

КАФЕДРА

КУРСОВОЙ ПРОЕКТ ЗАЩИЩЕН С ОЦЕНКОЙ  
РУКОВОДИТЕЛЬ

---

должность, уч. степень,  
звание

---

подпись, дата

---

инициалы, фамилия

## ПОЯСНИТЕЛЬНАЯ ЗАПИСКА К КУРСОВОМУ ПРОЕКТУ

Система управления беспилотными устройствами по складу  
компании ЭТМ

по курсу: Проектирование программных систем

РАБОТУ ВЫПОЛНИЛ

СТУДЕНТ ГР.

---

подпись, дата

---

инициалы, фамилия

Санкт-Петербург 2024

## РЕФЕРАТ

Отчет 120 с., 49 рис., 3 прил.

### АВТОМАТИЗИРОВАННАЯ СИСТЕМА УПРАВЛЕНИЯ БЕСПИЛОТНЫМИ УСТРОЙСТВАМИ, СКЛАДСКОЕ ПОМЕЩЕНИЕ КОМПАНИИ ЭТМ, ИНТЕРАКТИВНАЯ КАРТА, ФУНКЦИИ, УПРАВЛЕНИЕ

Объектом исследования является процесс управления беспилотными устройствами на складе, включающий функции мониторинга и контроля местоположения дронов, автоматизированной отправки дронов для выполнения задач, обработки данных о статусе дронов и взаимодействия с интерактивной картой склада.

Цель работы – разработка автоматизированной информационной системы, которая будет оптимизировать управление беспилотными устройствами в условиях складского хозяйства.

В процессе работы было проведено исследование предметной области, которое позволило определить основные требования и функциональные возможности системы.

В результате работы была создана автоматизированная система управления беспилотными устройствами для компании ЭТМ, улучшающая операционную эффективность и безопасность на складе.

## СОДЕРЖАНИЕ

<b>ВВЕДЕНИЕ .....</b>	<b>9</b>
<b>1. ЦЕЛИ И НАЗНАЧЕНИЕ СОЗДАНИЯ АВТОМАТИЗИРОВАННОЙ СИСТЕМЫ.....</b>	<b>11</b>
<b>1.1 Назначение программы.....</b>	<b>11</b>
<b>1.2 Бизнес-функции, для автоматизации которых предназначена система .....</b>	<b>11</b>
<b>2. ПРОЕКТИРОВАНИЕ И КОМПЛЕКСНОЕ МОДЕЛИРОВАНИЕ СИСТЕМЫ.....</b>	<b>13</b>
<b>2.1 Характеристика и структура программы .....</b>	<b>13</b>
<b>2.2 Описание основных особенностей программы.....</b>	<b>15</b>
<b>2.3 Основные пользователи системы.....</b>	<b>16</b>
<b>2.4 Функциональное моделирование в методике IDEF0.....</b>	<b>17</b>
2.4.1 Контекстная диаграмма.....	17
2.4.2 Декомпозиция контекстной диаграммы .....	18
2.4.3 Декомпозиция задачи A1 .....	19
2.4.4 Декомпозиция задачи A2 .....	20
2.4.5 Декомпозиция задачи A3 .....	21
2.4.6 Декомпозиция задачи A4 .....	22
<b>2.5 Функциональное моделирование в методике DFD .....</b>	<b>23</b>
2.5.1 Моделирование процесса A23 «отправки задачи от сервера» .....	23
<b>2.6 Анализ результатов функционального моделирования в методике IDEF .....</b>	<b>24</b>
<b>2.7 Объектное моделирование в методике UML .....</b>	<b>25</b>
2.7.1 Диаграмма развертывания .....	25
2.7.2 Общий план склада ЭТМ .....	26
2.7.3 Диаграмма пакетов .....	27

