АННОТАЦИЯ

Исполнитель: студент группы ИКБО-02-20 Сорокин А.А.

Руководитель: заведующий базовой кафедры №231, Стариков П.П.

Выпускная квалификационная работа на тему «Разработка системы расчета логистики в условиях чрезвычайных ситуаций».

Общее количество страниц: 72.

Работа включает в себя 25 рисунков, 12 таблиц, 8 приложений.

Список источников информации включает 20 позиций.

Ключевые слова: логистика, чрезвычайная ситуация, инцидент, паттерн, поиск предметов.

СОДЕРЖАНИЕ

ПЕРЕЧЕНЬ СОКРАЩЕНИЙ И ОБОЗНАЧЕНИЙ 7

ВВЕДЕНИЕ 8

1 ИССЛЕДОВАТЕЛЬСКИЙ РАЗДЕЛ 11

1.1 Характеристика предметной области 11

1.2 Обзор актуальности разработки 12

1.3 Обзор существующих решений 12

Выводы по исследовательскому разделу 15

2 АНАЛИТИЧЕСКИЙ РАЗДЕЛ 16

2.1 Анализ существующих программных комплексов 16

2.1.1 Достоинства и недостатки систем 16

2.1.2 Общие требования к системе 17

2.2 Выбор общего подхода к реализации приложения 18

2.3 Формулировка требований к прототипу приложения 19

2.3.1 Функциональные требования 19

2.3.2 Нефункциональные требования 20

Выводы по аналитическому разделу 21

3 ТЕХНОЛОГИЧЕСКИЙ РАЗДЕЛ 23

3.1 Проектирование архитектуры, структуры базы данных 23

3.1.1 Архитектурный стиль 23

3.1.2 Клиент-серверный стиль 23

3.1.3 Архитектурные контексты 24

3.1.4 Описание модели данных 25

3.1.5 Функциональная модель 27
3.2 Выбор средств реализации 29
3.2.1 Выбор средств разработки клиентской части 29
3.2.2 Выбор средств разработки серверной части 30
3.2.3 База данных 32
3.3 Реализация прототипа приложения 33
3.3.1 Разработка клиентской части прототипа приложения 33
3.3.1.1 Модуль складов 33
3.3.1.2 Модуль паттернов 34
3.3.1.3 Модуль инцидентов 34
3.3.2 Разработка серверной части прототипа приложения 35
3.3.2.1 Общий подход к разработке серверной части 35
3.3.2.2 Алгоритм расчёта логистики 35
3.3.3 Пример использования прототипа приложения 37
3.4 Описание процесса взаимодействия с прототипом 41
Выводы по технологическому разделу 42
4 ЭКОНОМИЧЕСКИЙ РАЗДЕЛ 43
4.1 Организация и планирование работ 43
4.2 Расчет затрат на проведение работ 45
4.2.1 Статья «Основная заработная плата» 46
4.2.2 Статья «Дополнительная заработная плата» 47
4.2.3 Статья «Страховые взносы» 47
4.2.4 Статья «Материалы, покупные изделия и полуфабрикаты» 48

4.2.5 Статья «Расходы на приобретение компьютерной техники и программного обеспечения» 48
4.2.6 Статья «Амортизационные отчисления» 49
4.2.7 Статья «Эксплуатационные расходы, связанные с использованием компьютерной техники» 50
4.2.8 Статья «Накладные расходы» 50
4.2.9 Статья «Прочие расходы» 50
4.2.10 Полная себестоимость проекта 51
Выводы по экономическому разделу 51
ЗАКЛЮЧЕНИЕ 53
СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 56
ПРИЛОЖЕНИЕ А 59
ПРИЛОЖЕНИЕ Б 62
ПРИЛОЖЕНИЕ В 65
ПРИЛОЖЕНИЕ Г 67
ПРИЛОЖЕНИЕ Д 69
ПРИЛОЖЕНИЕ Е 70
ПРИЛОЖЕНИЕ Ж 71
ПРИЛОЖЕНИЕ З 72

ПЕРЕЧЕНЬ СОКРАЩЕНИЙ И ОБОЗНАЧЕНИЙ

БД	–	База данных
ИНН	–	Идентификационный номер налогоплательщика
ИС	–	Информационная система
ПК	–	Персональный компьютер
ПО	–	Программное обеспечение
СУБД	–	Система управления базой данных
ЧС	–	Чрезвычайная ситуация
BPMN	–	Business Process Model and Notation (нотация и модель бизнес-процессов)
CSS	–	Cascading Style Sheets (каскадные таблицы стилей)
ER	–	Entity-relationship (сущность-связь)
HTML	–	HyperText Markup Language (язык гипертекстовой разметки)
IDEF0	–	ICAM DEFinition for Function Modeling (методология функционального моделирования)
MVC	–	Model View Controller (модель представление контроллер)
SQL	–	Structured Query Language (язык структурированных запросов)

ВВЕДЕНИЕ

В современном мире, где глобализация и технологический прогресс становятся все более важными, логистика играет ключевую роль в обеспечении эффективности и надежности бизнес-процессов. Однако, в условиях ЧС, таких как стихийные бедствия, пандемии или военные конфликты, традиционные системы логистики могут столкнуться с серьезными проблемами. В связи с этим, разработка системы расчета логистики, способной адаптироваться и функционировать в условиях ЧС, является актуальной и важной задачей.

Цель данной выпускной квалификационной работы - разработать информационную систему для расчёта логистики в условиях ЧС.

Объект - система расчета логистики.

Предметом исследования выступает обеспечение логистики в условиях ЧС.

ВКР состоит из 4 разделов, в рамках которых нужно реализовать следующие задачи:

1. Исследовательский:

- формирование характеристики предметной области;
- обзор актуальности разработки;
- обзор существующих решений.

2. Аналитический:

- анализ существующих программных комплексов;
- выбор и обоснование общий подход к реализации приложения;
- формирование требования к прототипу приложения.

3. Технологический:

- проектирование архитектуры приложения и структуру базы данных;
- выбор средства реализации прототипа приложения;

- реализация прототип приложения;
- описание процесса взаимодействия с прототипом.

4. Экономический:

- расчет стоимости проведения работ, связанных с реализацией проекта по проектированию и разработке прототипа приложения.

В исследовательском разделе описываются сфера деятельности, объект и предмет исследования, обосновывается актуальность разработки и рассматриваются существующие решения.

В аналитическом разделе проводится анализ существующих решений, выбираются и обосновываются общий подход для реализации приложения, а также формируются функциональные и нефункциональные требования к прототипу приложения.

В технологическом разделе разрабатывается архитектура приложения, производится выбор технологий для реализации, реализуется прототип приложения, а также описывается процесс взаимодействия с прототипом веб-приложения.

В экономическом разделе производится расчет стоимости выполнения работ.

В процессе написания выпускной квалификационной работы автор руководствовался следующими нормативными актами:

1. Федеральный закон "О защите населения и территорий от чрезвычайных ситуаций природного и техногенного характера" (ФЗ №68 от 21.12.1994).
2. Постановление Правительства РФ "Об утверждении Положения о единой государственной системе предупреждения и ликвидации чрезвычайных ситуаций" (Постановление №794 от 30.12.2003).
3. Федеральный закон "О гражданской обороне" (ФЗ №28 от 12.02.1998).

4. Государственный стандарт "ГОСТ Р 22.3.05-97. Безопасность в чрезвычайных ситуациях. Логистика. Основные положения".
5. Федеральный закон "О безопасности" (ФЗ №390 от 28.12.2010).
6. Федеральный закон "О защите населения от чрезвычайных ситуаций и ликвидации их последствий" (ФЗ №68-ФЗ от 21.12.1994, ред. от 31.12.2022).
7. Приказ Министерства здравоохранения РФ от 15 февраля 2013 г. № 70н "Об утверждении порядка взаимодействия медицинских организаций при оказании медицинской помощи пострадавшим при чрезвычайных ситуациях".
8. СанПиН 2.2.2/2.4.1340-03 «Гигиенические требования к персональным электронно-вычислительным машинам и организации работы».
9. Трудовой кодекс Российской Федерации от 30.12.2001 № 197-ФЗ.
10. Приказ Минздравсоцразвития РФ от 04.05.2012 № 477н «Об утверждении перечня состояний, при которых оказывается первая помощь, и перечня мероприятий по оказанию первой помощи».

1 ИССЛЕДОВАТЕЛЬСКИЙ РАЗДЕЛ

1.1 Характеристика предметной области

Предметная область данной работы – система расчета логистики в условиях ЧС. Логистика в сфере экстренных ситуаций, таких как стихийные бедствия, медицинские кризисы, военные конфликты, террористические угрозы или другие ЧС, играет критически важную роль в эффективной доставке необходимых ресурсов и материальных благ. Расчёт системы логистики в этих условиях становится жизненно важным делом.

Одной из ключевых проблем, с которой сталкиваются системы логистики в условиях ЧС, является необходимость адаптации к переменным и нестандартным условиям [1]. Традиционные системы могут быть неспособны эффективно и быстро реагировать на изменения в снабжении, например, если открывается временный склад снабжения на новом месте. Это может привести к прерывистости в поставках или даже к недостаточному или несвоевременному обеспечению необходимых материалов и помощи, что приведёт к ужасным последствиям.

Разрабатываемая система расчета логистики в условиях ЧС направлена на решение данных проблем. Она ориентирована на создание структуры, способной эффективно адаптироваться к новым условиям и управлять логистическими операциями в условиях ЧС, оказывая одноразовую помощь. Основной целью является обеспечение расчёта логистических процессов в чрезвычайных ситуациях для оперативного реагирования и предоставления необходимой помощи и ресурсов.

1.2 Обзор актуальности разработки

Система расчета логистики в условиях чрезвычайных ситуаций является крайне актуальной в современном мире. В условиях постоянно меняющихся и непредсказуемых обстоятельств эффективное управление логистикой становится жизненно важным.

В условиях ЧС, быстрый и эффективный отклик может спасти жизни или инфраструктуру. Система расчета логистики позволяет оптимизировать процесс предоставления необходимых ресурсов, таких как медицинское оборудование, продовольствие, вода, спасательное оборудование и т.д. в зоны ЧС.

Традиционные логистические системы могут быть неспособны быстро адаптироваться к изменяющимся условиям. Система расчета логистики в условиях ЧС предназначена для работы в условиях неопределенности и изменчивости, что позволяет ей быстро реагировать на изменения.

Эффективное управление логистикой может помочь снизить затраты, связанные с предоставлением ресурсов, а также уменьшить потери от недостаточного или несвоевременного обеспечения.

1.3 Обзор существующих решений

Рассматриваемая предметная область специфична, и не существует конкретных аналогов. Существуют такие приложения, как «MercuryGate» [2] (Рисунок 1), «Relog» [3] (Рисунок 2) или «1С:Предприятие 8. 1С-Логистика:Управление складом 3.01» [4] (Рисунок 3).

MercuryGate - это комплексная система управления транспортировкой (TMS), которая предлагает планирование и выполнение перевозок, управление претензиями, оптимизацию мультимодальных перевозок, видимость

отправлений и заказов, а также управление таможенным оформлением и торговым соблюдением. MercuryGate фокусируется на снижении затрат, упрощении процессов и уменьшении углеродного следа, предлагая решения для всех видов транспорта, включая дорожный, морской, железнодорожный и воздушный.

Relog - это ПО для планирования маршрутов междугородних перевозок, которое использует мощные алгоритмы искусственного интеллекта для управления доставкой. Relog предлагает рынок водителей, быстрый алгоритм маршрутизации, гибкость, а также аналитику и интуитивно понятный интерфейс.

1С:Предприятие 8. 1С-Логистика:Управление складом 3.01 - это решение для управления бизнесом, управление продажами, закупками, складом, персоналом, финансами и бизнес-аналитикой. Оно предназначено для малого бизнеса и позволяет управлять различными аспектами бизнеса в одной программе.

Все они больше сфокусированы на задачи логистики или управления складом, но при этом не обладают способностью адаптироваться к нестабильным сценариям, например, когда постоянно появляются и удаляются временные склады. Так же они не обладают возможностью динамического определения необходимых ресурсов сразу на нескольких складах для прямой доставки, они способны осуществлять прямую доставку ресурсов на точку только с 1 конкретного склада.

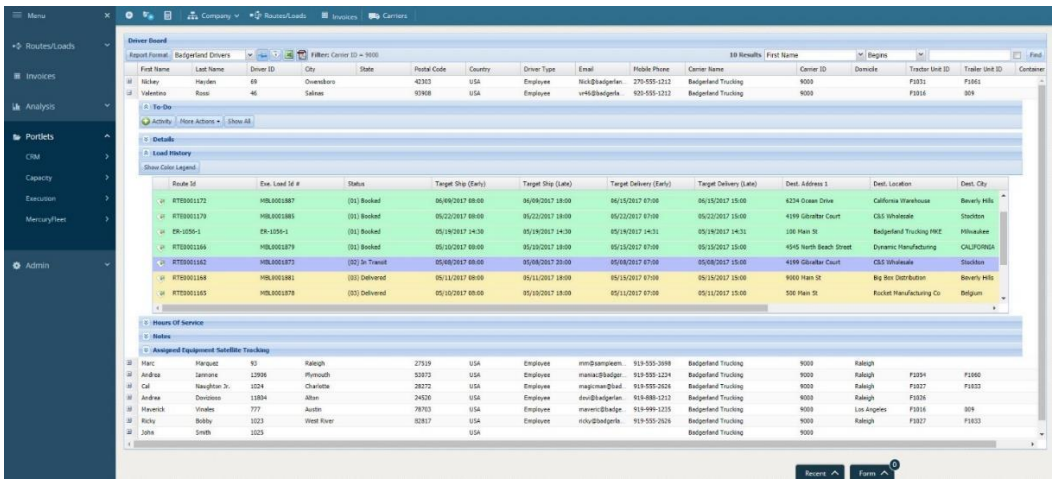

Рисунок 1 – Общий вид программы «MercuryGate»

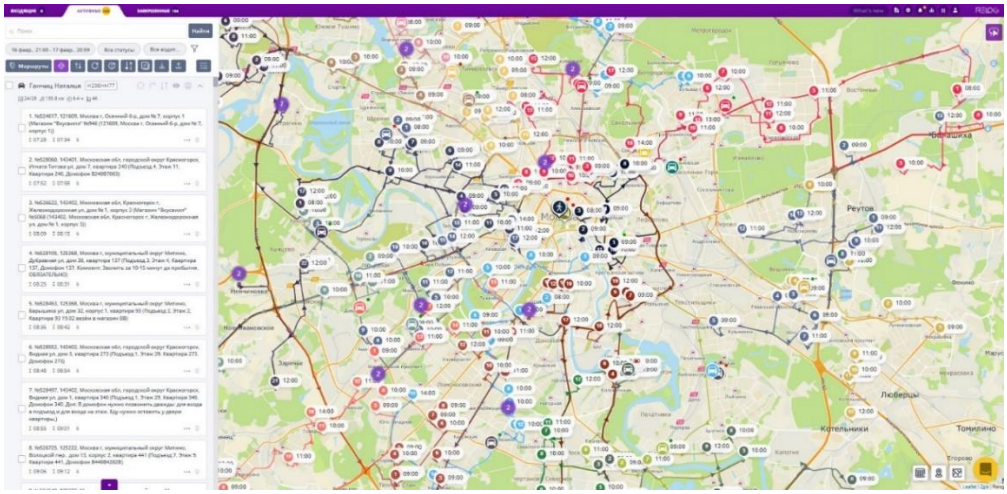

Рисунок 2 – Общий вид программы «Relog»

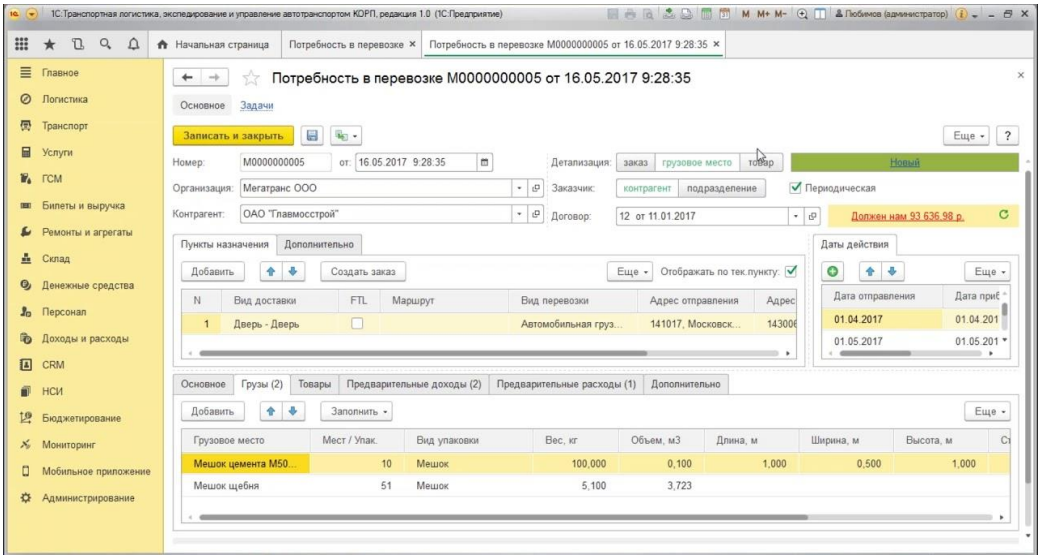

Рисунок 3 – Общий вид программы «1С:Предприятие 8. 1С-Логистика:Управление складом 3.01»

Выводы по исследовательскому разделу

Исследовательский раздел обозначает актуальность и необходимость разработки системы расчета логистики в условиях ЧС.

