, тестирование, а также поддержку кода. Компоненты могут быть легко использованы в разных частях приложения. Это снижает дублирование кода и ускоряет процесс разработки [16].

Одним из ключевых преимуществ React является использование виртуального DOM, который значительно улучшает производительность приложения за счет минимизации манипуляций с реальным DOM. Это особенно важно для сложных интерфейсов, где требуется частое обновление данных и взаимодействие с пользователем [17]. Виртуальный DOM в React

позволяет эффективно обновлять только те части интерфейса, которые изменились. При этом не затрагивая те части интерфейса, которые остались без изменений. Данный подход существенно ускоряет работу приложения [18, 19].

Стоит отметить, что React обладает обширной поддержкой сообщества, и крупных компаний, что обеспечивает надежность и постоянное развитие технологии. Многие компании используют React для разработки своих приложений, что свидетельствует о его высокой надежности и эффективности. Например, в настоящее время наиболее предпочтительным инструментом для разработки торговых приложений, где важна высокая производительность и быстрая реакция на изменения данных. Большое количество библиотек и инструментов, таких как `redux` для управления состоянием и `React router` для маршрутизации, делает разработку на React более удобной и эффективной, что позволяет разработчикам использовать готовые решения для типичных задач, что ускоряет процесс разработки и улучшает качество конечного произведенного продукта [20].

Обмен данными между клиентом и сервером осуществляется посредством языка запросов GraphQL. Этот язык позволяет запрашивать и получать только те данные, которые действительно необходимы, что значительно снижает объем передаваемой информации, экономит пользовательский трафик и ускоряет работу клиента при слабом интернет-соединении. Дополнительно, это снижает нагрузку на сервер благодаря оптимизации запросов для получения только требуемых данных.

5.2 Хранение данных у клиента

В процессе разработки веб-платформы для автоматизации процессов охраны труда и промышленной безопасности возникла необходимость временного хранения файлов документов, сгенерированных на стороне сервера. Для доступа к этим файлам генерируется специальная ссылка, представляющая собой случайную строку. Эта строка также сохраняется в

Redis в качестве ключа, причем время жизни записи ограничивает доступ к файлу определенным периодом. В значении записи хранится ссылка на файл в файловой системе и идентификатор пользователя, который может скачать этот файл. Через каждые несколько часов специальный процесс, запускаемый по расписанию с помощью cron, проверяет наличие всех ссылок на файлы в Redis, и при их отсутствии удаляет соответствующие файлы из файловой системы.

Для хранения ключей на клиенте используются cookie с опцией HTTPOnly, что предотвращает кражу ключей посредством XSS-атак. Также cookie имеют атрибут Domain, который позволяет автоматически отправлять ключ в HTTP-заголовках только к запросам на API-домен. Подобно Redis, cookie имеют время жизни, после истечения которого ключ автоматически удаляется [21].

Для верификации запросов от пользователя используются ключи, передаваемые в специально введенных HTTP-заголовках. Хранение этих ключей на сервере осуществляется с помощью резидентной СУБД Redis, работающей с структурами данных типа "ключ-значение". Redis является наиболее подходящей для данной задачи, поскольку обеспечивает сравнительно быстрый поиск по ключу по сравнению с реляционными базами данных. При каждом запросе осуществляется обращение к базе данных для подтверждения наличия ключа и сопоставления его с идентификатором пользователя. Дополнительно, в Redis реализована функция TTL (Time-To-Live), которая позволяет установить время жизни записи, тем самым ограничивая период, в течение которого ключ будет действителен.

5.3 СУБД Redis

Веб-приложения часто требуют механизмов для управления сессиями пользователей. Два основных подхода включают хранение сессий на сервере, например, с использованием Redis, и шифрование информации о сессии в строке с ее последующим хранением на клиенте. Рассмотрим преимущества

использования Redis для хранения сессий по сравнению с хранением зашифрованной информации о сессии на клиенте [22].

Хранение сессий на сервере в Redis обеспечивает высокий уровень безопасности, так как конфиденциальная информация не передается клиенту и не хранится на его стороне. Это уменьшает риски атак, связанных с подменой сессионных данных или их кражей при XSS-атаках. Как пример можно привести такую ситуацию, когда при хранении сессии на клиенте злоумышленник может попытаться расшифровать данные сессии и получить доступ к учетной записи пользователя.

Использование Redis для хранения сессий позволяет централизованно управлять всеми активными сессиями. Это упрощает администрирование, мониторинг и контроль за сессиями пользователей, что особенно важно для крупных приложений с большим числом пользователей. Администраторы могут легко завершать сессии, управлять временем их жизни и отслеживать активность пользователей в реальном времени [23].

Redis, являясь высокопроизводительной резидентной СУБД, обеспечивает быструю обработку запросов и низкие задержки при доступе к данным. Это критически важно для масштабируемых веб-приложений с высоким уровнем нагрузки. Redis поддерживает горизонтальное масштабирование, что позволяет обрабатывать увеличивающийся объем трафика без потери производительности.

Redis предоставляет встроенные механизмы для управления временем жизни сессионных данных с помощью функции TTL (Time-To-Live). Это позволяет автоматически удалять устаревшие сессии, что освобождает ресурсы и повышает безопасность, предотвращая длительное хранение неактуальной информации [24].

Redis поддерживает различные механизмы резервного копирования и репликации, что обеспечивает высокую надежность и отказоустойчивость системы. В случае сбоя основной базы данных сессий, резервные копии могут быть быстро восстановлены, минимизируя простой системы и потерю данных.

Хранение зашифрованной информации о сессии на клиенте делает ее более уязвимой к различным атакам, включая XSS и CSRF. Даже при использовании сильных методов шифрования существует риск утечки ключей шифрования или компрометации самого алгоритма шифрования. Кроме того, хранение информации о сессии на клиенте ограничивается максимальным размером cookie или токена, что может быть недостаточно для хранения всех необходимых данных. Это накладывает ограничения на объем информации, которая может быть сохранена в сессии.

При хранении сессий на клиенте сложно централизованно управлять активными сессиями, что затрудняет администрирование и контроль. Это также усложняет задачи, связанные с завершением сессий и управлением временем их жизни.

Использование Redis для хранения сессий обеспечивает более высокий уровень безопасности, централизованное управление, высокую производительность и масштабируемость, а также надежность и отказоустойчивость системы. Эти преимущества делают Redis предпочтительным выбором для управления сессиями в современных веб-приложениях по сравнению с хранением зашифрованной информации о сессии на клиенте.