2.7.4 Диаграмма классов .....	28
2.7.5 Диаграмма компонентов .....	29
2.7.6 Диаграмма деятельности .....	30
2.7.7 Диаграмма прецедентов .....	31
2.7.8 Диаграмма состояний .....	32
2.7.9 Диаграмма последовательности .....	33
<b>2.8 Словарь данных.....</b>	<b>34</b>
<b>3. ТРЕБОВАНИЯ К ИНФОРМАЦИОННОЙ СИСТЕМЕ .....</b>	<b>36</b>
<b>3.1 Требования к структуре АС в целом.....</b>	<b>36</b>
<b>3.2 Функциональные требования к системе .....</b>	<b>37</b>
3.2.1 Загрузка и запуск программы .....	37
3.2.2 Начало работы с системой .....	39
3.2.3 Автоматизированная отправка беспилотника к стеллажу .....	41
3.2.4 Ручное управление беспилотником .....	44
3.2.5 Автоматизация работы склада: импорт файла с задачами .....	47
3.2.6 Аварийная остановка работы системы .....	51
3.2.7 Возобновление работы системы .....	53
3.2.8 Создание отчёта .....	55
<b>4. РАЗРАБОТКА АВТОМАТИЗИРОВАННОЙ СИСТЕМЫ .....</b>	<b>59</b>
<b>4.1 Серверная часть системы .....</b>	<b>59</b>
4.1.1 Описание класса DroneController.....	60
4.1.2 Описание класса DroneObj.....	62
4.1.3 Описание класса LoginController .....	64
4.1.4 Описание класса ThymeleafProjectApplication .....	65
4.1.5 Описание класса WebSecurityConfig .....	68
<b>4.2 Клиентская часть системы .....</b>	<b>70</b>
4.2.1 Описание интерфейса Drones.html .....	71
4.2.2 Описание интерфейса Login.html .....	73

4.2.3 Описания функций Map.js.....	74
<b>ЗАКЛЮЧЕНИЕ .....</b>	<b>82</b>
<b>СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ .....</b>	<b>84</b>
<b>ПРИЛОЖЕНИЕ А.....</b>	<b>86</b>
<b>ПРИЛОЖЕНИЕ Б .....</b>	<b>87</b>
<b>ПРИЛОЖЕНИЕ В .....</b>	<b>106</b>

## Термины и определения

Термин	Определение
Беспилотник	Автоматизированное устройство, используемое для перемещения грузов на складе без прямого участия человека.
Складская зона	Специально отведённая область на складе, предназначенная для хранения, загрузки или разгрузки товаров. Включает зоны разгрузки, зарядки, хранения и офиса.
Аварийная остановка	Системная функция для немедленной остановки всех беспилотников в случае чрезвычайной ситуации.
Восстановление работы	Функция системы, позволяющая возобновить работу беспилотников после их аварийной остановки.
Интерактивная карта	Графическое представление складских зон и беспилотников в реальном времени, позволяющее пользователям управлять процессами перемещения и мониторинга.

## Перечень сокращений и обозначений

Сокращение/Обозначение	Описание
ЭТМ	Название компании, использующей систему управления беспилотными устройствами.
MAC OS 14 Sonoma	Операционная система, на которой работает программное обеспечение системы управления.
IntelliJ IDEA 2023.2.2	Версия интегрированной среды разработки программного обеспечения, требуемая для работы системы управления.
HTTPS	Протокол защищённой передачи гипертекста, используемый для обеспечения безопасности веб-доступа к системе управления.
PDF	Формат файла, используемый для сохранения отчетов, генерируемых системой.

LAN	<p>Локальная</p> <p>вычислительная сеть, через которую осуществляется подключение рабочих компьютеров и беспилотных устройств для обмена данными.</p>
-----	---



## ВВЕДЕНИЕ

Информационная система управления беспилотными устройствами на складе компании ЭТМ играет ключевую роль в оптимизации логистических и складских операций. Система предназначена для управления, мониторинга и автоматизации работы беспилотников, что позволяет ускорить процессы перемещения грузов, повысить точность операций и уменьшить риск человеческих ошибок.

Главная цель системы — повышение эффективности складской работы за счет автоматизации управления дронами. Внедрение такой системы позволяет значительно ускорить выполнение задач, связанных с перемещением и сортировкой товаров, а также повысить безопасность на складе.