Обзор актуальности подчеркивает жизненную важность эффективного управления логистикой в условиях постоянно меняющихся и непредсказуемых обстоятельств.

Таким образом, разработка системы расчета логистики в условиях ЧС становится важной в обеспечении быстрого необходимого реагирования на экстренные ситуации в условиях неопределенности и изменчивости.

2 АНАЛИТИЧЕСКИЙ РАЗДЕЛ

2.1 Анализ существующих программных комплексов

2.1.1 Достоинства и недостатки систем

Выявим достоинства и недостатки систем, представленных в пункте 1.3, чтобы выделить необходимые требования к приложению (пункт 2.1.2).

MercuryGate:

Достоинства:

- Хорошая гибкость и адаптивность к изменениям, что позволяет эффективно реагировать на динамичные сценарии;
- Уровни доступа в системе - обеспечивает гибкую систему управления уровнями доступа, что способствует безопасной работе.

Недостатки:

- Отсутствие геолокационной интеграции может быть недостаточным для некоторых логистических операций;
- Отсутствие возможности прямой доставки в 1 точку с нескольких складов;
- Интерфейс может быть не столь интуитивно понятным, что может потребовать дополнительного времени на освоение.

Relog:

Достоинства:

- Обладает гибкостью для адаптации к переменным условиям бизнеса;
- Имеет простой и интуитивно понятный интерфейс, что упрощает работу с системой;

- Предоставляет гибкую систему управления доступом для обеспечения безопасности.

Недостатки:

- Отсутствие геолокационной интеграции может быть ограничивающим для определенных логистических задач;
- Отсутствие возможности прямой доставки в 1 точку с нескольких складов.

1С:Предприятие 8. 1С-Логистика:Управление складом 3.01:

Достоинства:

- Богатый функционал, который позволяет более точно настраивать необходимые логистические процессы.

Недостатки:

- Не обладает достаточной гибкостью для эффективной адаптации к изменяющимся требованиям;
- Отсутствие геолокационной интеграции ограничивает возможности для оптимизации логистических процессов;
- Интерфейс может быть не очень удобным и требовать времени на освоение;
- Отсутствие возможности прямой доставки в 1 точки с нескольких складов.

2.1.2 Общие требования к системе

На основе анализа аналогов в пункте 2.1.1 были определены основные функции и требования, которые необходимо реализовать и соблюсти:

- предоставлять возможность пользователю авторизироваться в системе;

- предоставлять возможность администратору зарегистрировать нового пользователя;
- разделять пользователей по уровню доступа к системе и предоставлять только доступный им функционал;
- иметь геолокационную интеграцию;
- предоставлять пользователю возможность внести данные о складе;
- предоставлять пользователю возможность внести данные о происшествии;
- предоставлять пользователю возможность получить информацию о реакции на происшествие;
- предоставлять пользователю возможность внести данные о паттерне по реакции на происшествие;
- отображать полученные результаты работы;
- предоставлять пользователю возможность автоматической отправки результата работы системы во внешнюю систему.

2.2 Выбор общего подхода к реализации приложения

Определим основные подходы к реализации прототипа приложения.

В качестве архитектуры будет использована клиент-серверная архитектура. Выбрана она была за простоту в использовании и развёртывании, а также за высокие возможности масштабируемости, что важно в подобных системах.

В качестве основного языка программирования будет использован JavaScript, так как он обладает богатой экосистемой инструментов и библиотек. А также за возможность использования одного языка в обеих частях приложения (фронтенд и бекенд частях). Это упрощает разработку и

поддержку кода.

В качестве картографической основы приложения будет использованы карты OpenStreetMap (OSM) [5]. Они являются открытыми и богатыми на точные данные с подробной информацией о дорожной инфраструктуре, маршрутах и объектах на карте. Так же OSM не предоставляет жёстких требований к использованию своих карт, в отличии от Яндекс Карт[6] или 2gis[7], что и стало решающим фактором в выборе данной картографической основы. А интеграция с GraphHopper [8] позволяет обеспечить приложение функциональностью маршрутизации и геолокации.

Выбор общего подхода к реализации приложения основан на стремлении к построению гибкой, масштабируемой и эффективной системы, которая будет соответствовать требованиям, прописанным в пункте 2.1.2.

2.3 Формулировка требований к прототипу приложения

2.3.1 Функциональные требования

Для описания функциональных требований к информационной системе, воспользуемся пользовательскими историями — они помогут лучше понять, какой функционал необходим в зависимости от роли пользователя. Ниже приведены роли в разрабатываемой системе.

Роли в системе:

1. Старший администратор.
2. Администратор.
3. Старший пользователь.
4. Пользователь.

В таблице А.1 в приложении А приведены пользовательские истории.

2.3.2 Нефункциональные требования

Требования безопасности:

- Доступ к системе должен предоставляться по логину и паролю;
- Пользовательские пароли должны храниться в захешированном виде;
- Доступ с должен быть ограничен у конкретных пользователей к модулям системы;
- Должна быть возможность полного запрета доступа пользователя к системе;
- Возможности пользователя в системе должны быть ограничены в соответствии с его статусом в ней;
- Система должна иметь защиту от SQL инъекций.

Доступность:

- Доступ к системе обеспечивается на ПК, имеющим доступ в сеть интернет;
- Доступ обеспечивается через браузеры «Яндекс браузер» версии не ниже 24.4.2.956, «Google Chrome» версии не ниже 125.0.6422.77.

Программные требования для web-клиента системы:

- ОС семейства Linux (64-bit Ubuntu 18.04 (и выше), Debian 10 (и выше), openSUSE 15.2 (и выше) или Fedora Linux 32 (и выше)), семейства Windows (Windows 11, Windows 10, Windows 8.1, Windows 8, Windows 7), macOS 10.15 (и выше) (обосновано требованиями «Яндекс браузер» [9]);
- Наличие браузера «Яндекс браузер» версии не ниже 24.4.2.956, «Google Chrome» версии не ниже 125.0.6422.77;
- Поддержка выполнения JavaScript.

Аппаратные требования для web-клиента системы:

- ОЗУ: не менее 512 МБ (обосновано требованиями «Яндекс браузер» [9]);

- ПЗУ: не менее 600 МБ (обосновано требованиями «Яндекс браузер» [9]);
- Стабильное подключение к сети на скорости не менее 20 Мбит/с.

Программные требования для сервера системы:

- ОС семейства Linux, версия ядра не менее 5.15 [10], Windows (Windows 11, Windows 10);
- Установленный Node.js, версии не ниже 14.0;
- Установленный React, версии не ниже 18.0;
- Установленный npm, версии не ниже 6.0;
- Установленный PostgreSQL, версии не ниже 13.0.

Аппаратные требования для запуска среды выполнения:

- Центральный процессор с частотой от 1 ГГц (обосновано требованиями Node.js [6]);
- ОЗУ: не менее 512 МБ (обосновано требованиями Node.js [10]);
- ПЗУ: не менее 20 МБ (обосновано требованиями Node.js [10]);
- Стабильное подключение к сети на скорости не менее 100 Мбит/с.

Выводы по аналитическому разделу

Анализ существующих программных комплексов, проведенный в разделе 2.1, позволил выделить как достоинства, так и недостатки каждой из рассмотренных систем.

На основе выявленных достоинств и недостатков были сформулированы общие требования к системе в разделе 2.1.2, включая функциональные и нефункциональные требования.

Был выбран общий подход к реализации приложения в разделе 2.2 основан на стремлении к построению гибкой, масштабируемой и эффективной системы.

Таким образом, аналитический раздел позволил выделить ключевые аспекты разработки системы управления логистикой в условиях ЧС и определить основные требования к её реализации.

3 ТЕХНОЛОГИЧЕСКИЙ РАЗДЕЛ

3.1 Проектирование архитектуры, структуры базы данных

3.1.1 Архитектурный стиль

В качестве основного архитектурного стиля выбран паттерн MVC (Model-View-Controller) [11] для исключения множественных зависимостей, разделения логики проекта и разделения таких понятий как «логика» и «представление».

Компоненты MVC:

- Модель – отвечает за данные, логику их обработки, связь с внешними системами, такими как базы данных;
- Представление – отвечает за отображение результата работы и взаимодействие с пользователем;
- Контроллер – отвечает за связь между моделью и представлением. Он принимает действия от представления и решает, как система должна отреагировать на действие, выбирая необходимые модели.

3.1.2 Клиент-серверный стиль

Как уже говорилось в пункте 2.2, бы выбран клиент-серверный стиль архитектуры [12], который предполагает разделение системы на два или более компонента. Один из компонентов называется клиентом, который инициирует запросы к другому компоненту, называемому сервером, который обрабатывает запросы и возвращает ответы клиенту.

В разрабатываемом приложении это трехзвенная клиент-серверная архитектура [13], которая является разновидностью клиент-серверного стиля, в которой система разделена на три уровня:

1. Уровень клиента - включает в себя пользовательский интерфейс и обрабатывает все взаимодействия пользователя с системой. Он отвечает за взаимодействие с пользователем.

2. Уровень сервера - обрабатывает бизнес-логику приложения. Он получает запросы от клиента, выполняет необходимые операции, включая обработку данных, вычисления и доступ к ресурсам, и отправляет обратно клиенту результаты запроса. Здесь располагается ядро приложения, которое обрабатывает бизнес-процессы и принимает решения на основе поступающих данных.

3. Уровень сервера баз данных обеспечивает хранение, изменение, получение и удаление информации из базы данных. Здесь содержится вся необходимая для работы приложения информация.

3.1.3 Архитектурные контексты

Для описания архитектуры будет использоваться контекстуальный подход, включающий в себя следующие контексты:

- контекст данных (пункт 3.1.4);
- контекст функций (пункт 3.1.5);
- контекст логики взаимодействия (пункт 3.4).

Перечисленные архитектурные контексты будут рассмотрены далее.

3.1.4 Описание модели данных

Для описания модели данных использовалась нотация ER (Рисунок 4). Она состоит из 10 таблиц:

- users – информация о пользователе системы. В access указывается его доступ к системе. Пользователь может быть как физическим, так и юридическим лицом, поэтому у пользователя есть поле type. Для идентификации пользователя в personal_data записываются его уникальные данные: для физического лица – номер паспорта, а для юридического – ИНН организации;
- storages – таблица с информацией о складах;
- items – таблица с информацией о ресурсах;
- items_in_storages – таблица с информацией о наличии ресурсов на складах;
- patterns – таблица с информацией о типах происшествий и как о том, как реагировать на определённые типы происшествий;
- incidents – таблица с информацией о происшествиях: где и какие были, а так же результат обработки;
- settings – таблица с изменяемыми настройками системы;
- statuses – таблица сданными о существующих статусах в системе;
- items_types – таблица с типами предметов;
- external – таблица с информацией о внешних системах, включая url, данные для авторизации, для отправки результатов в них.

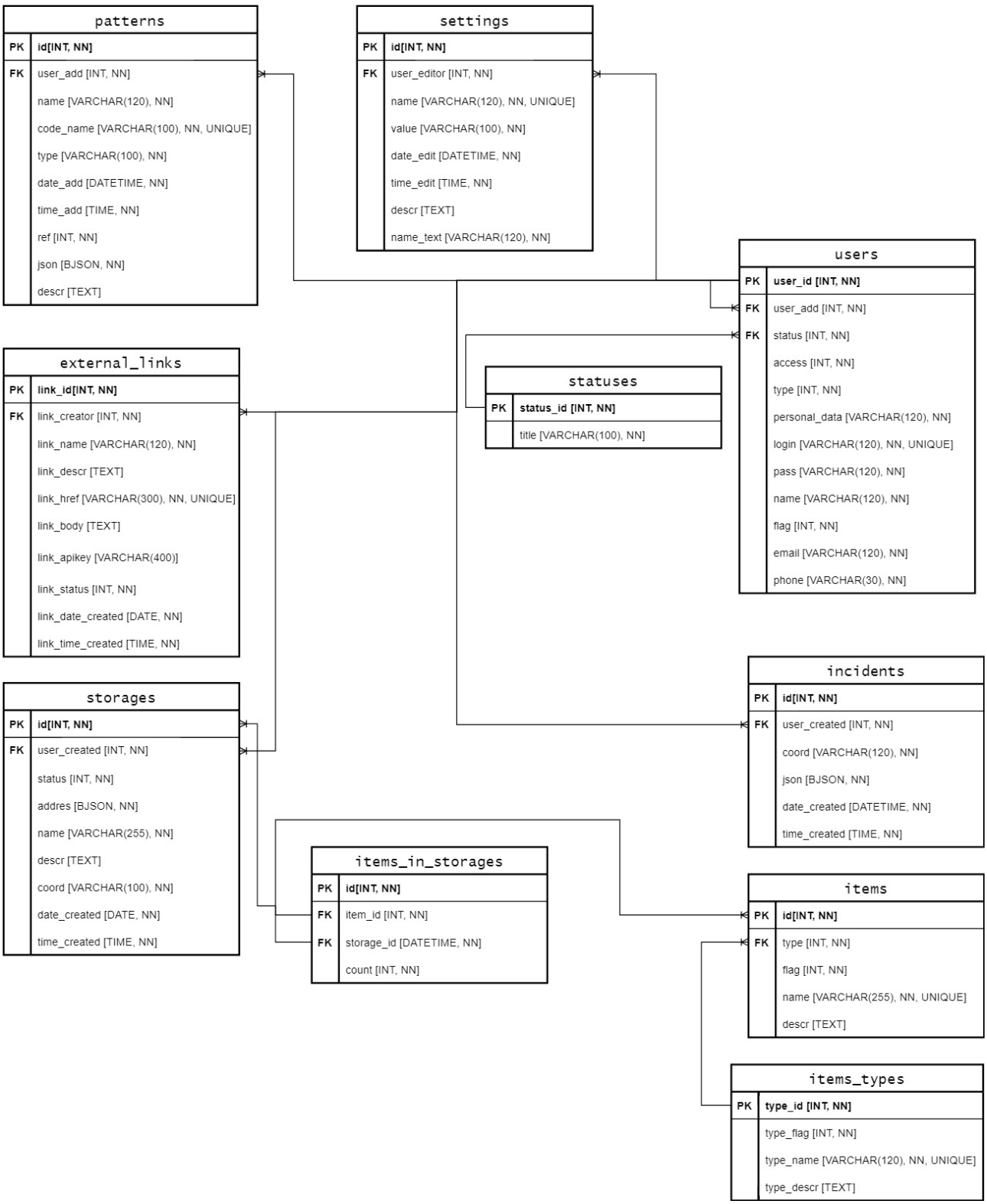

Рисунок 4 – Физическая модель БД

3.1.5 Функциональная модель

Для описания функциональной модели использовалась нотация IDEF0.

Все рисунки функциональной модели представлены в приложении Б.

На рисунке Б.1 изображён верхний уровень под названием «Рассчитать логистику».