5.4 Интерфейс клиентской части приложения

При первом посещении сайта, программа отправляет пользователя на главную страницу сайта. На рисунке 5.1 представлена главная страница сайта. В верхней части страницы находятся кнопки входа и регистрации в приложении. В центральной части страницы находится ролик, рассказывающий про основные особенности web-платформы.

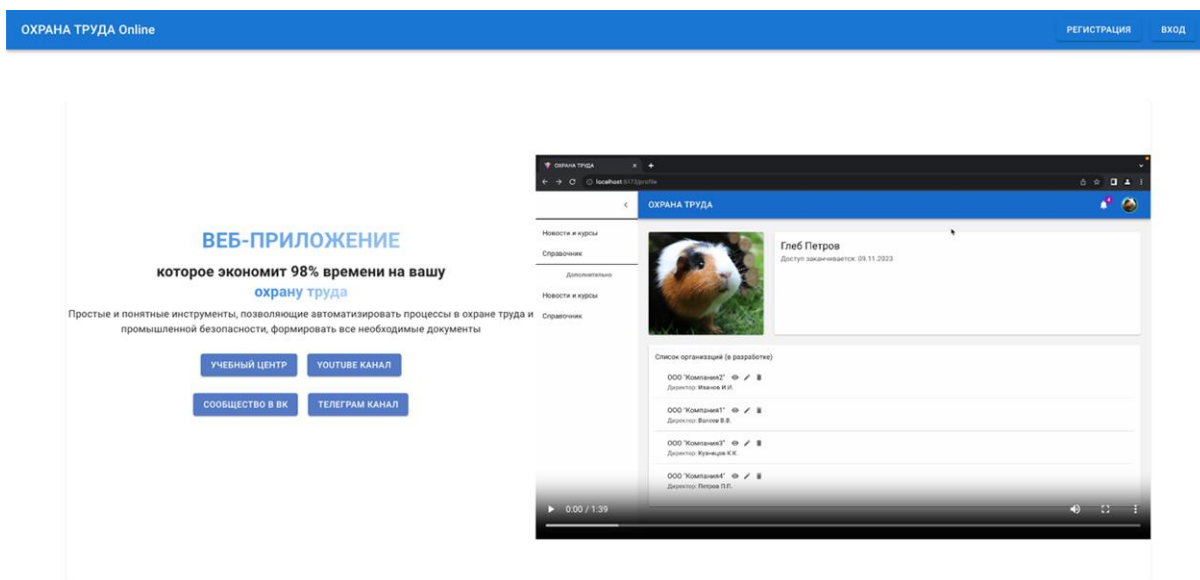


Рисунок 5.1 – Пример главного окна

На рисунке 5.2 представлен пример модального окна «Добавление сотрудника». В поле ввода «Должность» добавлен выпадающий список должностей, который создан на основе приказа Минтруда России №804 от 21 декабря 2022 г. В данном приказе есть перечень должностей, которые могут быть на производстве. Пользователь может написать свою должность, либо выбрать из выпадающего списка. При вводе символов, выпадающий список сортируется по совпадениям с введенными символами. Таким образом пользователь может быстро найти нужную должность.

На рисунке 5.3 приведен пример модального окна при заполнении карточки КИО – карта идентификации опасностей. В данной карточке пользователь должен определить классификации опасностей работника, указать на какой территории работает сотрудник и с какими опасностями контактирует. В каждое поле СУОТ может ввести свои параметры или выбрать из выпадающего списка, таким образом приложение ускоряет работу с данными.

ОХРАНА ТРУДА Online

000 TEST

+

+

Профиль

Справочник

Документы

Имя

Отчество

Подразделение

Должность

Рабочее место

Карточки ОПР

Фамилия

Имя

Отчество

Подразделение

Должность

Рабочее место

Класс условий труда

Объект исследования

Табельный номер

Дата приема на работу

Дата перевода на другую должность

Добавить нового сотрудника

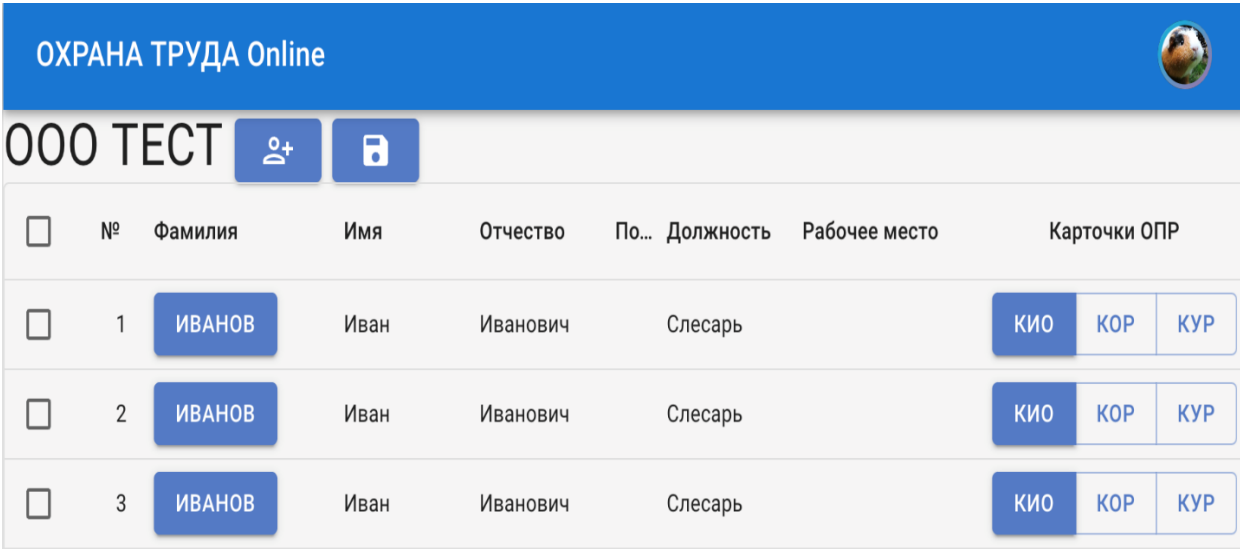
Рисунок 5.2 – Пример модального окна «Добавление сотрудника»

5.5 Создание документов в клиентской части

На рисунке 5.4 представлен интерфейс главной страницы организации, где отображены все добавленные сотрудники организации, а также их данные. Над самими данными представлено название той или иной организации, в которой ведётся работа над представленными данными. Редактирование данных сотрудников осуществляется с помощью отдельного меню, которое открывается нажатием на активную область, расположенной на ФИО сотрудника. Кнопки справа от сотрудника редактируют данные для создания документов по оценке профессиональных рисков сотрудника.