Задачи внедрения системы управления беспилотными устройствами включают:

- Эффективное планирование и использование беспилотников для выполнения складских операций;
- Минимизация времени на выполнение складских задач и повышение их качества;
- Оперативное решение возникающих задач в процессе работы беспилотников;
- Обеспечение высокого уровня безопасности операций с участием дронов;
- Автоматизация процессов сбора данных и создания отчетов о выполненных операциях.

Для реализации проекта необходимо выполнить следующие задачи:

- Провести исследование предметной области для определения функциональных требований и проектирования системы;

- Анализировать процессы складской логистики для определения специфических требований к автоматизации;
- Осуществить моделирование работы системы и ее компонентов для оптимизации процессов;
- Выбрать подходящие технологические решения и инструменты для разработки системы;
- Разработать и тестировать систему в условиях, максимально приближенных к реальной эксплуатации на складе.

## **1. ЦЕЛИ И НАЗНАЧЕНИЕ СОЗДАНИЯ АВТОМАТИЗИРОВАННОЙ СИСТЕМЫ**

Полное наименование автоматизированной системы: "Система Управления Беспилотными Устройствами" на складе компании ЭТМ.

Сокращенное название: "СУБУ ЭТМ".

Цель внедрения — это оптимизация складских операций для повышения эффективности работы склада. Внедрение автоматизированной системы позволит ускорить процессы перемещения и учета товаров, уменьшить количество ошибок, связанных с человеческим фактором, и улучшить контроль за операциями с беспилотниками.

### **1.1 Назначение программы**

Назначение автоматизированной системы заключается в управлении беспилотными устройствами на складе, что включает функции контроля, мониторинга, и автоматического выполнения заданий по перемещению грузов между стеллажами и зонами склада.

### **1.2 Бизнес-функции, для автоматизации которых предназначена система**

Функциональность системы позволяет выполнять следующие операции:

- Регистрация и авторизация пользователей с различными уровнями доступа;
- Просмотр и управление статусами беспилотников в реальном времени;
- Автоматическое отправление беспилотников к заданным координатам стеллажей для загрузки и разгрузки грузов;

- Мониторинг и планирование маршрутов перемещения беспилотников на карте склада;
- Автоматическое управление исключительными ситуациями, такими как технические неполадки;
- Ведение журнала событий, связанных с работой беспилотников и обработкой грузов;
- Формирование и печать отчетов о выполненных операциях и статусах беспилотников.

## **2. ПРОЕКТИРОВАНИЕ И КОМПЛЕКСНОЕ МОДЕЛИРОВАНИЕ СИСТЕМЫ**

### **2.1 Характеристика и структура программы**

В информационную систему "Система Управления Беспилотными Устройствами" (СУБУ ЭТМ) входят следующие функциональные подсистемы:

- Подсистема управления беспилотниками: обеспечивает управление движением и операциями беспилотников на складе, включая автоматическое и ручное управление.
- Подсистема интерактивной карты склада: отображает в реальном времени расположение беспилотников, стеллажей и грузов, а также позволяет планировать и модифицировать маршруты движения.
- Подсистема мониторинга и логирования: фиксирует все операции, выполняемые беспилотниками, и обеспечивает возможность анализа и отчетности по данным действиям.
- Подсистема безопасности: контролирует доступ к системе, обеспечивает защиту данных и управляет правами пользователей.
- Подсистема обработки заданий: управляет заданиями для беспилотников, основанными на запросах пользователя или автоматизированных сценариях работы склада.

Информационный обмен между компонентами системы осуществляется через локальную сеть, что обеспечивает высокую скорость доступа и обновления данных в режиме реального времени. Коммуникация данных происходит через защищенные протоколы, интегрированные в рамках локальной сетевой инфраструктуры и серверов, расположенных непосредственно на территории предприятия.