Данные на вход:

- Данные пользователя – данные, под которыми пользователь пытается воспользоваться системой;
- Данные по происшествию – такая информация как тип, координаты, и иные данные.

Управление:

- Уровень доступа пользователя к системе – управляет возможностью пользователя использовать отдельные функции в зависимости от его уровня доступа;
- Геосистема – внешняя система, которая предоставляет возможность работать с геоданными.

Механизмы:

- База данных – место хранения данных;
- ПО – программное обеспечение, которое осуществляет функционирование системы;
- Администратор системы – глава управления в системе, регистрирует пользователей и управляет их возможностями в системе.

Выход:

- Запрет доступа – сообщение пользователю о том, что он пытается делать запрещённые действия;
- Ошибка регистрации происшествия– сообщение пользователю о том, что присланное им происшествие содержит ошибку, которая не позволяет продолжить обработку;

- Данные по реакции на происшествие – обработанные системой итоговые данные.

На рисунке Б.2 представлена декомпозиция блока «Рассчитать логистику» на 4 блока: аутентифицировать пользователя, зарегистрировать происшествие, обработать происшествие и обработать результат.

На рисунке Б.3 представлена декомпозиция блока «Аутентифицировать пользователя». Здесь происходит авторизация пользователя для работы с системой. Если у пользователя нет аккаунта, то он может создать его. Регистрирует пользователя администратор системы. После авторизации, пользователь может продолжить работать с системой или получить запрет доступа.

На рисунке Б.4 представлена декомпозиция блока «Обработать происшествие». Здесь происходит основная обработка происшествия: ищутся ближайшие склады, в них ищутся необходимые ресурсы. Затем из найденных ресурсов составляются списки для каждого склада – какие ресурсы именно и сколько они должны отправить на происшествие, после чего рассчитывается количество необходимых рейсов от склада до точки происшествия с ресурсами. Более подробно алгоритм расписан в пункте 3.3.2.2.

На рисунке Б.5 представлена декомпозиция блока «Обработать результат». На вход приходит список ресурсов и рейсов, для них мы формируем окончательный отчёт о времени, дистанции, товарах и прочих данных. Данный отчёт мы сохраняем в систему и показываем пользователю.

3.2 Выбор средств реализации

3.2.1 Выбор средств разработки клиентской части

При разработке клиентской части, помимо JavaScript, описанного в пункте 2.2 были использованы следующие технологии:

- HTML – является основным языком разметки, используемым для создания структуры и отображения контента веб-страниц. Он представляет собой комбинацию набора тегов и гипертекста;
- CSS – каскадные таблицы стилей, который используется для оформления и визуального представления веб-страниц. CSS определяет, как элементы HTML должны быть отображены на экране, включая их цвет, шрифты, размеры, расположение и другие аспекты внешнего вида;
- Bootstrap - бесплатный набор инструментов с открытым исходным кодом для разработки веб-сайтов. Он содержит HTML- и CSS-шаблоны для типографики, форм, кнопок, навигации и других интерфейсных компонентов, а также дополнительные расширения JavaScript.

В качестве фреймворка был выбран React.js (Таблица 1), поскольку он предоставляет мощные инструменты для создания интерактивных пользовательских интерфейсов. Одним из основных преимуществ React является его использование виртуального DOM, что обеспечивает быструю отрисовку компонентов и эффективное управление состоянием. Кроме того, React позволяет многократно использовать один и тот же компонент при помощи JSX (JavaScript XML), а также обладает огромным сообществом разработчиков, множеством сторонних библиотек и инструментов, а также отличной документацией, что упрощает разработку и поддержку проектов на React. Сравнение React.js происходило с другим популярным фреймворком - Vue.js [14] – это JavaScript фреймворк для создания пользовательских

интерфейсов и одностраничных приложений на основе HTML шаблонов. Он разрабатывается с открытым исходным кодом и поддерживается сообществом. Vue позиционируется как легковесный, быстрый и гибкий инструмент для создания современных веб-приложений.

Главной причиной выбора React.js, так же является удобство использования его JSX компонентов.

Таблица 1 – Сравнительная таблица фреймворков

Функция	**React**	**Vue.js**
Виртуальный DOM	да	да
Производительность	высокая	высокая
Сообщество и поддержка	да	да
Компонентный подход	да	да
Шаблоны	JSX	HTML

3.2.2 Выбор средств разработки серверной части

Для разработки серверной части приложения было решено использовать JavaScript с фреймворком NodeJS.

Выбор происходил между JavaScript, Python и PHP (Таблица 2). Они все хорошо справляются со своей задачей, но решающем критерием стало единство языка при разработке бекенда и фронтенда.

Таблица 2 – Сравнительная таблица языков программирования (бекенд)

Функция	**JavaScript**	**Python**	**PHP**
Высокая производительность	+	-	+
Поддержка обработки событий	+	-	-
Нативная поддержка JSON	+	+	-
Поддержка сторонних библиотек	+	+	+
Асинхронность	+	+	-

Node.js - кроссплатформенная среда выполнения JavaScript, построенная на движке V8 JavaScript от Google Chrome. Она позволяет выполнять JavaScript на сервере, что отличает ее от традиционного использования JavaScript только в браузере. Он также обеспечивает быстрое выполнение кода благодаря

использованию движка V8. Благодаря тому, что и фронтенд и бекенд написаны на одном языке программирования, их разработка упрощается. Также Node.js имеет огромную систему модулей и пакетов. Это означает, что можно легко использовать готовые модули для выполнения различных задач, таких как работа с базами данных, обработка HTTP-запросов, авторизация пользователей и многое другое. Большинство популярных библиотек и фреймворков также имеют версии для Node.js.

Так же были использованы следующие технологии

- Express - минималистичный и гибкий фреймворк для приложений Node.js, который облегчает разработку веб-приложений и API;
- Bcrypt - библиотека для хеширования паролей. Она использует алгоритм хеширования Blowfish для создания хеш-паролей. Bcrypt обеспечивает безопасное хранение паролей в базе данных и защищает от атак перебора;
- Jsonwebtoken - библиотека, реализующая стандарт JSON Web Token (JWT). Она предоставляет набор методов для создания, подписывания и верификации токенов аутентификации в формате JWT. Эти токены безопасны, так как jsonwebtoken предоставляет механизмы для подписывания токенов, что позволяет проверить их целостность и подлинность [15]. Это защищает от возможных изменений или подделок токенов во время их передачи или хранения. При верификации токена библиотека также проверяет его подпись, что обеспечивает доверие валидности токена;
- Pg - библиотека для работы с PostgreSQL в среде Node.js. Она предоставляет удобный интерфейс для подключения к базе данных PostgreSQL, выполнения SQL-запросов;
- Dotenv - библиотека для работы с dotenv файлами для упрощения управления системой;
- Nodemailer- библиотека для отправки электронных писем.

3.2.3 База данных

В качестве базы данных рассматривались PostreSQL, MySQL и MongoDB (Таблица 3).

- PostreSQL - мощная и расширяемая объектно-реляционная система управления базами данных (СУБД), которая поддерживает реляционный подход, но при этом и полуструктурированные данные, такие как JSON, предоставляя мощные инструменты работы с ними.
- MySQL - открытая реляционная СУБД.
- MongoDB - это документо-ориентированная СУБД (NoSQL).

Таблица 3 – Сравнительная таблица СУБД

Функция	**PostreSQL**	**MySQL**	**MongoDB**
SQL	+	+	-
Внешние ключи	+	+	-
Поддержка JSON	+	+	+
Поддержка транзакций	+	+	-
Встроенная поддержка полнотекстового поиска	+	-	+

Некоторые данные разрабатываемого прототипа приложения строго взаимосвязаны, что требует реляционного подхода, но при этом много и полуструктурированных данных. Для удобной работы одновременно с 2 подходами было решено использовать PostreSQL.

3.3 Реализация прототипа приложения

3.3.1 Разработка клиентской части прототипа приложения

Как говорилось в пункте 3.2.1, клиентская часть приложения разработана на основе ReactJS. Использование данной технологии и его механизма компонентов позволило упростить процесс разработки.

Изначально все запросы попадают в компонент index.js, где подключаются основные настройки системы, оттуда запросы попадают в компонент App.js, где происходит основная маршрутизация запросов и контроль доступа к разделам. Поле этого подгружаются компоненты либо для не авторизованного пользователя (например, страница авторизации), либо компонент для авторизованного пользователя, где далее на основе запроса подбираются необходимые компоненты системы. Такой подход помогает избежать неправомерного доступа пользователя к разделам системы, к которым у него доступа быть не должно, так же это значительно упрощает разработку приложения.

Система состоит из 10 модулей: модуль пользователя, модуль информации о складах, об инцидентах, о предметах, о паттернах, о пользователях, о внешних ссылках и с настройками системы. Основными в работе системы логистики можно назвать склады, инциденты и паттерны, посмотрим их страницы.

3.3.1.1 Модуль складов

Все рисунки модуля складов представлены в приложении В.

Модуль складов состоит из 3 частей: список складов с основной информацией о них и картой, где можно визуально перейти на любой склад (Рисунок В.1), страница создания склада, где можно создать новый склад или отредактировать существующий (Рисунок В.2) и страница с информацией о складе и информацией о товарах на данном складе (Рисунок В.3).

3.3.1.2 Модуль паттернов

Все рисунки модуля паттернов представлены в приложении Г.

Страница паттернов также состоит из 3 частей: список паттернов по названиям с возможностью фильтровать по типу (Рисунок Г.1), страница создания паттерна, где можно создать новый паттерн или отредактировать существующий (Рисунок Г.2) и страница с информацией о паттерне (Рисунок Г.3).

3.3.1.3 Модуль инцидентов

Все рисунки модуля инцидентов представлены в приложении Д.

Страница складов состоит из 2 частей: список инцидентов с основной информацией о них и картой, где можно визуально перейти на любой склад (Рисунок Д.1) и страница с полной информацией об инциденте, включая то, что было заявлено, реакцию по инцидентам и реакцию по складам (Рисунок Д.2).

3.3.2 Разработка серверной части прототипа приложения

3.3.2.1 Общий подход к разработке серверной части

Как говорилось в пункте 3.2.2, серверная часть приложения разработана на основе NodeJS.

В пункте 3.1.2 было решено, что разработка будет вестись на основе архитектурного паттерна MVC, что предполагает разделение логики приложения на контроллер, модель и представление. Представлением в системе является фронтенд часть, а значит нужно разделить логику серверной части на контроллер и модель (в качестве которой будут выступать сервисы). Благодаря этому структура бекенда выглядит следующим образом: запрос попадает в основной роутер приложения, где на основе запроса выбирается модуль, в который нужно этот запрос переправить, подроутер модуля принимает запрос и выбирает контроллер, которому стоит передать данный запрос. Так же подроутер проверяет общий доступ клиента к данному запросу (проверяет авторизацию и уровень доступа, если необходимо). Каждый контроллер в свою очередь передаёт запрос на сервисы для выполнения задач, которые возвращают ответ контроллеру, а он в свою очередь отправляет его на представление.

3.3.2.2 Алгоритм расчёта логистики

Поскольку основная цель работы – создание системы логистики, более подробно остановимся на алгоритме реакции на происшествие.

После отправки информации о происшествии, которая включает в себя координаты и информацию о происшествии, алгоритм следует следующим шагам:

1. Алгоритм проверяет поступившую информацию на наличие ошибок и если таковые есть, то возвращает об этом информацию клиенту, если ошибок нет, то сохраняет её и возвращает ответ об успешном сохранении.

2. Алгоритм находит на основе времени реакции из паттерна и константной скорости автомобиля движения из настроек радиус поиска складов по прямой, также определяет все необходимые товары, их количество, а также минимальное количество, которое можно отправить, что определяется процентом из настроек.

3. Алгоритм на основе настроек определяет границы поиска складов (минимальный и максимальный радиус). Сделано это для того, чтобы впоследствии оптимизировать работы с внешним сервисом для построения маршрутов.

4. В пределах максимального радиуса алгоритм находит все склады, где есть необходимые товары.

5. Начиная с минимального радиуса алгоритм идёт с определённым шагом, заданным в настройках, по радиусам до максимального или пока не пройдёт все найденные склады, или пока не найдёт все товары.

6. В данном радиусе алгоритм проверяет, достаточно ли товаров для завершения обхода по радиусам. Под достаточно подразумевается, что все необходимые товары находятся в пределах от минимально необходимого количества до необходимого. Так же для каждого склада строятся маршруты по дорогам и записывается их расстояние.

7. Когда обход складов окончен, алгоритм сортирует их в порядке возрастания по дистанции по дорогам и начиная с ближайшего проходится по складам.

8. Со склада он берёт товары в пределах процента взятия, заданного в настройках, но не менее 1 единицы. Сделано это для того, чтобы не опустошить склад одним инцидентом, но при этом в конечном итоге использовать все товары со склада.

9. Ели товаров недостаточно, то алгоритм идёт на следующий склад и повторяет действие.

10. Если все склады пройдены, а необходимый минимум товаров не собран, то алгоритм снова идёт по складам, начиная с пункта 8. Обход заканчивается, если все товары собраны или последний обход, начиная с пункта 8, не дал новых товаров.

11. Алгоритм просматривает все товары со складов и на основе процента допущения из настроек выбирает те, с которых отправлять товары может быть не целесообразно (если везётся товаров меньше процента допущения).

12. Алгоритм проверяет, можно ли убрать данную доставку, смотря на минимально необходимые товары, так как меньше этого числа быть не может, и если все товары с данного склада не уходят за приделы минимально необходимых товаров, то данная доставка убирается. Благодаря этому мы можем избавиться от невыгодных доставок товаров на места происшествий.

13. Алгоритм сохраняет данные и отправляет их во внешнее сервисы, сохраняя результат отправки.

3.3.3 Пример использования прототипа приложения

Посмотрим работу алгоритма, описанного в пункте 3.3.2 в действии. Предварительно, был создан паттерн обрушения дома, который принимает в числовом формате количество пострадавших и выбирает предметы по правилам: 0,5 аптечек на пострадавшего (с округлением в большую сторону)

и 15 бутылок воды на инцидент. Также было создано 3 склада со следующими товарами на них:

- Склад 52: 5 бутылок воды, 5 аптечек;
- Склад 53: 12 бутылок воды, 10 аптечек;
- Склад 54: 0 бутылок воды, 45 аптечек.

Настройки системы следующие:

- Процент взятия товаров с 1 склада за проход: 90%;
- Допущение: 10%;
- Скорость движения транспорта: 35 км/ч;

Создадим инцидент, указав 16 пострадавших (Рисунок 5).