Поскольку большинство сотрудников, связанных с охраной труда, работают в табличном редакторе Microsoft Excel, решено использовать табличное представление данных в интерфейсе меню главной страницы организации. Таким образом интерфейс позволяет добавлять или удалять столбцы и строки.

На рисунке 5.5 представлен интерфейс настройки документов, которые создаются по шаблону и скачиваются на персональное устройство пользователя.



<input type="checkbox"/>	№	Фамилия	Имя	Отчество	По...	Должность	Рабочее место	Карточки ОПР
<input type="checkbox"/>	1	ИВАНОВ	Иван	Иванович		Слесарь		КИО КОР КУР
<input type="checkbox"/>	2	ИВАНОВ	Иван	Иванович		Слесарь		КИО КОР КУР
<input type="checkbox"/>	3	ИВАНОВ	Иван	Иванович		Слесарь		КИО КОР КУР

Рисунок 5.4 – Таблица сотрудников организации

000 ТЕСТ

ОПР (КИО, КУР, КОР).docx

Перечень мер контроля управление рисками.docx

Реестр профессиональных рисков.docx

Председатель

ФИО
Иванов Иван Иванович

Приказ

Приказ

Дата приказа Дата

Дата Приказа 01.02.2022

01.02.2022

Комиссия

ФИО
Иванов Иван Иванович

Добавить члена комиссии

Сохранить

Рисунок 5.5 – Интерфейс настройки документов

На рисунке 5.6 представлен пример созданного и скачанного пользователем документа, созданного на основе шаблона с данными, которые ввёл пользователь.

Перечень мер контроля рисков (защиты от опасности), созданный на основании идентификации опасностей

№ п/п	Наименование выявленных опасностей	Опасное событие	Риск	Мероприятия	Ответственный	Срок исполнения мероприятий, до
1	2	3	4	5	6	7
1.1	Поверхности, имеющие высокую температуру (воздействие конвективной теплоты)	Тепловой удар от воздействия окружающих поверхностей оборудования, имеющих высокую температуру, ожог кожных покровов работника	Средний (существенный) риск. Требуются меры по снижению риска в установленные сроки	Охлаждение нагретых материалов, изделий и передвижного оборудования непосредственно в рабочих помещениях на специальноном участке, оборудованном устройством для местного удаления выделяемого тепла и защиты работающих от теплового облучения	Петров Глеб	03.03.2024
1.2	Дикие или домашние животные	Тепловой удар от воздействия окружающих поверхностей оборудования, имеющих высокую температуру, ожог кожных покровов работника	Средний (существенный) риск. Требуются меры по снижению риска в установленные сроки	Охлаждение нагретых материалов, изделий и передвижного оборудования непосредственно в рабочих помещениях на специальноном участке, оборудованном устройством для местного удаления выделяемого тепла и защиты работающих от теплового облучения	Петров Глеб3	07.04.2024
1.3	Груз, инструмент или предмет, перемещаемый или поднимаемый, в том числе на высоту	Укус животного, травма, нанесенная зубами и когтями животного, нападение животного, отравление ядами животного	Малый (умеренный) риск. Меры не требуются	Кормление животных и уборка их помещений с помощью специального инвентаря, который поддерживается исправным, является легким, удобным и достаточно длинным,	Петров Глеб	24.03.2024

Рисунок 5.6 – Документ, созданный по шаблону

6 БИЗНЕС-МОДЕЛЬ

6.1 Концепция стартап-проекта

Что такое стартап

6.1.1 Описание продукта

Объектом разработки является web-платформа автоматизации процессов охраны труда и промышленной безопасности. Задача состоит в разработке клиент-серверного приложения, которое бы упрощало бы создание документов в сфере охраны труда.

Платформа будет состоять из 5 модулей. Каждый модуль будет содержать уникальные шаблоны, создавать уникальные документы.:

- Оценка профессиональных рисков;
- Средства индивидуальной защиты;
- Пожарная безопасность;
- Электробезопасность;
- Инструктажи.

Область применения системы – малые и средние компании, которым невыгодно держать сотрудника по охране труда. Благодаря web-платформе малые и средние компании не будут тратить много денег на охрану труда.

Основной особенностью платформы, на фоне других приложений по охране труда, является низкая стоимость, система подписки, и удобство использования на мобильных приложениях.

6.1.2 Рынок охраны труда

Малые и средние предприятия играют важную роль в экономике России. МСП являются основным источником занятости для значительной части населения [25].

Одно из преимуществ МСП является их способность быстро адаптироваться к изменениям на рынке и внедрять новшества. В отличие от крупных компаний, МСП могут оперативно реагировать на потребности потребителей и изменять свою стратегию в соответствии с текущими условиями. Это делает их более устойчивыми в условиях экономической нестабильности и позволяет находить новые ниши на рынке.

К субъектам малого и среднего предпринимательства относятся внесенные в единый государственный реестр юридических лиц потребительские кооперативы и коммерческие организации (за исключением государственных и муниципальных унитарных предприятий), а также физические лица, внесенные в единый государственный реестр индивидуальных предпринимателей и осуществляющие предпринимательскую деятельность без образования юридического лица (далее — индивидуальные предприниматели), крестьянские (фермерские) хозяйства, соответствующие перечисленным ниже условиям.

Для малого и среднего бизнеса существуют ограничения по численности работника. В зависимости от средней численности работников за календарный год предприятия подразделяются на:

- микропредприятия - до 15 работников
- малые предприятия - до 100 работников;
- средние предприятия - от 101 до 250 работников [26].

Таким предприятиям экономически не выгодно нанимать еще одного сотрудника, который работал с документами по охране труда. Экономически выгодно дать это задание на аутсорсинг или воспользоваться web-платформой. Таким образом МСП могут сократить расходы и повысить эффективность управления. Сотрудничество с внешними специализированными организациями обеспечивает доступ к высококвалифицированным специалистам, которые обладают необходимыми знаниями и опытом для выполнения задач по охране труда.