Характеристики подсистем:

1. Подсистема управления беспилотниками: Эта подсистема является основной в системе СУБУ ЭТМ и предназначена для мониторинга и управления перемещениями беспилотников в реальном времени. Она включает функции для ручного управления беспилотниками через интерфейс пользователя, а также для автоматического выполнения заданий, таких как доставка грузов до указанных стеллажей или в разгрузочные зоны. Подсистема обеспечивает прием и обработку команд, а также отправку статусов операций на центральный сервер.

2. Подсистема интерактивной карты склада: отвечает за визуализацию местоположения всех активных беспилотников, стеллажей, и грузов на электронной карте склада. Пользователи могут в режиме реального времени видеть статус каждого беспилотника (активен, заряжается, в ожидании задачи) и управлять их перемещениями через этот интерфейс.

3. Подсистема мониторинга и логирования: фиксирует все события и операции, выполняемые беспилотниками, включая время старта, завершения задачи и любые ошибки или системные сбои. Данные логи используются для анализа эффективности работы системы, отладки и улучшения процессов обслуживания.

4. Подсистема безопасности: контролирует доступ к системе через авторизацию и аутентификацию, обеспечивает шифрование данных и защиту от неавторизованного доступа. Важной функцией является предотвращение неправомерных действий и соблюдение политик конфиденциальности и безопасности данных компании.

5. Подсистема обработки заданий: обрабатывает входящие запросы на перемещение грузов, планирует и оптимизирует маршруты беспилотников, учитывая текущее состояние склада и доступность путей. Подсистема также включает функции для автоматической отмены или перепланирования заданий в случае возникновения препятствий или ошибок в работе.

Каждая подсистема тесно интегрирована и взаимодействует с другими через централизованный сервер, что обеспечивает единую точку контроля за всей системой управления беспилотниками. Информационный обмен между подсистемами осуществляется через защищенные каналы связи, что гарантирует целостность и актуальность данных в системе.

## **2.2 Описание основных особенностей программы**

Авторизация и роли пользователей: при запуске программы СУБУ ЭТМ необходимо пройти процесс авторизации. Доступ к различным функциональным возможностям системы регулируется в зависимости от уровня доступа пользователя. Система поддерживает несколько типов пользовательских ролей, включая:

- Работник склада: Управление задачами, беспилотниками и мониторинг состояния беспилотников.
- Администратор системы: Настройка системы, управление пользователями и конфигурация безопасности.

Коммуникации и безопасность данных: Все данные передаются между клиентом и сервером по локальной вычислительной сети, используя https, что обеспечивает надежность и скорость обмена информацией.

Хранение данных: Вся информация, собираемая и обрабатываемая системой, хранится на сервере. Данные организованы в структурированной форме, что позволяет обеспечить их целостность и доступность для анализа и отчетности.

Интерфейс и взаимодействие: Программа обладает графическим пользовательским интерфейсом, который отображает интерактивную карту склада с текущим расположением беспилотников и стеллажей. Пользователи могут взаимодействовать с картой для отправки беспилотников к заданным точкам, а также отслеживать статус каждого устройства в реальном времени.

Эти особенности делают программу СУБУ ЭТМ надежным инструментом для оптимизации складских операций и управления беспилотными устройствами, повышая общую эффективность работы персонала и минимизируя риски, связанные с человеческим фактором.

### **2.3 Основные пользователи системы**

В системе СУБУ ЭТМ определены следующие ключевые роли пользователей, каждая из которых имеет свои специфические функции и уровень доступа к системным ресурсам:

- Складской работник: Эта роль предназначена для управления операциями на складе, включая отправку и прием грузов с помощью беспилотных устройств. Складские работники используют систему для:
  - Мониторинга и управления местоположением беспилотников в реальном времени.
  - Запуска задач по перемещению грузов между стеллажами и разгрузочными зонами.
  - Просмотра статуса задач и эффективности работы беспилотников.
  - Ведения учета складских запасов и оптимизации складских площадей.
- Администратор системы: Ответственен за настройку и обслуживание системы, управление пользователями и настройку параметров безопасности. Администраторы выполняют следующие функции:
  - Регистрация и управление учетными записями пользователей.
  - Настройка параметров доступа и ролей в системе.
  - Обновление и поддержка программного обеспечения, включая установку обновлений и патчей.