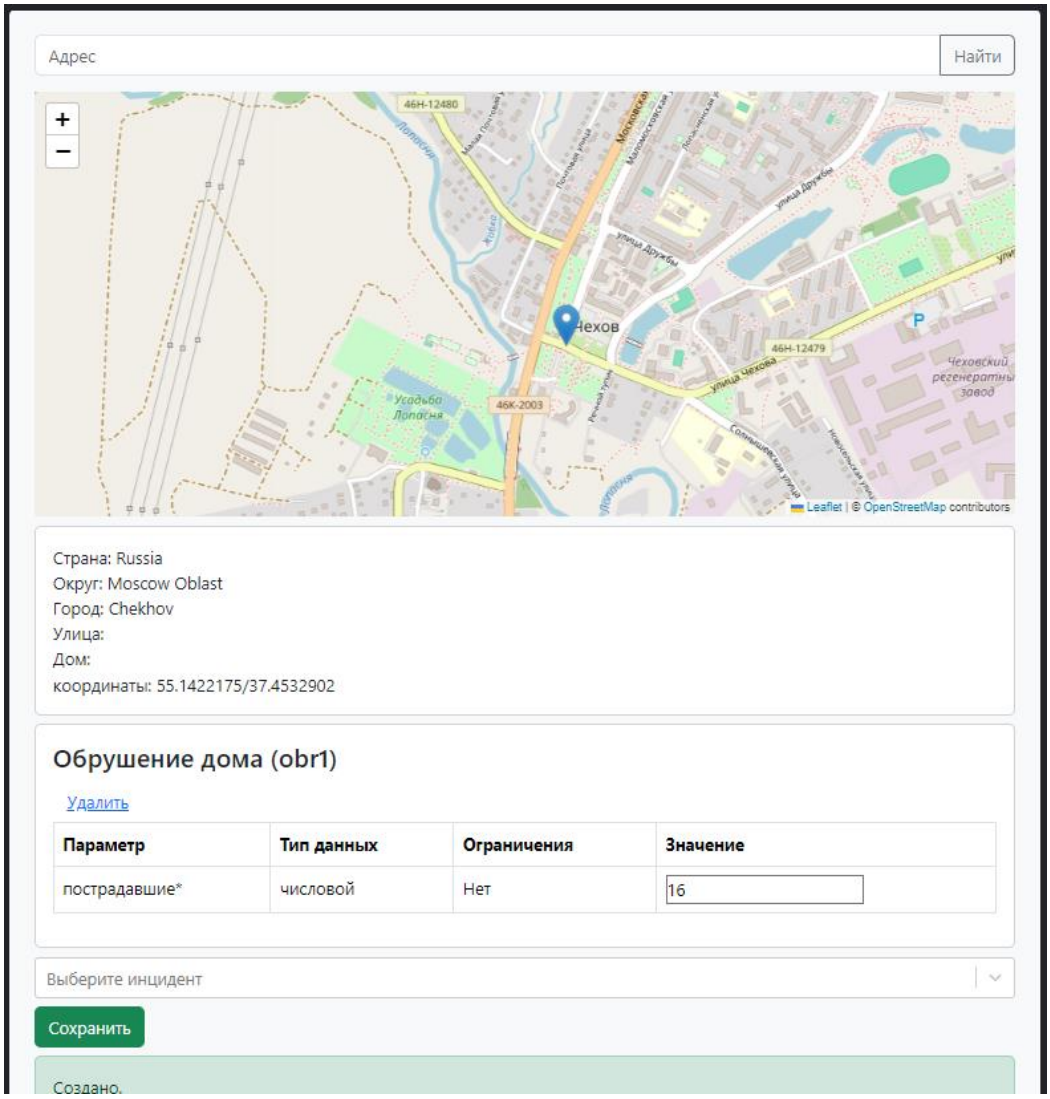

Рисунок 5 – Создание происшествия

Рассмотрим ответ системы (Рисунки 6 - 8). Видно, что согласно паттерну, она запросила 8 аптечек и 15 воды (Рисунок 7). В реакции на инцидент (Рисунок 8) видно, что система рассчитала минимальный необходимый минимум (8 аптечек и 14 воды, согласно проценту допущения) и взяла его со складов 52 (он рассматривался первым, так как ближе) и 53 (он рассматривался вторым). Так как минимум был достигнут на данных складах, система не рассматривает склад 54 для доставки. Так как за проход берётся не более 90% товаров со склада, то с 52 склада алгоритм взял 4 аптечки вместо всех 5 на складе. Остальные 4 он взял уже на 53 складе.

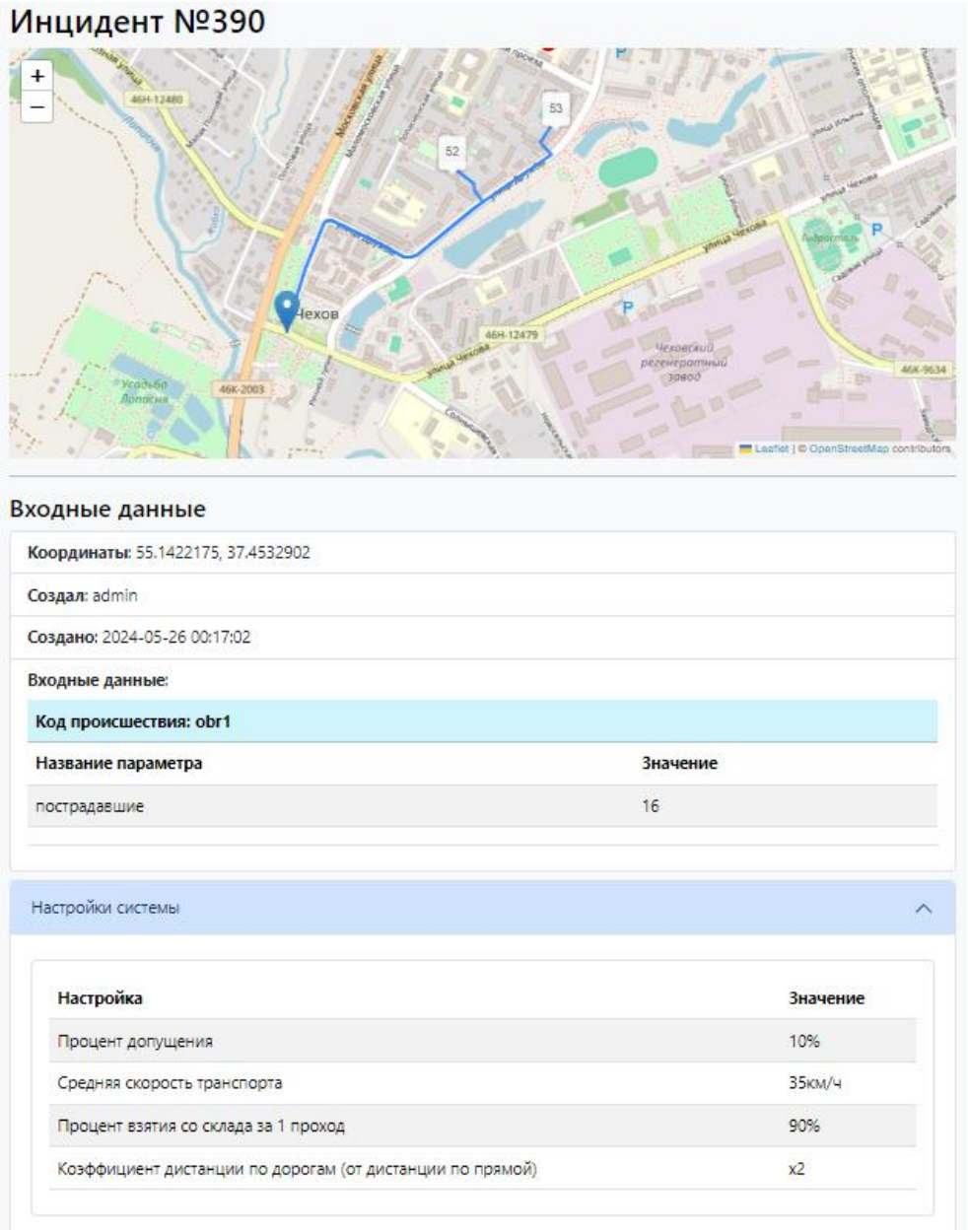

Рисунок 6 – Страница происшествия (входные данные и настройка системы)

Запрошенные предметы

Название предмета	Количество
Аптечка	8
Вода в бутылках	15

Рисунок 7 – Страница происшествия (запрошенные предметы)

Реакция по инцидентам

Код инцидента: obr1

Примерное время реакции: 20 - 40 мин

Радиус поиска: 11667м

Радиус поиска максимальный: 17501м

Радиус поиска минимальный: 2334м

Взятые предметы:

Название предмета	Количество
Аптечка	8
Вода в бутылках	14

Запрошенные предметы:

Название предмета	Количество	Минимальное количество
Аптечка	8	8
Вода в бутылках	15	14

Полученные предметы по складам:

Склад №52

Название предмета	Количество	Процент от общего числа данного предмета
Аптечка	4	50%
Вода в бутылках	4	27%

Склад №53

Название предмета	Количество	Процент от общего числа данного предмета
Аптечка	4	50%
Вода в бутылках	10	67%

Рисунок 8 – Страница происшествия (реакция по инцидентам)

Также мы можем увидеть разбиение по взятым со складов товарам и результаты отправки во внешние системы (Рисунок 9).

Реакция по складам

№ склада: №52

Адрес: Страна Russia, г. Chekhov, улица улица Дружбы, д. 14 к1

Координаты: 55.1459689, 37.4602447611949

Дистанция по дороге от склада до происшествия: 922.413м

Данные:

Название параметра	Значение
Аптечка	4
Вода в бутылках	4

№ склада: №53

Адрес: Страна Russia, г. Chekhov, улица улица Дружбы, д. 6

Координаты: 55.147031150000004, 37.464590878562504

Дистанция по дороге от склада до происшествия: 1125.949м

Данные:

Название параметра	Значение
Аптечка	4
Вода в бутылках	10

Отправка данных

URL	Код ответа	Сообщение
kjhygtfr	0	No response received from server
jkhgbfvds	0	No response received from server
http://localhost:4000/	200	Hello World!
http://localhost:4000/api/externalLink/testResp	401	{"type":"UNAUTHORIZED","message":"Не авторизован 1"}

Рисунок 9 – Страница происшествия (реакция по складам и результат отправки данных во внешние системы)

3.4 Описание процесса взаимодействия с прототипом

Для описания логики взаимодействия использовалась нотация BPMN. Модель процесса «Рассчитать логистику» отображена на рисунке В.1 в приложении Е.

Выводы по технологическому разделу

В данном разделе была спроектирована архитектура приложения на основе выбранных паттерна MVC. Была построена ER и IDEF0 диаграммы. Был создан рабочий прототип приложения для логистики в условиях ЧС. Также было выполнено описание взаимодействия с прототипом приложения с помощью нотаций BPMN.

4 ЭКОНОМИЧЕСКИЙ РАЗДЕЛ

4.1 Организация и планирование работ

В составе работы задействовано 3 человека:

Руководитель ВКР (Стариков Павел Павлович, заведующий кафедрой) – отвечает за грамотную постановку задачи, контролирует отдельные этапы работы, вносит необходимые коррективы и оценивает выполненную работу в целом.

Консультант по экономической части (Черненькая Ирина Геннадьевна, кандидат экономических наук, доцент) – отвечает за консультирование экономической части выпускной квалификационной работы.

Разработчик системы (студент 4-го курса Сорокин Андрей Александрович, ИКБО-02-20) – анализирует предметную область, проектирует и разрабатывает программное решение в соответствии с поставленной задачей.

На проектирование и разработку приложения отводится 70 дней. Этапы разработки [16] представлены в таблице 4.

Таблица 4 – Календарный план выполнения проекта

Этап	Дата начала	Дата окончания	Количество рабочих дней	Исполнители
1 Формирование требований к проекту				
1.1 Характеристика предметной области и обоснование необходимости создания системы	12.02.2024	13.02.2024	2	Разработчик
1.2 Формирование требований к приложению	14.02.2024	16.02.2024	3	Руководитель, Разработчик

Продолжение таблицы 4

1.3 Разработка и утверждение технического задания	19.02.2024	22.02.2024	4	Руководитель, Разработчик
2. Разработка концепции проекта				
2.1 Анализ предметной области	26.02.2024	28.02.2024	3	Разработчик
2.2 Анализ аналогов	29.02.2024	01.03.2024	2	Разработчик
2.3 Анализ и выбор технологий для реализации приложения	04.03.2024	06.03.2024	3	Разработчик
3 Проектный раздел				
3.1 Проектирование архитектуры веб-приложения	07.03.2024	15.03.2024	6	Разработчик
3.2 Проектирование функциональной схемы веб-приложения	18.03.2024	20.03.2024	3	Разработчик
3.3 Проектирование базы данных	21.03.2024	26.03.2024	4	Разработчик
3.4 Практическая разработка архитектуры компонентов приложения	27.03.2024	01.04.2024	4	Руководитель, Разработчик
4 Технологический раздел				
4.1 Разработка клиентской части приложения	02.04.2024	23.04.2024	16	Разработчик
4.2 Разработка серверной части приложения	24.04.2024	15.05.2024	12	Разработчик
4.3 Доработка программного кода	16.05.2024	23.05.2024	6	Разработчик, Руководитель
5 Экономический раздел				
5.1 Организация и планирование работ по теме	24.05.2024	24.05.2024	1	Консультант, Разработчик

Окончание таблицы 4

5.2 Расчет стоимости проведения работ по теме	27.05.2024	27.05.2024	1	Консультант, Разработчик

Для отображения календарного плана была выбрана диаграмма Ганта [17, 18]. График, разработанный в соответствии с данными из таблицы 4 [19], представлен в приложении Ж (Рисунок Ж.1).

4.2 Расчет затрат на проведение работ

Себестоимость проектирования и разработки приложения складывается из затрат по следующим статьям:

- основная заработная плата;
- дополнительная заработная плата – 20-30% от основной заработной платы;
- страховые взносы – 30% от фонда оплаты труда (ФОТ), а также 0,2% ставка за травматизм;
- материалы, покупные изделия и полуфабрикаты, а также до 20% от соответствующей суммы на транспортно-заготовительные расходы (ТЗР);
- расходы на приобретение компьютерной техники и программного обеспечения, а также до 20% от соответствующей суммы на ТЗР;
- амортизационные отчисления по используемому в проекте оборудованию;
- эксплуатационные расходы, связанные с использованием компьютерной техники;
- накладные расходы – до 150% от основной заработной платы;
- прочие расходы.

4.2.1 Статья «Основная заработная плата»

Таблица 5 – Результаты анализа размера оплаты труда

Исполнитель (должность)	Ряд значений	Источник данных	Сред. арифм. (μ)	Медиана (Md)	Квартили
Руководитель ВКР	126 000	https://hh.ru/vacancy/94930360	144 250	144 500	127 250
	110 000	https://hh.ru/vacancy/99599026			144 500
	178 000	https://hh.ru/vacancy/98545718			161 250
	163 000	https://hh.ru/vacancy/98302108			
Консультант по экономической части	100 000	https://hh.ru/vacancy/98713309	83 363	85 225	74 113
	63 000	https://hh.ru/vacancy/99597329			85 225
	79 000	https://hh.ru/vacancy/98247761			92 613
	91 450	https://hh.ru/vacancy/99600816			
Разработчик проекта	120 000	https://hh.ru/vacancy/91721929	132 500	130 000	125 000
	150 000	https://hh.ru/vacancy/98657947			130 000
	140 000	https://hh.ru/vacancy/89847268			140 000
	120 000	https://hh.ru/vacancy/98927600			

Среднее и значение медианы примерно равны, а следовательно ряды уровней оплаты труда являются репрезентативными.

Расчет дневной ставки и оплата за этапы для каждого из участников проекта представлена в таблице 6.

Таблица 6 – Расчет основной заработанной платы

Наименование этапа	Исполнитель (должность)	Мес. оклад (руб.)	Оплата за день (руб.)	Количество рабочих дней	Оплата за этап (руб.)
1 Формирование требований к проекту	Руководитель	144 500	5 780	7	40 460
	Консультант	85 225	3 409	0	0
	Разработчик	130 000	5 200	9	46 800
2. Разработка концепции проекта	Руководитель	144 500	5 780	0	0
	Консультант	85 225	3 409	0	0
	Разработчик	130 000	5 200	8	41 600
3 Проектный раздел	Руководитель	144 500	5 780	4	23 120
	Консультант	85 225	3 409	0	0
	Разработчик	130 000	5 200	17	88 400
4 Технологический раздел	Руководитель	144 500	5 780	6	34 680
	Консультант	85 225	3 409	0	0
	Разработчик	130 000	5 200	34	176 800
5 Экономический раздел	Руководитель	144 500	5 780	0	0
	Консультант	85 225	3 409	2	6 818

Таблица 6 – Окончание таблицы

	Разработчик	130 000	5 200	2	10 400
				ИТОГО:	469 078

Таким образом, заработная плата исполнителей составит 469 078 рублей.

4.2.2 Статья «Дополнительная заработная плата»

Расходы по данной статье составляют 20-30% от суммы основной заработной платы и рассчитываются по формуле (1):

$$ДЗП = (20 \dots 30)\% \times ОЗП \quad (1)$$

где ОЗП – основная заработная плата.

В таком случае, ДЗП будет равен:

$$ДЗП = 0.2 \times 469\,078 = 93\,816 \text{ руб.}$$

Таким образом, расходы на дополнительную заработную плату составляют 93 816 руб.

4.2.3 Статья «Страховые взносы»

Фонд оплаты труда (ФОТ) считается по формуле (2):

$$ФОТ = ОЗП + ДЗП \quad (2)$$

где ОЗП – основная заработная плата;

ДЗП – дополнительная заработная плата.