Большой документооборот в области охраны труда имеют предприятия, у которых существует большое количество рисков для работников. Например, в клининговых компаниях работники часто взаимодействуют с химическими веществами, что повышает риск получить травму во время работы. Также в строительных компаниях рабочие постоянно подвержены риску, поэтому требуется провести большое количество работ, связанных с выявлением опасностей и найти способ избежать такие ситуации.

На 2021 год по данным Росстата насчитывается 6 миллионов МСП. В них работают 10 миллионов человек, а суммарный оборот предприятий составил 57 197 200 000 руб. Их них 5 тысяч клининговых компаний. 5,9% от всех МСП являются строительными компаниями.

Приложения, предназначенные для обеспечения безопасности и охраны труда, имеются на рынке, однако некоторые из них не охватывают все необходимые аспекты данной области. При создании нового приложения, необходимо учитывать все основные аспекты охраны труда, чтобы обеспечить его полноценную функциональность.

6.2 Анализ рынка конкурентов

Одним из ключевых игроков на рынке автоматизации охраны труда выступает приложение 1С:Производственная безопасность. Охрана труда. Данное приложение разработано на базе “1С:Предприятие” для автоматизации процессов и задач охраны труда на разных предприятиях. Функционал имеет: учет, планирование, контроль и отчетность по охране труда в рамках законодательства России.

Программа 1С имеет не дружелюбный интерфейс, по сравнению с современными тенденциями дизайна приложений. На основе опроса специалистов охраны труда оказалось, что пользователям сложно привыкнуть к приложению на основе 1С и требует много времени, чтобы разобраться с интерфейсом.

Большинство работников сферы охраны труда в данный момент используют Excel для хранения данных, изменения данных и создания отчетов. Ведение документации по охране труда в Excel может быть неудобным по нескольким причинам:

1. Ограниченность функционала: Excel представляет собой таблицы и калькуляторы, которые могут ограничивать возможности организации информации и взаимодействия с ней. Например, сложно автоматизировать определенные процессы или предоставить удобный доступ к данным для различных пользователей.

2. Сложность масштабирования: при увеличении объема информации в Excel-файле может стать сложно поддерживать его структуру и обеспечивать эффективное взаимодействие между несколькими пользователями.

3. Ограниченные возможности аналитики и отчетности: В Excel сложно создавать сложные отчеты и анализировать данные, особенно если требуется обработка больших объемов информации или использование специализированных инструментов аналитики.

В таблице 6.1 представлены аналоги разрабатываемого приложения и выделены плюсы и минусы, в сравнении с разрабатываемым приложением.

Таблица 6.1 – Сравнительный анализ систем.

Характеристика	1С:Охрана труда	Актион	Plan Radar
Создание документов по шаблонам в сервисе	+	–	+
Обзор изменений в нормах	–	+	–
Заказ разработки шаблона документа	–	+	+
Заказ разработки шаблона документа	–	+	+
Есть приложение на телефон или планшет	–	–	+

Окончание таблицы 6.1

Стоимость системы	39 900	Полугодие 106 931	126 евро (12 355) в месяц
-------------------	--------	----------------------	---------------------------------

Из данных в таблице можно заключить что на сегодняшний день на рынке представлены небольшое количество приложений, которые упрощают работу по охране труда. Недостатками данных приложения являются отсутствие использование с мобильного устройства, отсутствие русского языка, автоматическое заполнение данных в шаблон и цена. Наше приложение будет выгодно для малого и среднего бизнеса, а также удобно в использовании по сравнению с конкурентами.

6.3 Целевые сегменты и ёмкость рынка

Так как разрабатываемое приложение будет иметь систему подписки, то расчёты прибыли будем вести по месяцам.

Целевым сегментом рынка является B2B, а именно малые и средние предприятия, в основном клининговые компании, строительные компании, аутсорсинговые компании в сфере охраны труда

Web-платформа имеет клиент-серверную архитектуру, что означает, что его можно продавать по всей территории Российской Федерации. В России на 2023 год, по данным Росстата насчитывается более 6 300 000 малых и средних бизнесов. Потенциальная емкость рынка составила 6 300 тыс. подписок.

Разрабатываемое web-приложение ранее было прорекламировано среди экспертов в области IT. Из отобранной 1000 потенциальных пользователей приложение приобрели 10 человек. Исходя из этого можно сделать предположение, что в покупке этого web-приложения заинтересован как минимум 1% от основного числа МСП по России. Таким образом доступная емкость рынка составила 63 000 подписок В первый месяц продаж планируется привлечь сотню пользователей по всей России, а во второй и

третий месяцы 200 и 300 пользователей соответственно. Стоимость товара будет равна 9 000 руб. в месяц.

6.4 Процесс производства. Планирование объёмов производства.

Разрабатываемая web-платформа предусматривает несколько модулей, каждый модуль будет выполнять отдельные виды работ по охране труда. При разработке платформы можно постепенно выпускать модуль и обновлять платформу, так модули будут появляться постепенно в приложении. Для более быстрой разработки платформы потребуются 2 frontend разработчика, 2 backend разработчика, 1 DevOps-программист, 1 аналитик, 1 product-менеджер.

Для выпуска полного приложения требуется потратить время на разработку самой web-платформы. В первый месяц будет производиться планирование и анализ требований, этим будут заниматься аналитик и product-менеджер. Во второй месяц Product-менеджер займется проектированием архитектуры приложения, UI/UX дизайнер разработкой дизайна web-платформы. 3-6 месяц frontend и backend разработчики будут заниматься разработкой web-платформы. В седьмой месяц разработки пройдет тестирование приложения и устранение неполадок. В 8 месяц DevOps инженер займется развертыванием и запуском приложения на удаленном сервере.

Таким образом работники не работают все 8 месяцев разработки, а начинают работать только в свой этап создания приложения и после окончания своего этапа уходят с должности. Такой план поэтапной разработки WEB-платформы позволяет эффективно распределить задачи среди сотрудников, и существенно снизить финансовые затраты. Таким образом грамотно спланированные этапы разработки и последовательное вовлечение специалистов позволят максимально использовать их время, снижая затраты и повышая производительность.

Связи между сотрудниками представлены на рисунке 6.1.