- Мониторинг состояния системы и ее компонентов для обеспечения стабильной и безопасной работы.
- Разработка отчетов по эффективности использования ресурсов и оптимизации работы склада.

Каждая из ролей в системе СУБУ ЭТМ имеет свои уникальные инструменты и интерфейсы, адаптированные под специфические задачи и потребности пользователей. Это обеспечивает эффективное взаимодействие всех участников процесса и повышает общую продуктивность работы склада.

## 2.4 Функциональное моделирование в методике IDEF0

Методика IDEF0 была применена для создания функциональной модели СУБУ ЭТМ, чтобы детально представить структуру управления беспилотными устройствами на складе, функции системы и потоки информации, управляющие этими функциями. Целью моделирования является визуализация процессов управления и взаимодействий внутри системы, облегчая понимание и последующее совершенствование системы.

### 2.4.1 Контекстная диаграмма

Контекстная диаграмма системы для представления поступающей и исходящей информации в систему.



Рисунок 1 - Контекстная диаграмма

## 2.4.2 Декомпозиция контекстной диаграммы

Декомпозиция контекстной диаграммы в методологии IDEF0 для СУБУ ЭТМ предназначена для более детального анализа и понимания каждой функции системы. Начиная с контекстной диаграммы уровня A0, мы разделяем систему на основные компоненты или функции, каждая из которых обрабатывает конкретные аспекты управления беспилотными устройствами на складе. Это разделение помогает углубиться в специфику работы каждой подсистемы и определить взаимодействие между ними.

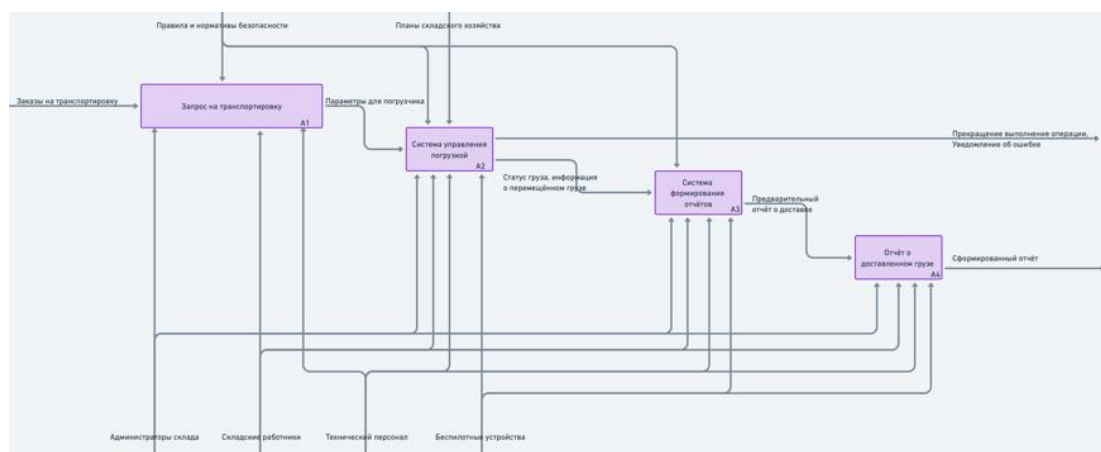


Рисунок 2 - Декомпозиция контекстной диаграммы

### 2.4.3 Декомпозиция задачи A1

Декомпозиция задачи «запрос на транспортировку» на 3 подзадачи.