Страховые взносы составляют 30% по всем направлениям и 0.2% ставка взносов на травматизм от ФОТ и считаются по формуле (3):

$$СВ = 30.2\% \times ФОТ \quad (3)$$

где ФОТ – фонд оплаты труда.

Таким образом, страховые взносы равны:

$$СВ = 0.302 \times (469\,078 + 93\,816) = 169\,994 \text{ руб.}$$

Итого, затраты по статье «Страховые взносы» составляют 169 994 руб.

4.2.4 Статья «Материалы, покупные изделия и полуфабрикаты»

Расчет стоимости материалов, покупных изделий и других материальных ценностей представлен в таблице 7.

Таблица 7 – Стоимость материалов

№ п/п	Наименование материалов	Единицы измерения	Количество	Цена за единицу (руб.)	Стоимость (руб.)
1	Флешка 8Гб	шт.	2	461	922
2	Бумага А4	пачка	1	492	492
3	Картридж для принтера	шт.	1	1 136	1 136
4	Ручка	шт.	2	29	58
Итого материалов					2 608
Транспортно-заготовительные расходы					300
Итого:					2 908

Таким образом, общая сумма затрат по статье «Материалы, покупные изделия и полуфабрикаты» составит 2 908 руб.

4.2.5 Статья «Расходы на приобретение компьютерной техники и программного обеспечения»

В качестве программного обеспечения необходимо облачное хранилище и виртуальные вычислительные мощности. Сравнительный анализ разных хранилищ и виртуальных вычислительных мощностей приведен в таблице 8.

Таблица 8 – Обзор возможных к использованию облачных хранилищ

	Критерии для выбора оптимального варианта				
Облачное хранилище	**Техниче-ские характе-ристики**	**Дополни-тельные возможности**	**Програм мное обеспече ние**	**Ограничения**	**Затраты**

Таблица 8 – Окончание таблицы

hostiman.ru	Минимум 3,7ГГц 1 ядро + 1 Гб RAM + NVMe 20Гб	Безлимитный трафик.	Ubuntu	Отсутствие дополнительных услуг.	От 500 руб/мес.
Reg.ru	Минимум 2.2ГГц + SSD 13Гб	Удобная веб-консоль прямо в браузере, бесплатная зашита от DDoS, бесплатное хранение резервных копий.	CentOS	Максимальный размер одной базы данных – 4Гб	От 390 руб./мес.
Clo.ru	Минимум 3,2ГГц 1 ядро + 2 Гб RAM + NVMe 10Гб	Точная настройка ресурсов облачного сервер.	Ubuntu	Защита от DDoS	От 1 450 руб./мес.

В итоге проведённого анализа был выбран hostiman.ru. Расходы на программное обеспечение приведены в таблице 9.

Таблица 9 – Стоимость программного обеспечения

Наименование	**Количество**	**Цена за единицу (руб.)**	**Стоимость (руб.)**
hostiman.ru	3	500	1500
		Итого:	1500

Таким образом, сумма затрат на программное обеспечение составит 1500 руб.

4.2.6 Статья «Амортизационные отчисления»

Трат по данной статье не предусмотрено, так как новое оборудование не приобретается.

4.2.7 Статья «Эксплуатационные расходы, связанные с использованием компьютерной техники»

Стоимость 1 кВт составляет 6,43 руб. [20]. Расчет эксплуатационных расходов представлен в таблице 10.

Таблица 10 – Эксплуатационные расходы

Наименование техники	**Мощность, кВт**	**Время использования, ч.**	**Стоимость 1 кВт, руб.**	**Расходы, руб.**
Компьютер	0.25	550	6.43	885.12
Принтер	0.2	2		2.57
			Итого:	887.69

Итого, амортизационные отчисления составляют 887.69 руб.

4.2.8 Статья «Накладные расходы»

Сумма данных расходов определяется процентом от суммы основной заработной платы (ОЗП). Вычисляется по формуле (4):

$$\text{НР} = (100 \dots 130)\% \times \text{ОЗП} \quad (4)$$

где ОЗП – основная заработная плата.

Итого, по подсчетам:

$$\text{НР} = 100\% \times 469\,078 = 469\,078 \text{ руб}$$

Затраты по статье «Накладные расходы» составляют 469 078 руб.

4.2.9 Статья «Прочие расходы»

Трат по данной статье не предусмотрено.

4.2.10 Полная себестоимость проекта

Расчет полной себестоимости проекта приведен в таблице 11.

Таблица 11 – Общий объем финансирования проекта

Номенклатура статей расходов	**Расходы (руб.)**	**Доля расходов, %**
1 Основная заработная плата	469078.00	38,85
2 Дополнительная заработная плата	93816.00	7,77
3 Страховые взносы	169994.00	14,08
4 Материалы, покупные изделия и полуфабрикаты	2908.00	0,24
5 Расходы на приобретение компьютерной техники и программного обеспечения	1500.00	0,12
6 Амортизационные отчисления	0.00	0
7 Эксплуатационные расходы, связанные с использованием компьютерной техники	887.69	0,07
8 Накладные расходы	469078.00	38,85
9 Прочие	0.00	0
Итого:	1207261,69	100,00

Для визуализации долевого состава статей затрат в общей себестоимости проекта была создана круговая диаграмма, представленная в приложении З (Рисунок З.1).

Выводы по экономическому разделу

В ходе проделанной работы по экономическому разделу были выполнены следующие задачи:

- определен список участников проекта и произведено распределение количества времени выполнения для каждого участника проекта;
- составлен календарный план выполнения проекта на диаграмме Ганта;
- произведены расчеты затрат на проведение работ по статьям.

Результатом работы по экономической практике является календарный план-график и полная себестоимость проекта.

ЗАКЛЮЧЕНИЕ

В ходе выполнения выпускной квалификационной работы была рассмотрена система расчёта логистики в условиях чрезвычайных ситуаций, а также спроектирован и разработан прототип приложения. Для описания взаимодействия использовались методологии BPMN и IDEF0. В результате анализа существующих информационных систем было выявлено, что аналоги не обладают функционалом, необходимым для управления логистикой в условиях чрезвычайных ситуаций, что ограничивает их практическую ценность в таких условиях. Разрабатываемое приложение предоставляет новый функционал, способный дополнить и улучшить существующие решения.

Для реализации проекта были выбраны следующие средства разработки: Node.js для серверной части, база данных PostgreSQL, а такжеReactJS, HTML, CSS для фронтенд-разработки. В ходе описания реализации базы данных была построена физическая модель базы данных.

В работе проведено технико-экономическое обоснование проекта. Длительность разработки составляет 4 месяца, включающие 70 рабочих дней, при этом общая стоимость проекта составила 1 207 261.69 рубль.

В ходе работы были успешно реализованы все поставленные задачи, включая формирование характеристики предметной области, анализ существующих решений, выбор и обоснование технологий для реализации алгоритма и веб-приложения, формирование требований к прототипу приложения, организация и планирование работ, расчет стоимости проведения работ, проектирование архитектуры веб-приложения, разработка клиентской и серверной частей веб-приложения, и описание процесса взаимодействия с прототипом.

Таким образом, результаты данной работы могут быть использованы для дальнейшего улучшения и развития сервиса, а также для проведения

дополнительных исследований в данной области. Это может включать добавление новых функций, улучшение пользовательского интерфейса, интеграцию с другими системами и сервисами, а также проведение дополнительных тестов и экспериментов для оптимизации работы сервиса. В целом, данная работа является важным шагом на пути к созданию эффективных и удобных инструментов для расчёта логистики в условиях чрезвычайных ситуаций.

Ссылка на полный исходный код компонентов прототипа приложения: https://github.com/AndreyErr/EmergencyLogiApp

During the completion of the final qualification work, the logistics calculation system in emergency situations was reviewed, and a prototype application was designed and developed. BPMN and IDEF0 methodologies were used to describe the interaction. As a result of the analysis of existing information systems, it was revealed that analogues do not have the functionality necessary for logistics management in emergency situations, which limits their practical value in such conditions. The application under development provides new functionality that can complement and improve existing solutions.

The following development tools were selected for the project: Node.js for the backend, PostgreSQL database, as well as REACTJS, HTML, CSS for frontend development. During the description of the database implementation, a physical database model was built.

A feasibility study of the project was carried out in the work. The duration of the development is 4 months, including 70 working days, while the total cost of the project was 1,207,261.69 rubles.

In the course of the work, all the tasks were successfully implemented, including the formation of the characteristics of the subject area, analysis of existing solutions, selection and justification of technologies for the implementation of the algorithm and web application, formation of requirements for the prototype of the application, organization and planning of work, calculation of the cost of work,

design of the architecture of the web application, development of client and server parts of the web-applications, and a description of the interaction process with the prototype.

Thus, the results of this work can be used to further improve and develop the service, as well as to conduct additional research in this area. This may include adding new features, improving the user interface, integrating with other systems and services, as well as conducting additional tests and experiments to optimize the service. In general, this work is an important step towards creating effective and convenient tools for calculating logistics in emergency situations.

Link to the full source code of the application prototype components: https://github.com/AndreyErr/EmergencyLogiApp

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. pubmed [Электронный ресурс] / Emergency Logistics in a Large-Scale Disaster Context: Achievements and Challenges – Режим доступа: https://www.mdpi.com/1660-4601/16/5/779 (Дата обращения: 27.05.2024)

2. mercurygate [Электронный ресурс] / about company – Режим доступа: https://mercurygate.com/company/ (Дата обращения: 27.05.2024)

3. relog [Электронный ресурс] / О решении – Режим доступа: https://getrelog.com/ (Дата обращения: 27.05.2024)

4. 1С:Предприятие 8. 1С-Логистика:Управление складом 3.0 [Электронный ресурс] / О решении – Режим доступа: https://solutions.1c.ru/catalog/wms/features (Дата обращения: 27.05.2024)

5. OpenStreetMap [Электронный ресурс] / OpenStreetMap – Режим доступа: https://www.openstreetmap.org (Дата обращения: 27.05.2024)

6. Яндекс.Карты [Электронный ресурс] / Условия использования API Яндекс.Карт– Режим доступа: https://yandex.ru/dev/maps/jsapi/doc/2.1/terms/ (Дата обращения: 27.05.2024)

7. 2gis [Электронный ресурс] / Правовая информация по API 2ГИС – Режим доступа: https://law.2gis.ru/api-rules (Дата обращения: 27.05.2024)

8. GraphHopper [Электронный ресурс] / GraphHopper – Режим доступа: https://www.graphhopper.com/ (Дата обращения: 27.05.2024)

9. Яндекс Справка [Электронный ресурс] / О решении – Режим доступа: https://yandex.ru/support/browser/about/install.html (Дата обращения: 27.05.2024)

10. systemreq [Электронный ресурс] / Node.js – Режим доступа:

https://systemreq.ru/node-js/ (Дата обращения: 27.05.2024)

11. hexlet [Электронный ресурс] / Что такое MVC: рассказываем простыми словами – Режим доступа: https://ru.hexlet.io/blog/posts/chto-takoe-mvc-rasskazyvaem-prostymi-slovami (Дата обращения: 27.05.2024)

12. ibm [Электронный ресурс] / Клиент и сервер – Режим доступа: https://www.ibm.com/docs/ru/aix/7.1?topic=systems-client-server (Дата обращения: 27.05.2024)

13. ibm [Электронный ресурс] / Что такое трехуровневая архитектура– Режим доступа: https://www.ibm.com/topics/three-tier-architecture (Дата обращения: 27.05.2024)

14. timeweb [Электронный ресурс] / Vue.js– Режим доступа: https://timeweb.com/ru/community/articles/obzor-vue-js-1 (Дата обращения: 27.05.2024)

15. proglib [Электронный ресурс] / JWT простым языком– Режим доступа: https://proglib.io/p/json-tokens (Дата обращения: 27.05.2024)

16. ГОСТ Р 59793-2021 [Электронный ресурс] / Информационные технологии. Комплекс стандартов на автоматизированные системы. Автоматизированные системы. Стадии создания. – Режим доступа: https://www.consultant.ru/cons/cgi/online.cgi?req=doc&base=STR&n=28446&ysclid=lh58d1eqwn824473254#PYvE2dTmaWIAahGI (Дата обращения 27.05.2024)

17. 1С:Предприятие 8 [Электронный ресурс] / Диаграмма Ганта – Режим доступа: https://v8.1c.ru/platforma/diagramma-ganta/ (Дата

обращения 27.05.2024)

18. GanttPRO [Электронный ресурс] / Диаграмма Ганта для управления проектами - Режим доступа: https://app.ganttpro.com/ (Дата обращения 27.05.2024)

19. calendar555.ru [Электронный ресурс] / Календарь на 2024 год с праздниками и выходными – Режим доступа: https://calendar555.ru/ (Дата обращения 27.05.2024)

20. energoseti.ru [Электронный ресурс] / Тарифы на электроэнергию в Москве – Режим доступа: https://energoseti.ru/rate/moskva (Дата обращения 27.05.2024)

ПРИЛОЖЕНИЕ А

Таблица А.1 – пользовательские истории

Как (роль)	Я хочу	Для того чтобы	Постановка задачи
Старший администратор, Администратор, Старший пользователь, Пользователь	Иметь возможность авторизоваться	Иметь персональный доступ к возможностям системы	Разработать модуль авторизации
Старший администратор, Администратор, Старший пользователь	Иметь возможность зарегистрировать пользователя	Создавать учётные записи других пользователей	Разработать модуль управления пользователями
Старший администратор, Администратор, Старший пользователь	Иметь возможность ограничить функционал пользователя	Ограничивать доступный на аккаунте функционал	Разработать функционал для возможности ограничить функционал системы у конкретных пользователей
Старший администратор, Администратор, Старший пользователь, Пользователь	Иметь возможность редактировать пользователя	Обновлять данные о пользователях на актуальные	Разработать модуль управления пользователями
Старший администратор, Администратор, Старший пользователь	Иметь возможность создавать и редактировать склады	Добавлять или изменять информацию о складах	Разработать модуль управления складами

Таблица А.1 – Продолжение таблицы

Старший администратор, Администратор, Старший пользователь	Иметь возможность создавать и редактировать предметы	Добавлять или изменять информацию о предметах	Разработать модуль управления предметами
Старший администратор, Администратор	Иметь возможность создавать и редактировать категории предметов	Добавлять или изменять информацию о категориях предметов	Разработать функционал управления категориями предметов
Старший администратор, Администратор	Иметь возможность создавать и редактировать паттерны происшествий	Добавлять или изменять информацию о паттернах происшествий	Разработать модуль управления паттернами
Старший администратор, Администратор, Старший пользователь, Пользователь	Иметь возможность создавать происшествия	Добавлять информацию о происшествиях	Разработать модуль управления происшествиями
Старший администратор, Администратор, Старший пользователь, Пользователь	Иметь возможность просмотра информации о происшествии	Просматривать информацию о происшествии	Разработать функционал для удобного просмотра информации о происшествии
Старший администратор, Администратор, Старший пользователь, Пользователь	Иметь возможность отправлять данные о происшествии во внешние системы	Отправлять информацию о происшествии для использования в иные системы	Разработать функционал для отправки информации во внешнюю систему

Таблица А.1 – Окончание таблицы

Старший администратор, Администратор, Старший пользователь	Иметь возможность сменить пароль учётной записи через почту	Восстановить доступ к своему аккаунту или аккаунту пользователя в случае его утери	Разработать функционал для восстановления пароля
Старший администратор, Администратор, Старший пользователь	Иметь возможность заблокировать или разблокировать доступ пользователя к системе	Ограничить полный доступ к системе	Разработать функционал для управления доступом к системе
Старший администратор	Иметь возможность просмотреть и изменить настройки системы	Изменить поведение системы	Разработать функционал для просмотра и изменения настроек системы