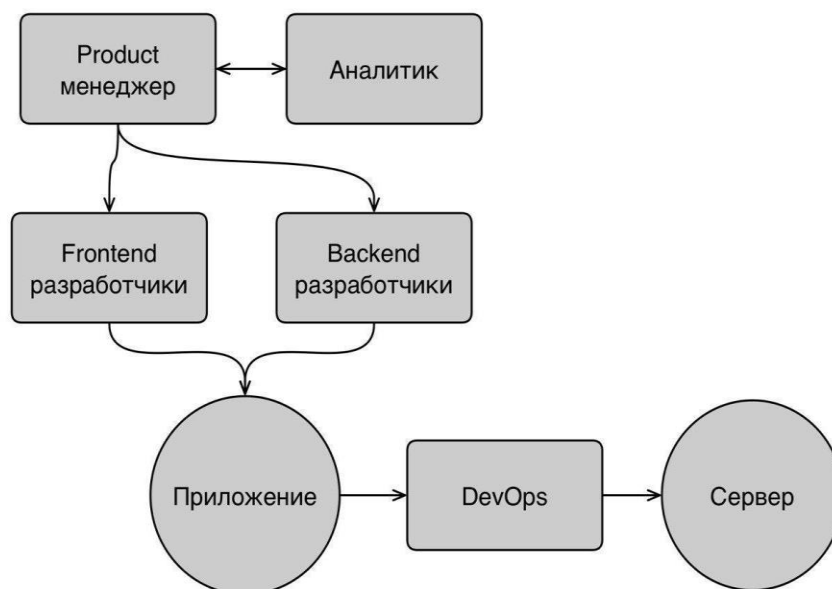


Рисунок 6.1 – Связи между сотрудниками

6.5 Планирование инвестиционных затрат

Инвестиционные затраты состоят из затрат на оборудование, с помощью которого будет вестись разработка web-платформы, затрат на оплату труда разработчиков и аренды сервера для тестирования приложения. В таблице 6.3 представлено оборудование, с помощью которого будет производится разработка web-платформы.

Таблица 6.3 – Оборудование для создания платформы

Наименование оборудования	Кол-во, шт.	Стоимость за шт. (без НДС), руб.	Итого стоимость (без НДС), руб.	НДС (20%), руб.	Итого стоимость (с НДС 20%), руб.	Срок полезного использования, лет
MacBook Air M1 2020	5	71 599,2	357 996	89 499	447 495	5
27'' Монитор АОС Q27G2G/ВК черный	5	17 599,2	87 996	21 999	109 995	5
Итого	-	89 198,4	445 992	111 498	557 490	-

Общие затраты на необходимое оборудование составило 224 984 руб. (без НДС).

Для разработки приложения необходимо будет привлекать следующих специалистов:

- Product-менеджер со сроком работы в 7 месяцев;
- Аналитик со сроком работы в 3 месяца;
- UI/UX дизайнер со сроком работы в 1 месяц;
- Frontend разработчики со сроком работы в 4 месяца;
- Backend разработчики со сроком работы в 4 месяца;
- Тестировщик со сроком работы в 1 месяц;
- DevOps инженер со сроком работы в 1 месяц;
- Консультант в сфере охраны труда со сроком работы в 8 месяцев.

Затраты на оплату труда разработчиков составляют 3,227 млн. руб. Более подробный план работы специалистов представлен в приложении Г.

В таблице 6.4 представлены инвестиционные затраты, в которые добавятся также материальные затраты на создание web-платформы, затраты на патентование и веб-сайт.

Таблица 6.4 – Инвестиционные затраты

Наименование показателя	Сумма с НДС, руб.
Оборудование	557 490
Аренда сервера и домен	42 000
Зарплата разработчиков	3 227 000
Прочие расходы	3 751 000
ИТОГО	7 577 490

Таким образом, просуммировав показатели оборудования, аренды сервера и домена, зарплаты разработчикам и прочие расходы общие инвестиционные затраты на 8 месяцев разработки составляют 7 577 490 руб. с учетом НДС.

6.6 Расчёт операционной прибыли

Планируемая цена подписки на 1 месяц составляет 9 000 руб. Исходя из выявленной емкости рынка была рассчитана выручка от коммерциализации приложения, которая показана в таблице 6.5.

Таблица 6.5 – Планирование прибыли

Показатель	1 месяц	2 месяц	3 месяц	4 -12 месяц
Планируемое количество подписок	100	200	500	700
Выручка, руб	900 000	1 800 000	4500000	6300000

Для поддержки приложения необходимы следующие специалисты: Frontend-программисты; Backend-программисты; DevOps-программист; аналитик; product-Менеджер; тестировщик; UI/UX дизайнер; консультант в сфере охраны труда. Производственный персонал представлен в таблице 6.6 с учетом заработной платы.

К административно-управленческому персоналу относятся следующие специалисты: директор; главный бухгалтер; менеджер по продажам; уборщица. Непроизводственный персонал представлен в таблице 6.7 с учетом заработной платы.

Таблица 6.6 – Трудовые затраты на производственный персонал

Персонал	Кол-во, чел.	Оклад, руб.	Сумма затрат на з/п, руб.	Отчисления в соц. фонды 30%, руб.
Frontend-программист	2	80 000	160 000	48 000
Backend-программист	2	80 000	160 000	48 000
DevOps	1	80 000	80 000	28 000
Аналитик	1	60 000	60 000	18 000
Product-Менеджер	1	60 000	60 000	18 000
Тестировщик	1	60 000	60 000	18 000
UI/UX дизайнер	1	70 000	70 000	21 000
Консультант в сфере охраны труда	1	50 000	50 000	15 000

Таблица 6.7 – Трудовые затраты на непроизводственный персонал

Персонал	Кол-во, чел.	Оклад, руб.	Сумма затрат на з/п, руб.	Отчисления в соц. Фонды 30%, руб.
Директор	1	100 000	100 000	30 000
Главный бухгалтер	1	60 000	60 000	18 000
Менеджер по продажам	1	60 000	60 000	18 000
Уборщица	1	25 000	25 000	7 500

Трудовые затраты составили 1 228 500 руб., из которых 945 000 руб. составляет заработная плата, а 283 500 руб. – отчисление в социальный фонд.

Накладные затраты содержат затраты на: аренду помещения, охранную организацию, командировочные (генерального директора), аренда сервера и домена, прочие затраты (коммунальные услуги). В таблице 6.8 представлена общая информация о накладных затратах, с указанием ежемесячной стоимости.