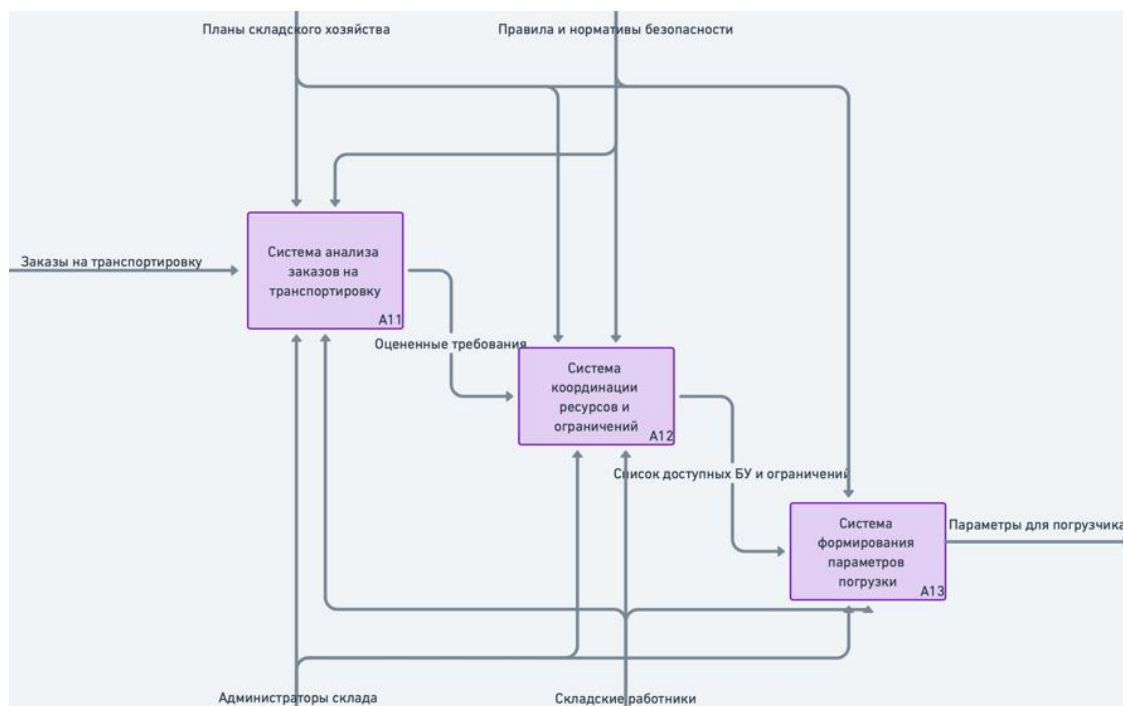


Рисунок 3 - Декомпозиция задачи A1

#### 2.4.4 Декомпозиция задачи A2

Декомпозиция задачи «система управления погрузкой» на 3 подзадачи.