ПРИЛОЖЕНИЕ Б

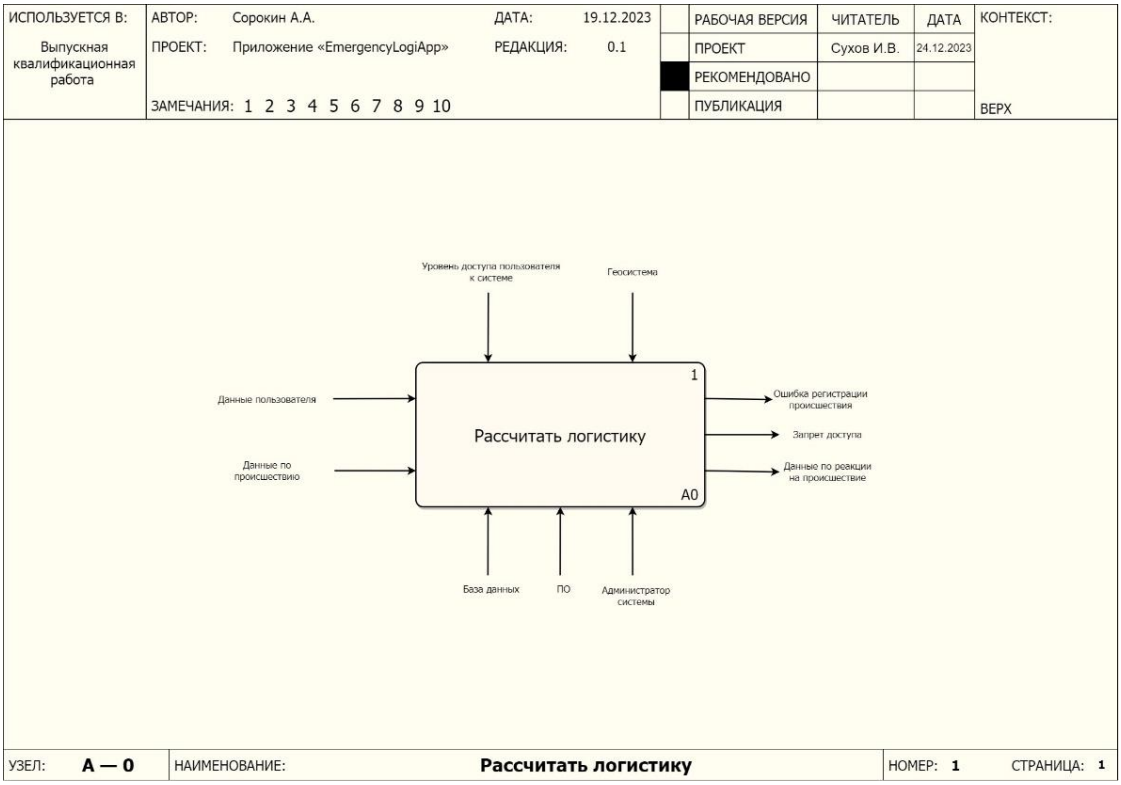

Рисунок Б.1 – Контекстная диаграмма процесса «Рассчитать логистику» (верхний уровень)

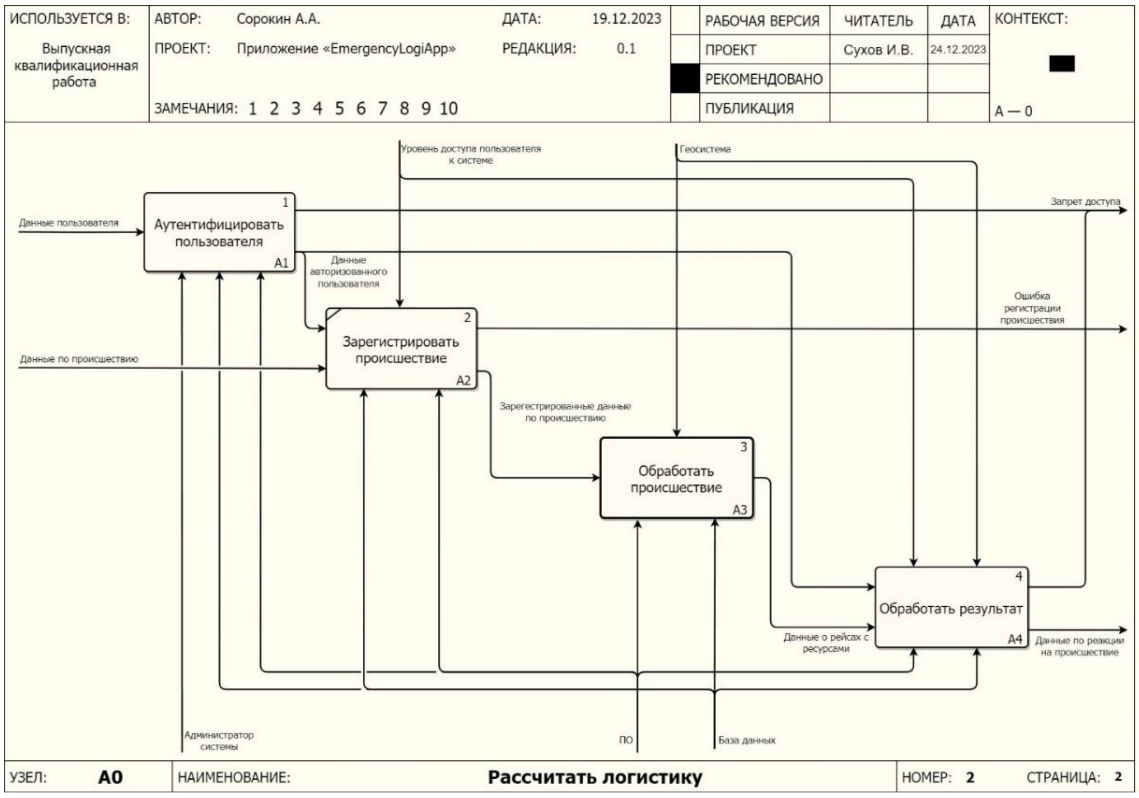

Рисунок Б.2 – Контекстная диаграмма процесса «Рассчитать логистику»

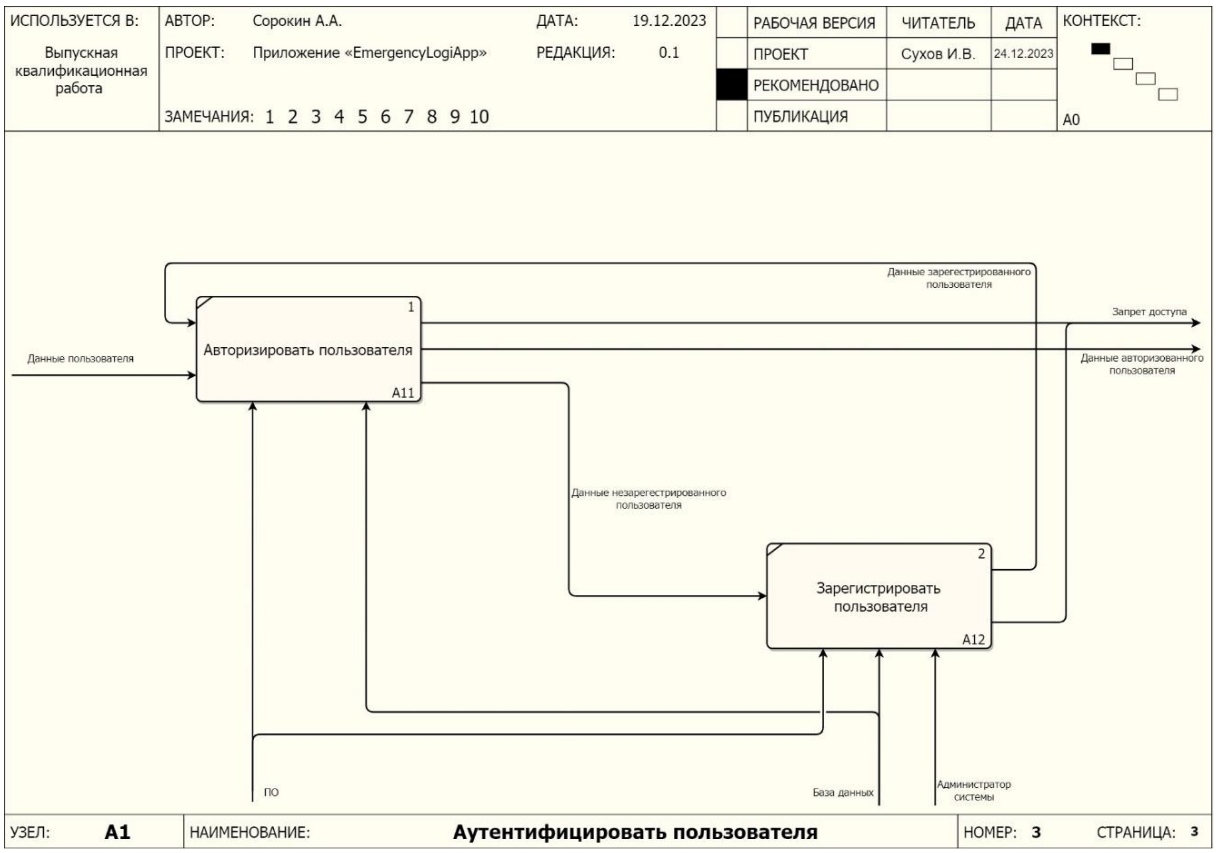

Рисунок Б.3 – Контекстная диаграмма процесса «Аутентифицировать пользователя»

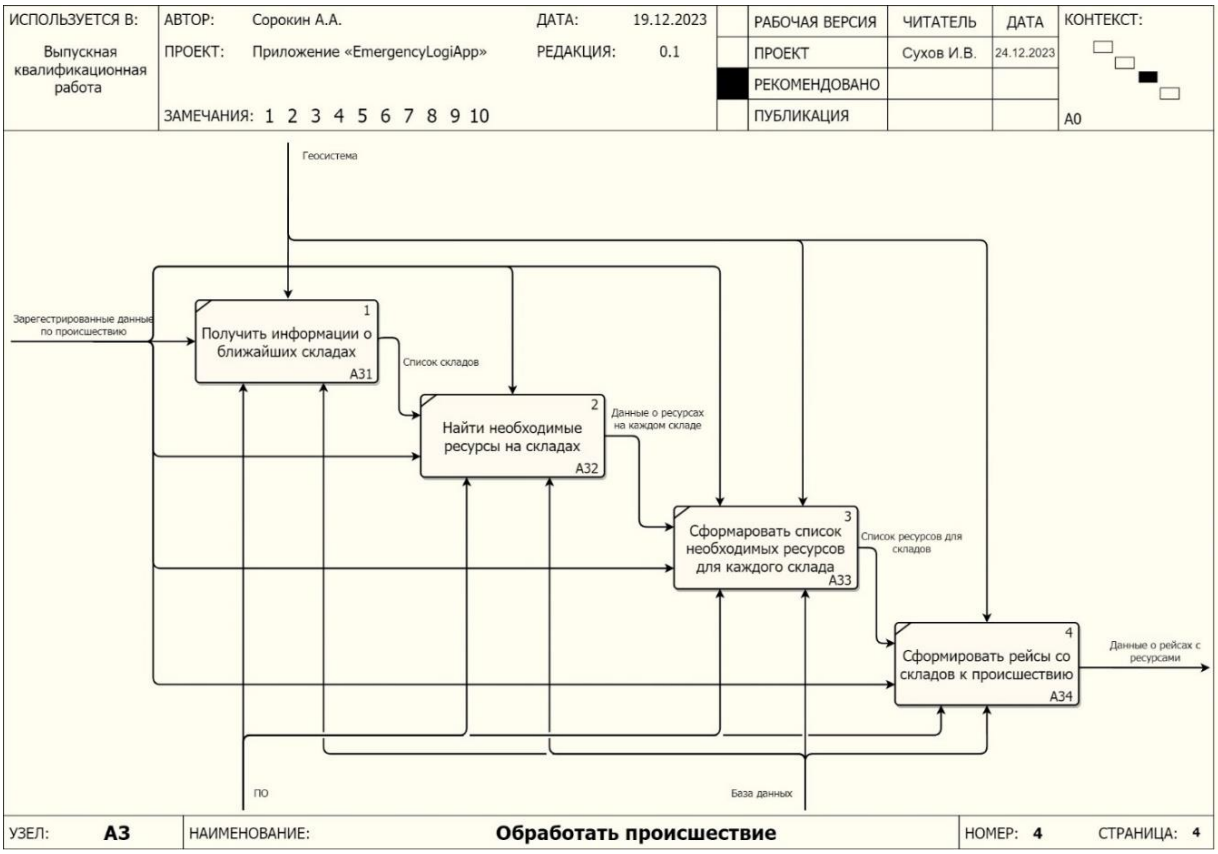

Рисунок Б.4 – Контекстная диаграмма процесса «Обработать происшествие»

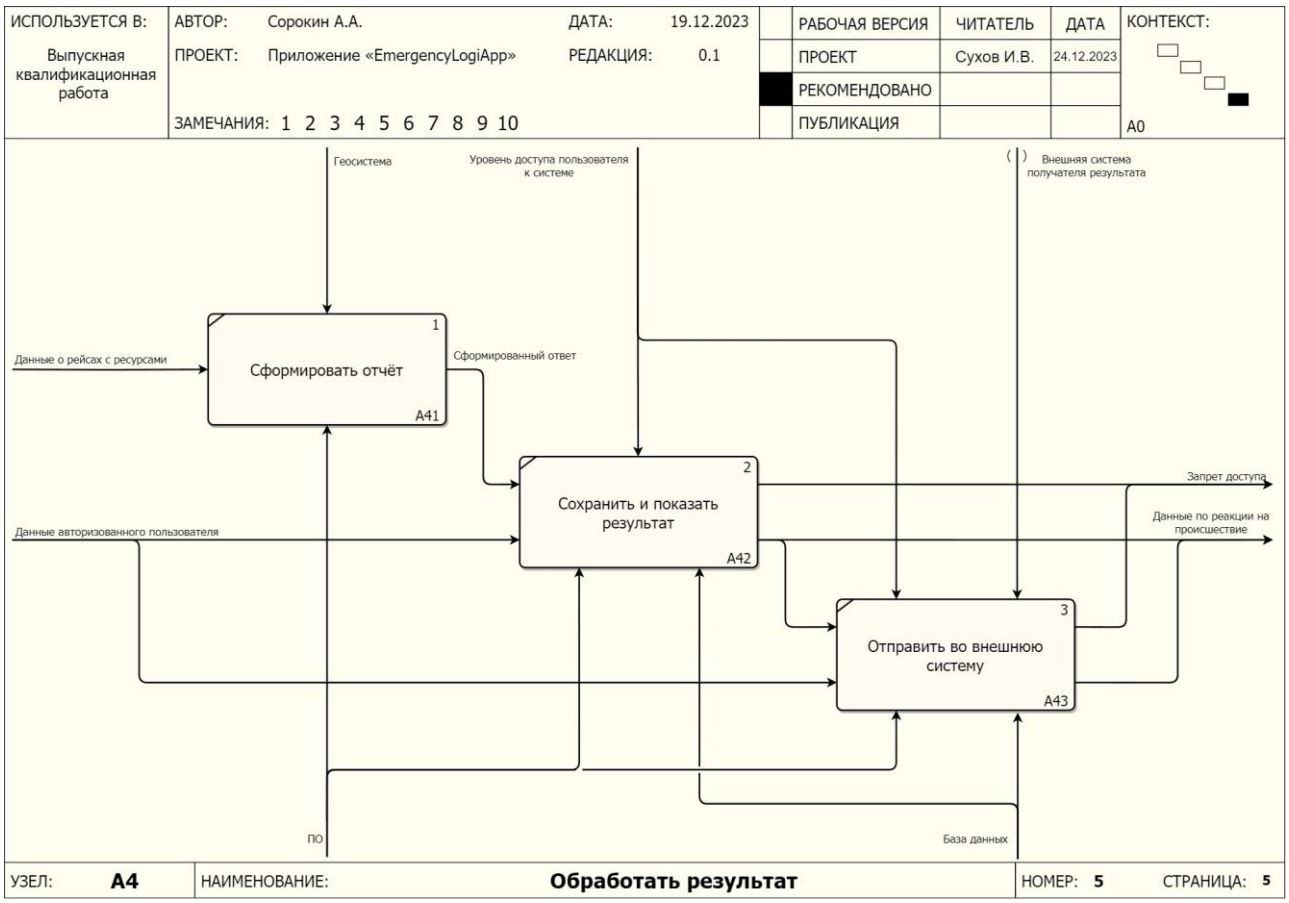

Рисунок Б.5 – Контекстная диаграмма процесса «Обработать результат»