Таблица 6.8 – Накладные затраты

Наименование	Итого в месяц, руб.
Аренда помещения	80 000
Командировочные (директор)	40 000
Прочие затраты (коммунальные услуги, канцелярия)	50 000
Аренда сервера, домена	10 000
Реклама продукта	50 000
Итого	230 000

Исходя из таблицы 6.8 следует, что накладные расходы составили 320 000 руб.

Следующим шагом следует расчёт прибыли проекта. В приложении Г представлена таблица расчета прибыли. В начале была определена выручка от продаж 8 500 подписок за 12 месяцев, которая составила 76 500 000 руб. и переменные расходы на разработку – 7 577 490 руб. Далее были определены постоянные (накладные расходы), которые составили 230 000 руб. Исходя из

этого была найдена прибыль EBITDA (прибыль до вычета процентов, налогов, износа и амортизации) и она составляет 4 841 500 руб.

Расчет точки безубыточности происходит по следующей формуле:

$$Q = \frac{C_{\text{пост}}}{(P - C_{\text{пер}})} \quad (6.2)$$

где $C_{\text{пост}}$ – затраты постоянные,

P – цена подписки,

$C_{\text{пер}}$ – переменные расходы

$$Q = \frac{1458500}{9000} = 163$$

Таким образом точка безубыточности составляет 163 подписки в месяц.

6.7 Обоснование эффективности проекта

Для определения эффективности проекта были проведены расчеты денежных потоков, представленные в приложении В. NPV (чистая приведенная стоимость) составило 24 346 194 руб. что означает, что инвестиции в данный проект приносят высокий доход и его стоит реализовывать, IRR (внутренняя норма доходности) составило 20%, что больше ставки дисконтирования, и говорит о высокой эффективности реализации проекта. Следует упомянуть, что период окупаемости и дисконтированный период окупаемости составляет 4 месяца и при этом индекс рентабельности равен 3,4. Таким образом каждый вложенный рубль в данный проект будет приносить 3,4 руб. дохода.

Для успешной реализации проекта проработка бизнес-модели является ключевой, особенно для описания бизнес-плана и дальнейшего продвижения на рынке.

Для описания бизнес-модели было принято решение взять схему бизнес -модели по А. Остервальдеру, так как она наглядно описывает перспективы развития и ключевые точки роста компании. В отличие от

многостраничного бизнес-плана, такой документ легко читается и поддается быстрому анализу.

На рисунке 6.1 описана бизнес-модель по А. Остервальдеру.

<i>Ключевые партнеры</i> Стратегическое сотрудничество со следующими партнерами: поставщикам и средств индивидуальной защиты; аутсорсинговые компании.	<i>Ключевые виды деятельности</i> Производство <u>Разработать web-платформу по охране труда, которая бы упростила работу с документами в сфере охраны труда.</u> Решение <u>проблем Уменьшение времени на создание документов в сфере охраны труда</u>	<i>Ценностные предложения</i> 1. Уменьшение временных затрат на создание документов по охране труда; 2. Уменьшение ошибок в создании документов по охране труда; 3. Уменьшение затрат на охрану труда;	<i>Взаимоотношения с клиентами</i> 1. Постоянная техническая и консультационная поддержка клиентов; 2. Обратная связь; 3. Услуги аутсорсинга по охране труда	<i>Потребительские сегменты</i> 1. Малые предприятия, в которых требуются документы по охране труда; 2. Аутсорсинг компании по охране труда;
	<i>Ключевые ресурсы</i> -		<i>Каналы сбыта</i> 1. Прямые продажи; 2. Продажа через посредников.	
<i>Структура издержек</i> Фиксированные издержки – заработная плата работникам, налоги, аренда. Переменные издержки – расходы на аренду сервера.		<i>Потоки поступления доходов</i> 1. Доход от продажи подписки и аутсорсинг заказов		

Рисунок 6.1 – Бизнес модель по А. Остервальдеру

Заключение

В ходе данной работы была выявлена проблема в документообороте в сфере охраны труда. Была разработана база данных, серверная часть и клиентская часть WEB-платформы для автоматизации процессов в сфере охраны труда и промышленной безопасности.

Была разработана архитектура приложения, которая соответствует современным требованиям разработки web-платформы. Выбранная архитектура способна масштабироваться без применения больших усилий и показала свою надежность в эксплуатации на основе других архитектур клиент-серверного приложения.

Был разработан дизайн web-платформы, который помогает новым пользователям быстро адаптироваться к приложению, также приложение имеет возможность адаптивно отображаться на экранах мобильных устройств, что позволяет пользователям работать в любой точке предприятия.

Также был проведен бизнес анализ, который показал хорошие показатели окупаемости продукта, а точка безубыточности равна 163 проданным подпискам в месяц. Это показывает, что чтобы приложению выйти в плюс в месяц требуется продать 163 подписки. Также для разработанной web-платформы был определен период окупаемости, который составил 4,5 месяца. Данные показатели демонстрируют, что web-платформа имеет хорошую рентабельность.

Данная WEB-Платформа была установлена на удаленном сервере. WEB-платформа была введена в эксплуатацию в компании ООО НПК “ТЕСАРТ” и продемонстрировала свою эффективность. Внедрение платформы дало компании возможность оптимизировать процессы документооборота.

В ходе эксплуатации были выявлены следующие преимущества использования данной WEB-платформы:

- Автоматизация процессов: Сокращение ручного труда и снижение вероятности ошибок в документации;

- Экономия ресурсов: Снижение затрат на управление охраной труда и уменьшение административной нагрузка;
- Доступность: Возможность доступа к платформе с любого устройства с интернет-соединением, что обеспечивает гибкость и удобство использования.
- Соответствие требованиям: Обеспечение актуальности документации и соблюдение всех законодательных норм и требований в области охраны труда.
- Повышение эффективности: Улучшение общей организации работы и повышение производительности труда за счет оптимизации процессов.

Данная работа показала, что внедрение автоматизированных систем управления документооборотом в сфере охраны труда и промышленной безопасности является решением, способствующим повышению эффективности управления и снижению административных затрат. Разработанная WEB-Платформа доказала свою практическую ценность и может быть рекомендована для широкого использования в малых и средних предприятиях.