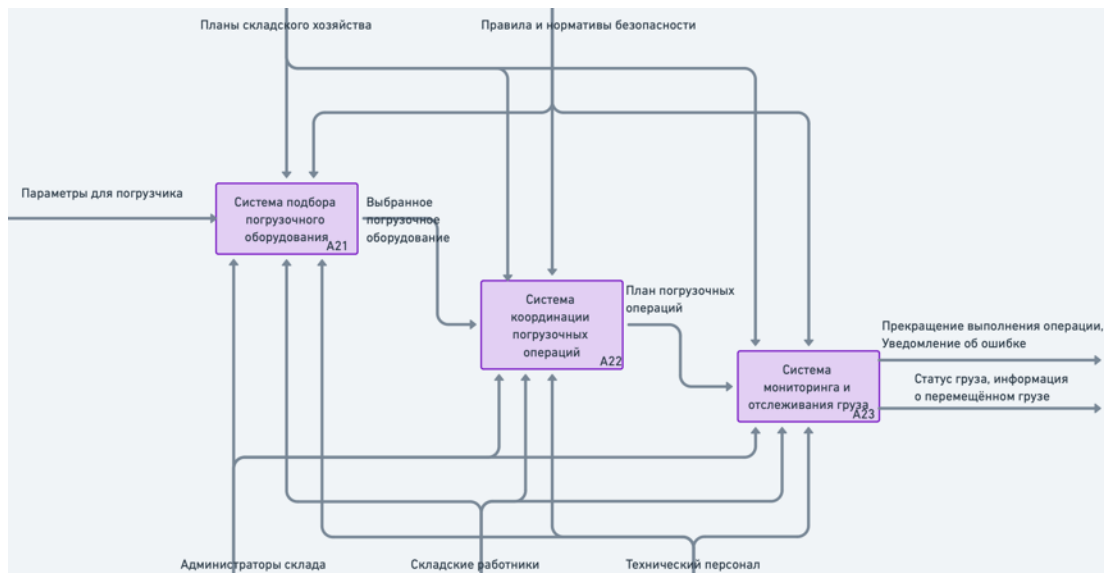


Рисунок 4 - Декомпозиция задачи A2

## 2.4.5 Декомпозиция задачи А3

Декомпозиция задачи «система формирования отчетов» на 3 подзадачи.

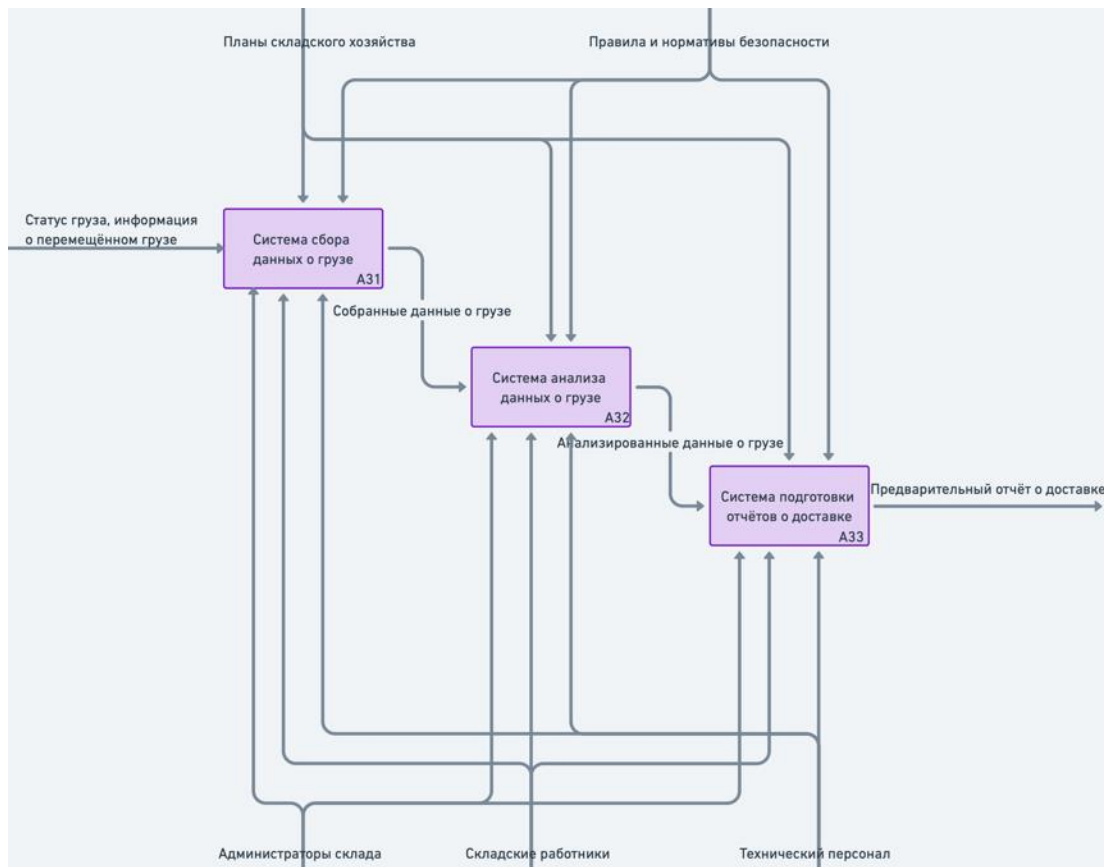


Рисунок 5 - Декомпозиция задачи А3

## 2.4.6 Декомпозиция задачи А4

Декомпозиция задачи «отчет о доставленном грузе» на 3 подзадачи.