ПРИЛОЖЕНИЕ В

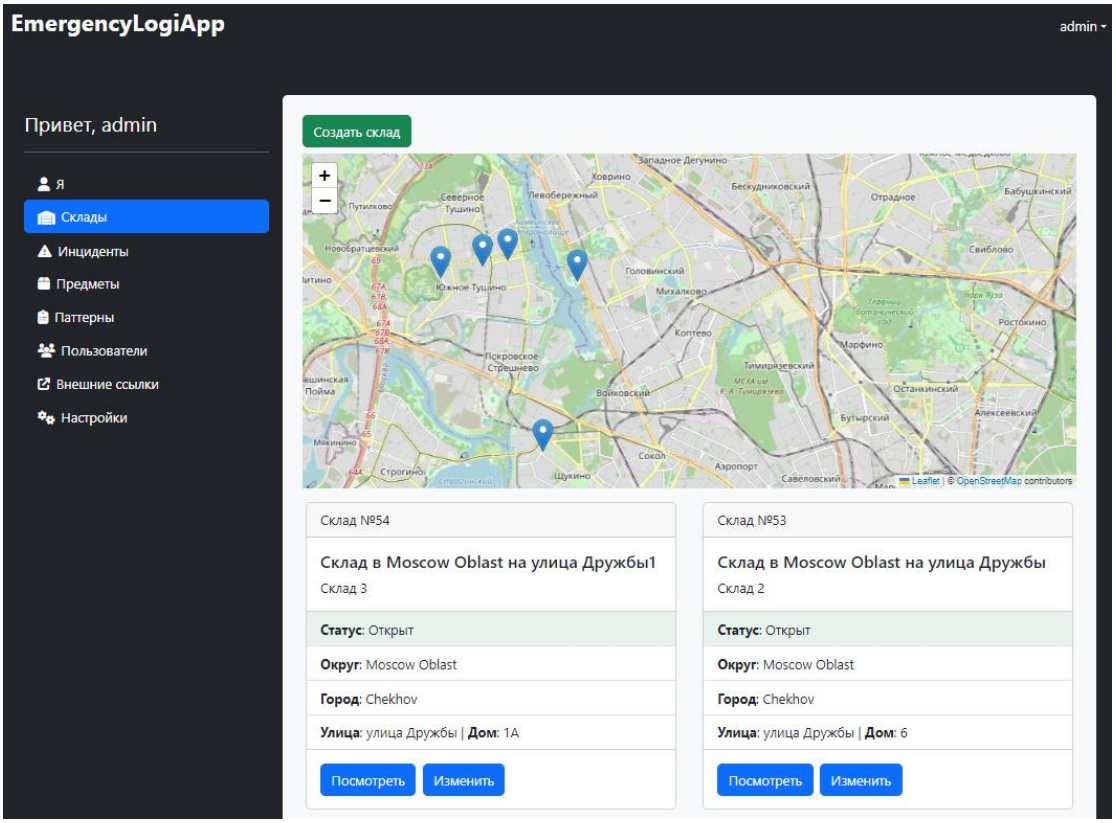

Рисунок В.1 – Страница со списком складов

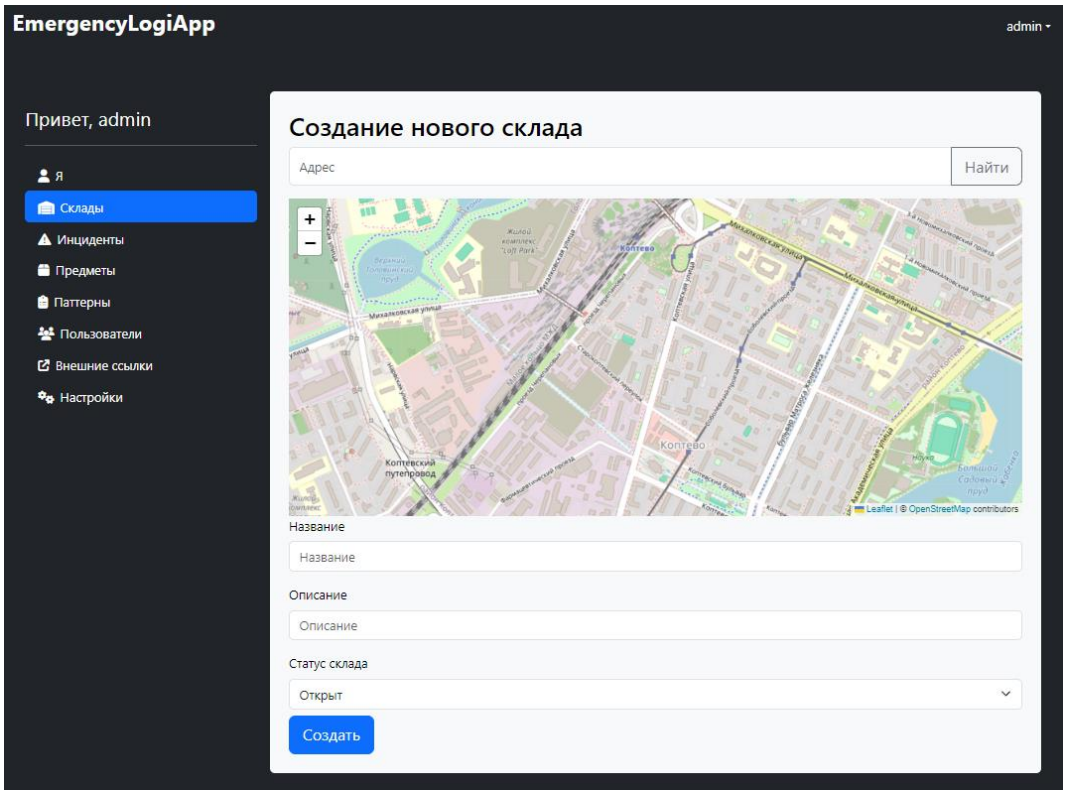

Рисунок В.2 – Страница создания нового склада

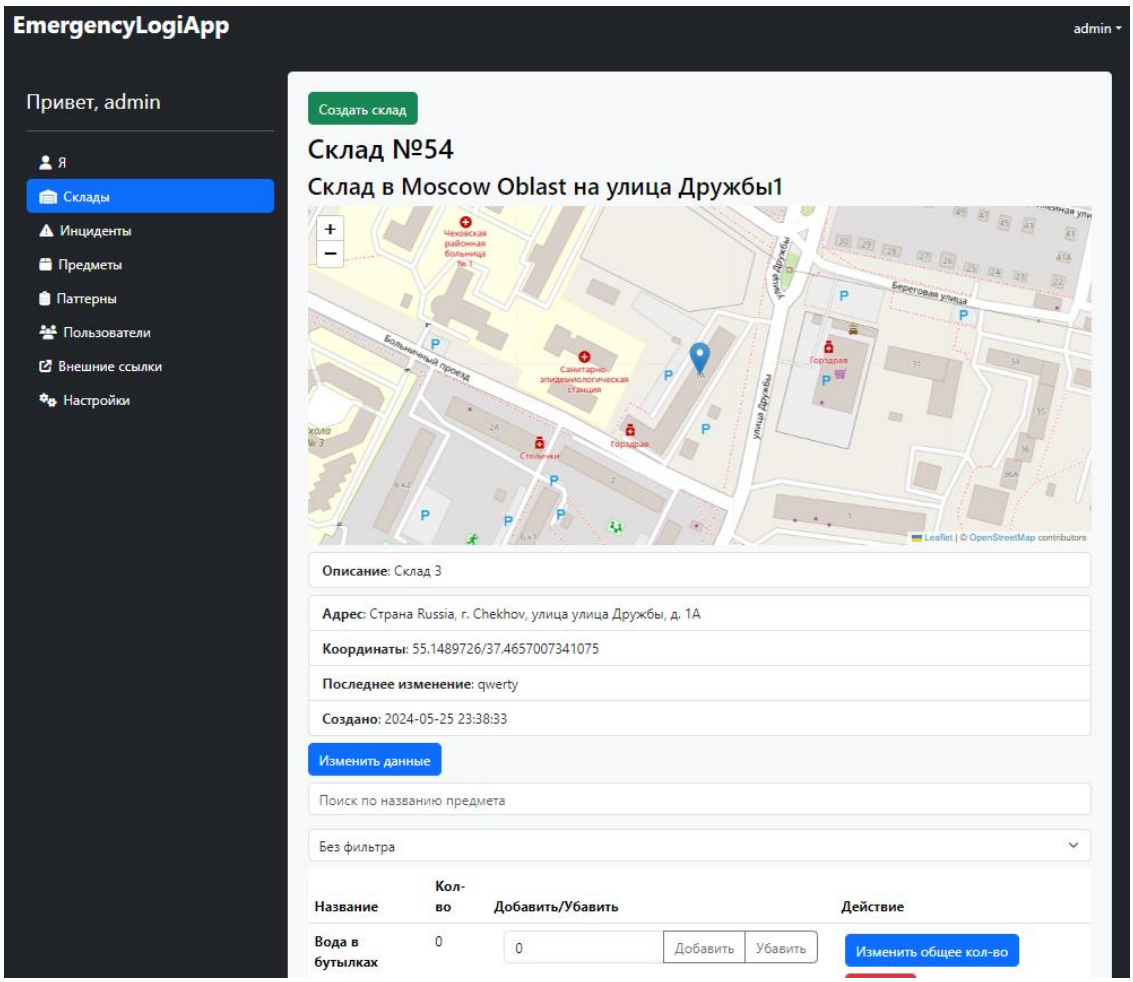

Рисунок В.3 – Страница склада

ПРИЛОЖЕНИЕ Г

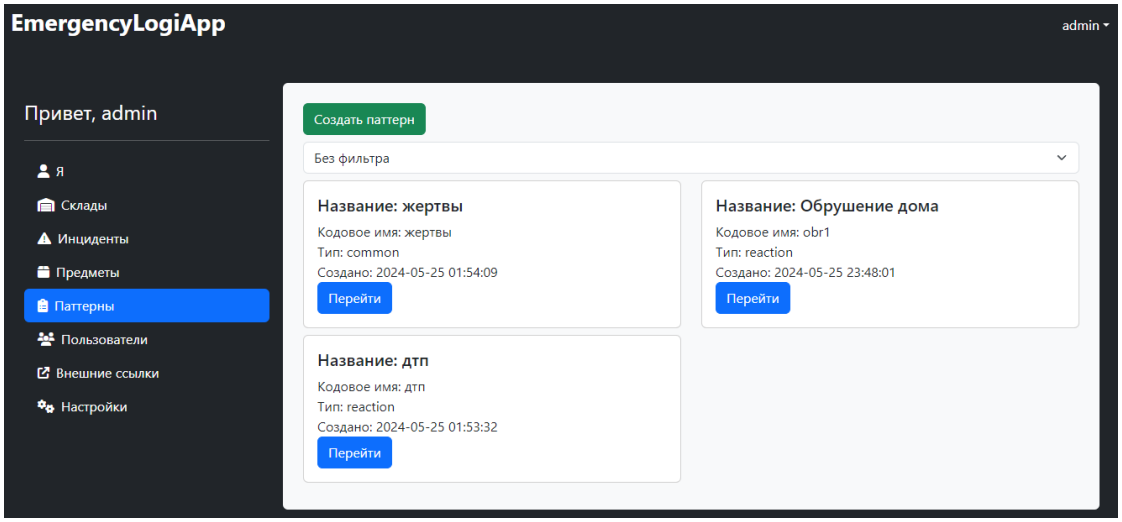

Рисунок Г.1 – Страница со списком паттернов происшествий

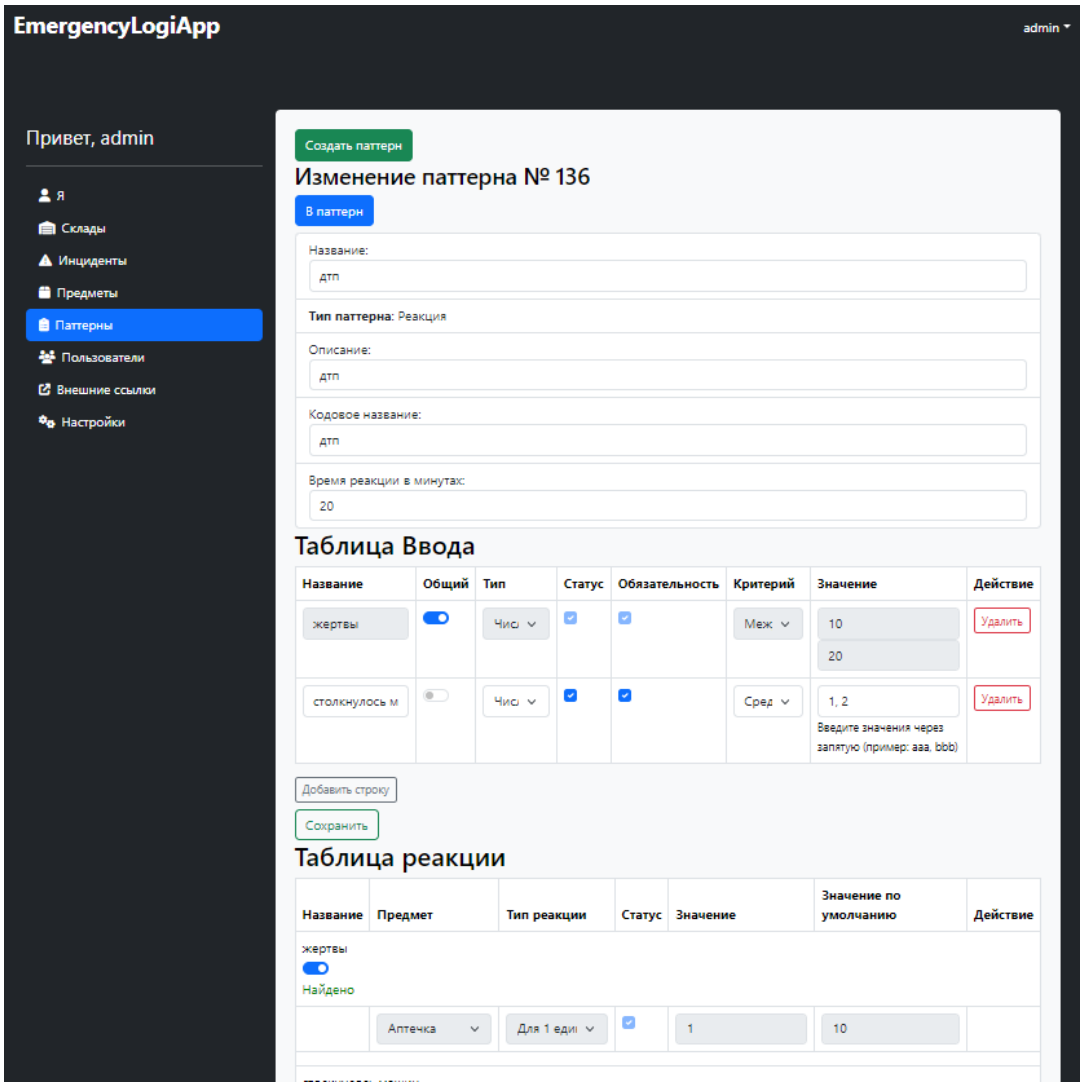

Рисунок Г.2 – Страница изменения паттерна происшествия

EmergencyLogiApp

admin

Привет, admin

- Я
- Склады
- Инциденты
- Предметы
- Паттерны
- Пользователи
- Внешние ссылки
- Настройки

Создать паттерн

Изменить паттерн

Паттерн № 136

Название: дтп

Тип паттерна: Реакция

Описание: дтп

Кодовое название: дтп

Время реакции в минутах: 20

Таблица Ввода

Название	Общий	Тип	Статус	Обязательность	Критерий	Значение
жертвы	✓	числовой	✓	✓	Между	От 10 до 20 включительно
столкнулось машин	✗	числовой	✓	✓	Среди	Среди 1, 2

Таблица реакции

Название	Предмет	Тип реакции	Статус	Значение	Значение по умолчанию
жертвы ✓ Найдено					
	Аптечка	Для 1 единицы	✓	1	10
столкнулось машин ✗					
	Бур	В общем	✓	2	