Список использованных источников

1. Трудовой кодекс Российской Федерации N 197-ФЗ (ред. от 06.04.2024). М.: Эксмо-Пресс, 2023. 320 с.
2. Антипенко А.А. Исследование уровня технологического развития в России // Вестник ОмГУ. Серия: Экономика. 2021. №3. С. 5-15.
3. Бавыкина Е.Н. Совершенствование системы охраны труда на предприятии // РППЭ. 2021. №8 (130). С. 107-113.
4. Приказ Минтруда России от 29.10.2021 N 766н [Электронный ресурс]: КонсультантПлюс. URL: https://www.consultant.ru/document/cons_doc_LAW_405210 (дата обращения: 10.02.2024).
5. Промышленное производство в России [Электронный ресурс]: официальный сайт Федеральной службы государственной статистики <https://rosstat.gov.ru/folder/210/document/13225> (дата обращения: 10.02.2024).
6. Трудовой Кодекс РФ [Электронный ресурс]: КонсультантПлюс. URL: https://www.consultant.ru/document/cons_doc_LAW_34683/4fe318e6d09155659a4381ef26a85e7df9ebcf94 (дата обращения: 10.03.2024).
7. Система управления базами данных: что это и зачем она нужна [Электронный ресурс]: Skillbox Media. URL: <https://skillbox.ru/media/code/sistema-upravleniya-bazami-dannykh-cto-eto-takoe-i-zachem-ona-nuzhna/> (дата обращения: 14.03.2024)
8. О PostgreSQL [Электронный ресурс]: официальный сайт СУБД PostgreSQL. URL: <https://www.postgresql.org/> (дата обращения: 15.03.2024).
9. The 15 Most Popular Programming Languages of 2023 [Электронный ресурс] URL: <https://www.hackerrank.com/blog/most-popular-languages-2023> (дата обращения: 28.02.2024).
10. Who Uses Go? Companies That Use Go and What Go Is Used For [Электронный ресурс] URL: <https://careerkarma.com/blog/who-uses-go> (дата обращения: 28.02.2024).

11. Why Go? Use Cases [Электронный ресурс]: официальный сайт языка программирования Golang. URL: <https://go.dev/solutions/use-cases> (дата обращения: 15.03.2024).
12. Go best practices, six years in [Электронный ресурс]: Peter Bourgon. URL: <https://peter.bourgon.org/go-best-practices-2016/#repository-structure%22> (дата обращения: 01.05.2024).
13. Что такое GraphQL [Электронный ресурс]: Хабр – Сообщество IT специалистов. URL: <https://habr.com/ru/articles/765064> (дата обращение 01.05.2024)
14. GraphQL learning [Электронный ресурс]: Официальный сайт GraphQL. URL: <https://graphql.org/learn> (дата обращения: 01.05.2024)
15. Зачем фронтендерам React, если есть JavaScript [Электронный ресурс]: HTML Academy: интерактивные онлайн-курсы. URL: <https://htmlacademy.ru/blog/js/react-js> (дата обращения: 15.03.2024).
16. GraphQL learning [Электронный ресурс]: Официальный сайт GraphQL. URL: <https://graphql.org/learn> (дата обращения: 01.05.2024)
17. Stoyan S. React: Up & Running / Sebastopol: O'Reilly Media, 2021. 233 p.
18. Komperla V., Deenadhayalan P,m Ghuli P. Pattar R. React: A detailed survey // Indonesian Journal of Electrical Engineering and Computer Science. 2022. P. 1710-1717.
19. Palak D., Abhishek K.J. ReactJS for trading applications / Dubai: IEEE 2022. 5 p.
20. Engineering International [Электронный ресурс]: Front-End Development in React. URL: <https://abc.us.org/ojs/index.php/ei/article/view/662> (дата обращения: 15.03.2024).
21. ReactJS for Trading Applications [Электронный ресурс]: IEEEExplore. URL: <https://ieeexplore.ieee.org/document/9995932> (дата обращения: 15.03.2024)

22. Артур О., Георг К. Информационные технологии и разработка одностраничника веб-приложение с использованием библиотеки react, 2022. С. 37–38.

23. Чен С., Тадури У.Р., Венката К.Р. Фронтенд-разработка в React // Engineering International, 2019. – С. 119–120.

24. Client-side Storage [Электронный ресурс]: Изучение веб-разработки| MDN. URL: https://developer.mozilla.org/ru/docs/Learn/JavaScript/Client-side_web_APIs/Client-side_storage (дата обращения: 09.02.2024).

25. Малое и среднее предпринимательство в России [Электронный ресурс]: Федеральная служба государственной статистики. – Москва, 2022. URL: <https://rosstat.gov.ru/folder/210/document/13223> (дата обращения: 03.03.2024).

26. Количество субъектов МСП в РФ выросло в 2023 году на 6% и достигло 6,3 млн. [Электронный ресурс]: официальный сайт Министерство экономического развития Российской Федерации. URL: https://www.economy.gov.ru/material/news/tatyana_ilyushnikova_kolichestvo_subektov_msp_v_rf_vyroslo_v_2023_godu_na_6_i_dostiglo_63 mln.html (дата обращения: 05.03.2024)

Приложение А

(обязательное)

Пример инициализации подключения к БД и запрос на получение данных

```
// InitPSQL – инициализирует подключение к базе данных и создает необходимые
таблицы.
// Аргументы:
// - dbLogin: логин пользователя базы данных.
// - dbPassword: пароль пользователя базы данных.
// - dbName: имя базы данных.
// - dbHost: хост базы данных.
func InitPSQL(dbLogin string, dbPassword string, dbName string, dbHost string)
{
    dsn := fmt.Sprintf(
        "sslmode=disable user=%v password=%v dbname=%v host=%v",
        dbLogin,      // Login для подключения к PostgreSQL
        dbPassword,   // Password для подключения к PostgreSQL
        dbName,       // Имя базы данных
        dbHost,       // Хост для подключения к СУБД
    )
    log := logging.Sugar()
    client, err := gorm.Open(postgres.Open(dsn), &gorm.Config{
        Logger: logging.IntegrationGorm(),
    })
    if nil != err {
        log.Fatalf("Не удалось подключиться к базе данных psql: %v", err)
    }
    sqlDataBase, err := client.DB()
    if nil != err {
        log.Fatalf("Ошибка инициализации базы данных psql: %v", err)
    }
    if sqlDataBase.Ping() != nil {
        Close()
        log.Fatalf("Неудачная попытка подключиться к базе данных psql: %v",
err)
    }
    checkingTables(client)
    database.gorm = client
    database.psql = sqlDataBase
}
```



```
}
```

```
// GetOprCard получение карточки для документов
```

```
func GetOprCard(orgID int64, internalID uint16) ([]models.OprCard, error) {  
    o := []models.OprCard{}  
    err := database.gorm.Find(&o).Where("org_id = ? AND person_id = ?",  
orgID, internalID).Error  
    if err != nil {  
        log.Print("Ошибка получения данных")  
    }  
    return o, nil  
}
```