Рисунок Г.3 – Страница паттерна происшествия

ПРИЛОЖЕНИЕ Д

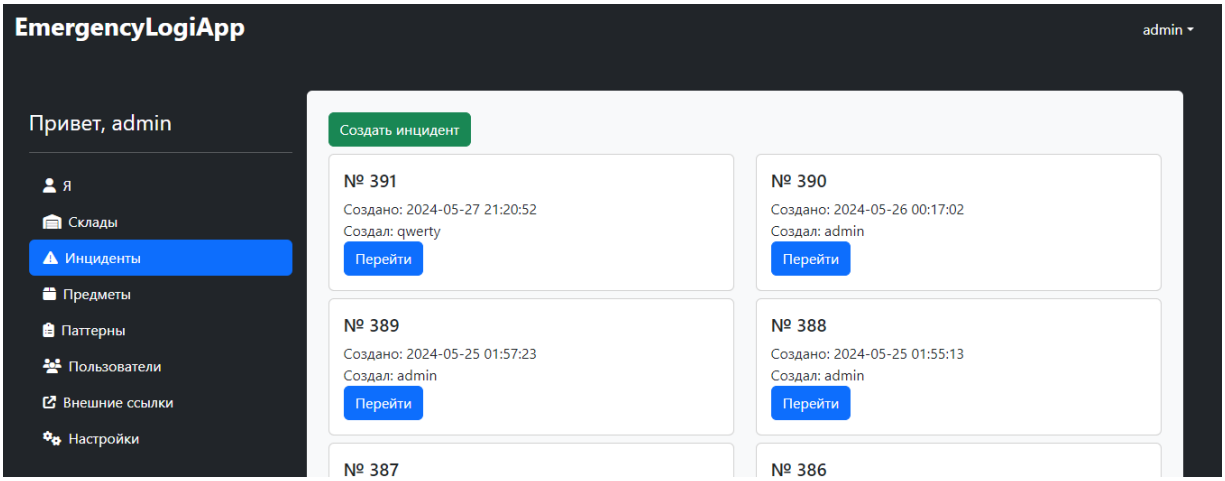

Рисунок Д.1 – Страница со списком инцидентов

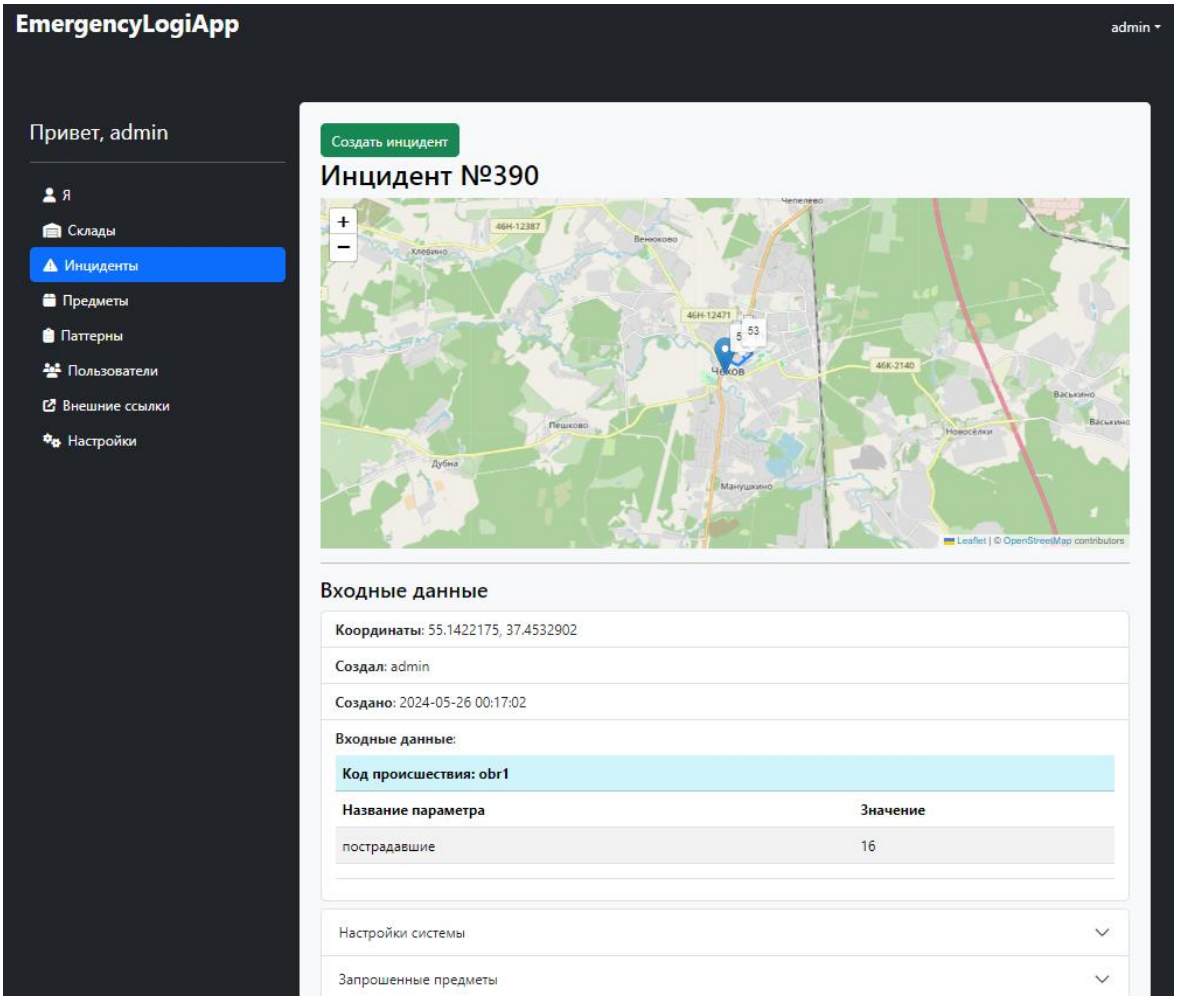

Рисунок Д.2 – Страница инцидента

ПРИЛОЖЕНИЕ Е

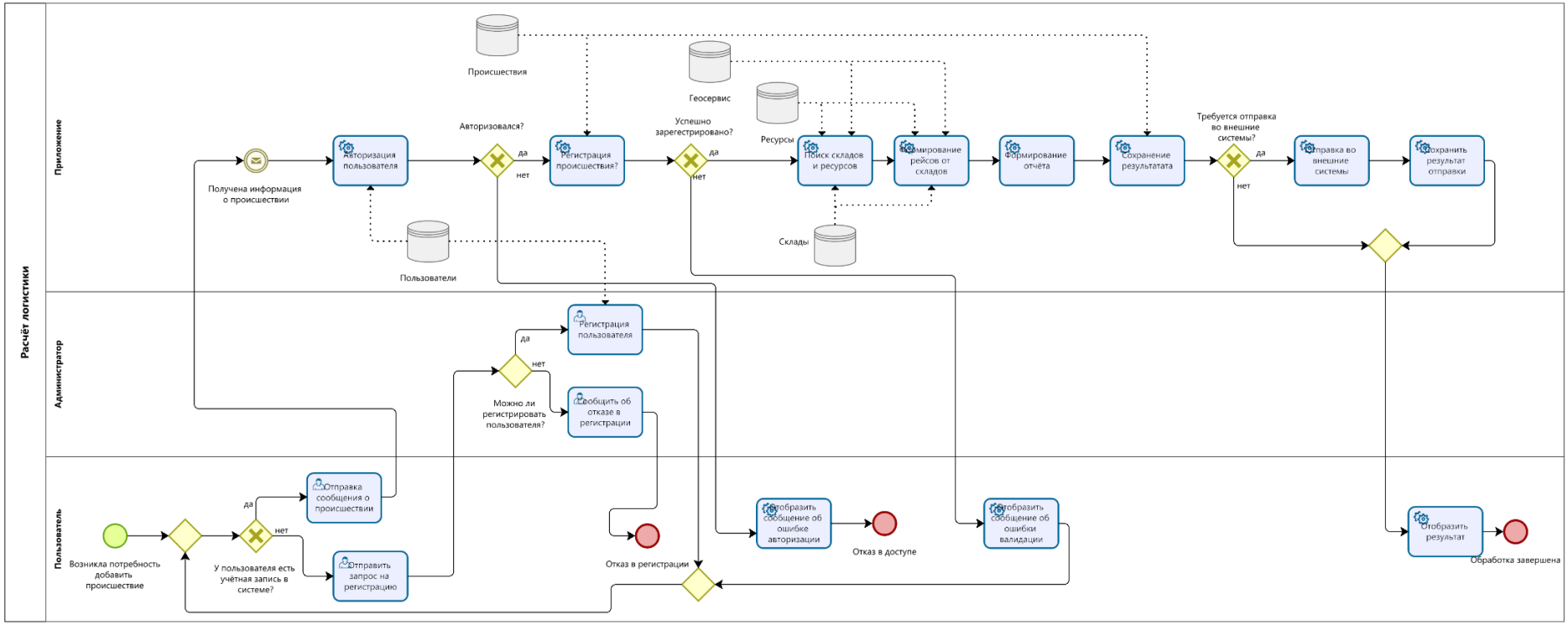

Рисунок Е.1 – Модель процесса «Рассчитать логистику»

ПРИЛОЖЕНИЕ Ж

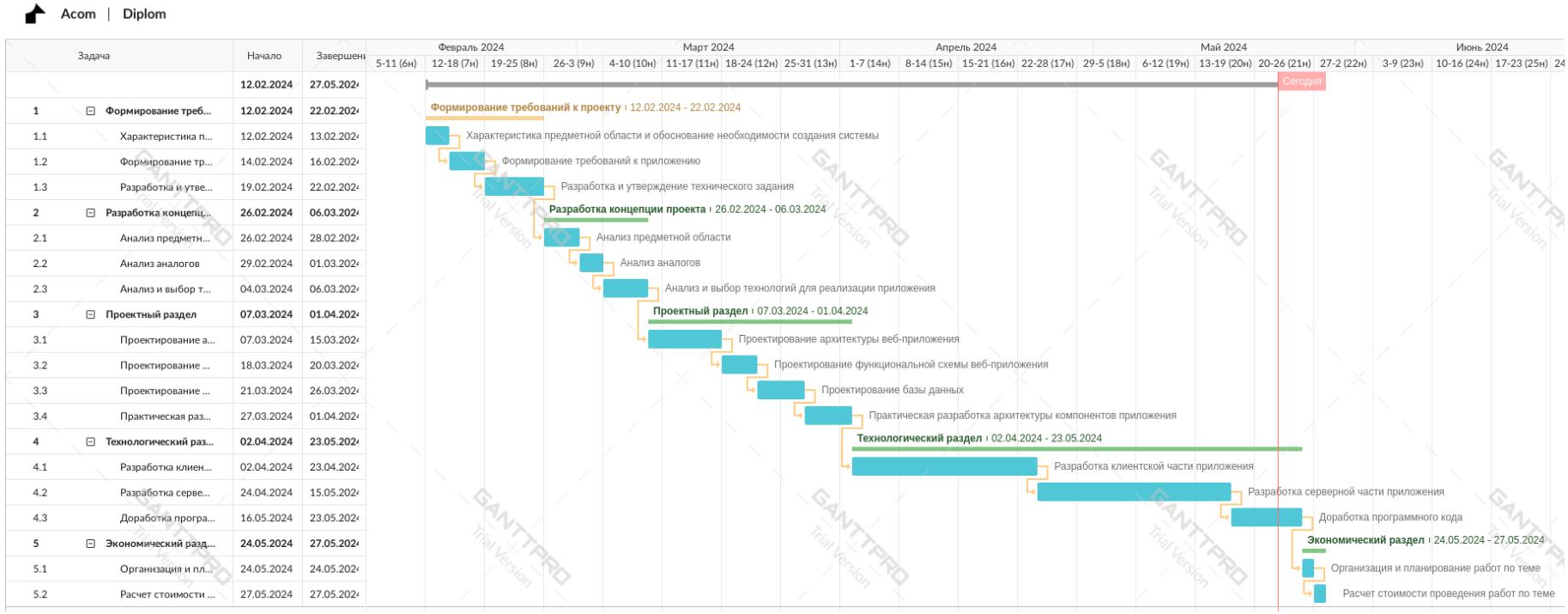

Рисунок Ж.1 – Модель процесса «Рассчитать логистику»

ПРИЛОЖЕНИЕ 3

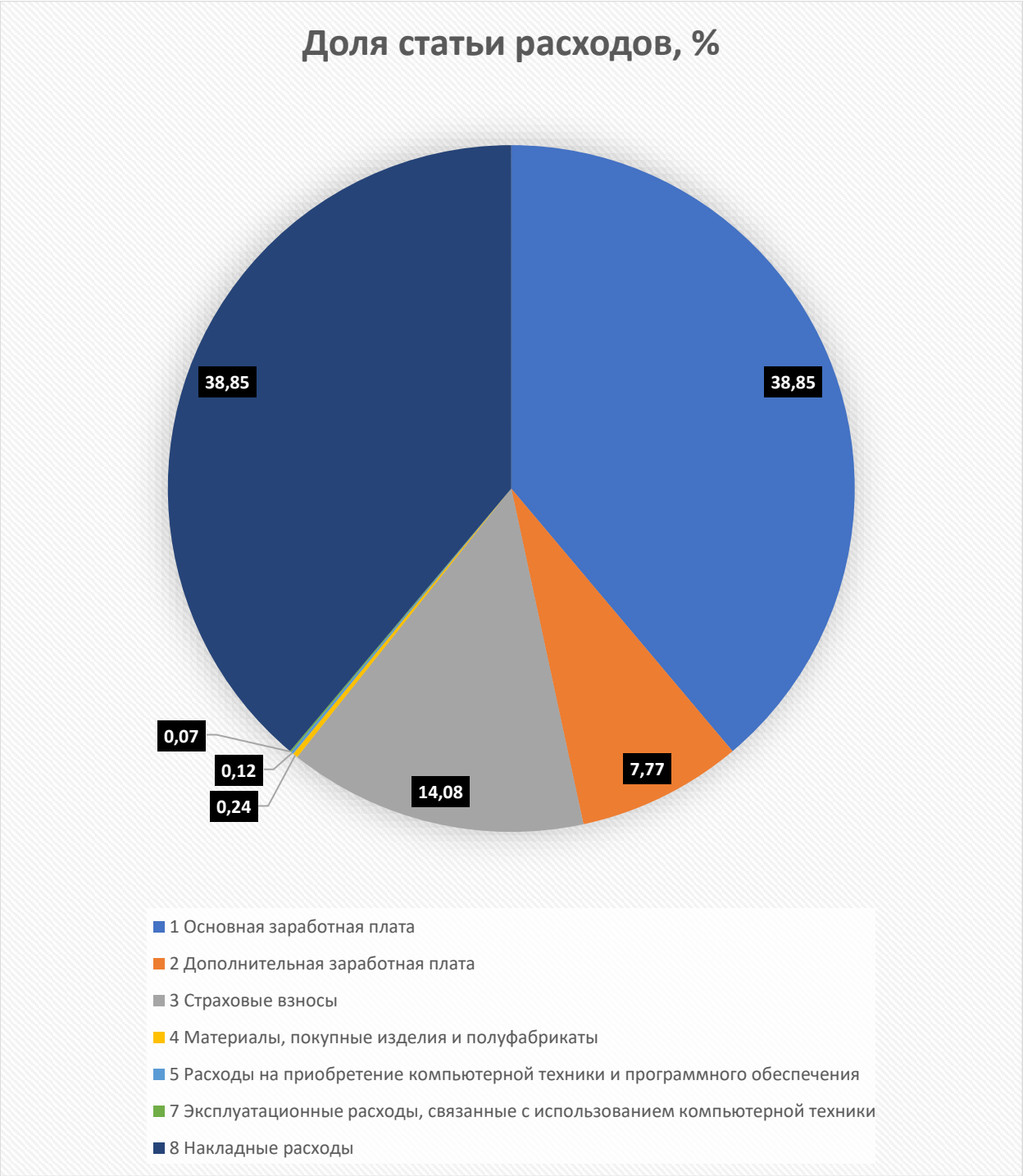

Рисунок 3.1 – Структура затрат по проекту