Приложение Б

(обязательное)

Пример инициализации логгера и его функции

```
// Init создает и возвращает новый экземпляр логгера.
func Init(debug bool) {
    var err error
    globalLogger, err = config.Logger(debug).Build()
    if err != nil {
        panic(fmt.Sprintf("Ошибка при инициализации логгера: %v", err))
    }
}

// Close останавливает и синхронизирует глобальный логгер.
func Close() {
    if globalLogger == nil {
        return
    }
    globalLogger.Sync()
}

// Sugar возвращает сахаристый логгер, который обортывает глобальный логгер.
func Sugar() *zap.SugaredLogger {
    return globalLogger.Sugar()
}

// Info логирует сообщение на уровне InfoLevel. Сообщение включает любые поля,
// переданные на месте логирования, а также любые поля, накопленные в логгере.
func Info(msg string, fields ...zap.Field) {
    globalLogger.Info(msg, fields...)
}

// Error логирует сообщение с уровнем Error. Сообщение включает любые поля,
// переданные на месте вызова логирования, а также любые поля, накопленные в
// логгере.
func Error(msg string, fields ...zap.Field) {
    globalLogger.Error(msg, fields...)
}

// Debug записывает отладочное сообщение на уровне DebugLevel.
// Сообщение включает любые поля, переданные на месте вызова логирования, а
// также любые поля, накопленные в логгере.
func Debug(msg string, fields ...zap.Field) {
    globalLogger.Debug(msg, fields...)
}
```

```
}  
// Warn регистрирует сообщение с уровнем WarnLevel. Сообщение включает все  
поля, переданные на месте регистрации, а также все накопленные поля на логгере.  
func Warn(msg string, fields ...zap.Field) {  
    globalLogger.Warn(msg, fields...)  
}  
// Panic логирует сообщение на уровне PanicLevel.  
// Сообщение включает любые поля, переданные на месте логирования, а также  
любые поля, накопленные в логгере.  
func Panic(msg string, fields ...zap.Field) {  
    globalLogger.Panic(msg, fields...)  
}
```

Приложение В

(обязательное)

Результат расчётов прибыли и план работы подчинённых

Таблица В.1 – Расчёт прибыли

Показатель	1	2	3	4-10	11	12
Выручка без НДС, в руб.	900 000	1 800 000	4 500 000	6 300 000	6 300 000	6 300 000
Постоянные (накладные) расходы без учета амортизации, в руб.	230 000	230 000	230 000	230 000	230 000	230 000
Заработная плата, в руб.	1 228 500	1 228 500	1 228 500	1 228 500	1 228 500	1 228 500
EBITDA, в руб.	-558 500	341 500	3 041 500	4 841 500	4 841 500	4 841 500
Амортизация новых основных средств (инвестиции) , в руб.	7 433	7 433	7 433	7 433	7 433	7 433
Прибыль до налогообложения, в руб.	-565 933	334 067	3 034 067	4 834 067	4 834 067	4 834 067
Налог на прибыль (ставка 20%), в руб.	-113 186,6	66813,4	606813,4	966813,4	966813,4	966813,4
Чистая прибыль проекта, в руб.	-452 746	267 254	2 427 254	3 867 254	3 867 254	3 867 254

Таблица В.2 – Расчёт денежных потоков и показателей эффективности проекта

Показатель	0 месяц	1 месяц	2 месяц	3 месяц	4-10 месяцы	11 месяц	12 месяц
Денежный поток по операционной деятельности (CF), руб.		-445 313	274 687	2 434 687	3 874 687	3 874 687	3 874 687
Чистая прибыль проекта, руб.		-452 746	267 254	2 427 254	3 867 254	3 867 254	3 867 254
Амортизация новых основных средств (инвестиции) руб.		7433	7433	7433	7433	7433	7433
Денежный поток по инвестиционной деятельности (IC), руб.	-7 577 490						
Инвестиционные затраты без НДС, руб.	-7 577 491						
Свободный денежный поток (FCF), руб.	-7 577 492	-445 313	274 687	2 434 687	3 874 687	3 874 687	3 874 687
Свободный денежный поток нарастающим итогом (FCF), руб.	-7 577 493	-8 022 806	-7 748 120	-5 313 433	-1 438 747 – 21 809 373	25 684 060	29 558 746
Ставка дисконтирования	2%						
Коэффициент дисконтирования		0,980	0,961	0,942	0,923 – 0,820	0,804	0,788
Дисконтированный денежный поток от операционной деятельности, руб.		-436 581	264 020	2 294 259	3 579 611 – 3 178 592	3 116 267	3 055 163
Дисконтированный свободный денежный поток нарастающим итогом, руб.	-7 577 493	-8 014 074	-7 750 054	-5 455 795	-1 876 183 – 18 174 762	21 291 030,09	24 346 194,03

Окончание таблицы В.2

NPV	19 422 507	–	–	–	–	–	–
IRR	20%	–	–	–	–	–	–
Период окупаемости, мес.	4,371	–	–	–	–	–	–
Дисконтированный период окупаемости, мес.	4,535	–	–	–	–	–	–
PI	4,21	–	–	–	–	–	–

Таблица В.3 – План работы разработчиков

Показатель	1 месяц	2 месяц	3-4 месяц	5-6 месяц	7 месяц	8 месяц
Product менеджер	Создание требований	Проектирование архитектуры	Руководство командами	Руководство командами	Руководство командами	-
Аналитик	Создание требований	-	-	-	Аналитика	Аналитика
UI/UX дизайнер	-	Разработка дизайна	-	-	-	-
Frontend разработчики	-	-	Разработка приложения	Разработка приложения	-	-
Backend разработчики	-	-	Разработка приложения	Разработка приложения	-	-

Окончание таблицы В.3

Тестировщик	-	-	-	-	Тестирование приложения	
DevOps инженер	-	-	-	-	-	Загрузка приложения на сервер
Консультант в сфере охраны труд	Консультация	Консультация	Консультация	Консультация	Консультация	Консультация
Стоимость, руб.	221 000	224 000	1 118 000	1 118 000	299 000	247 000