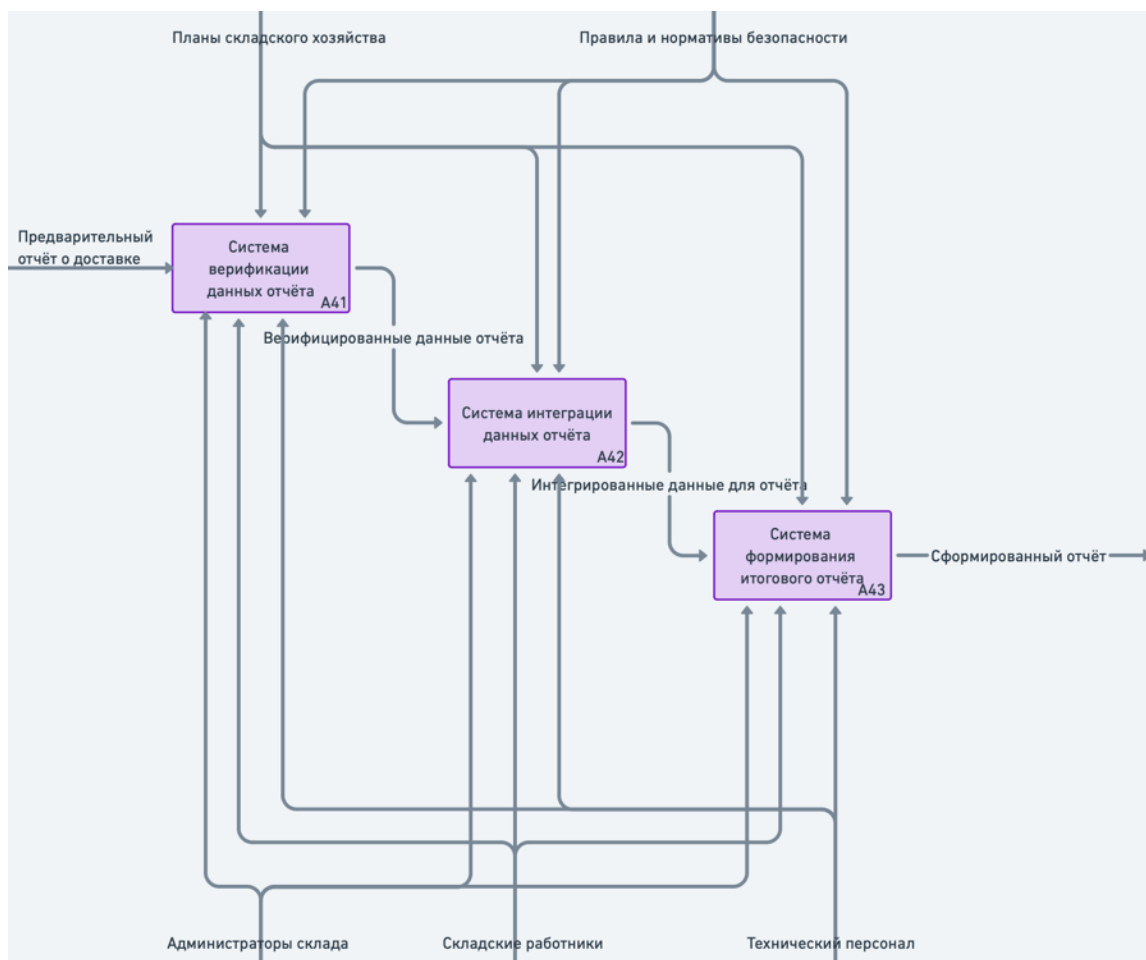


Рисунок 6 - Декомпозиция задачи А4

## 2.5 Функциональное моделирование в методике DFD

DFD диаграммы в отличие от других нотаций позволяют визуально показать все процессы с точки зрения данных. DFD-модели могут быть использованы в дополнение к модели IDEF0 для более наглядного отображения текущих операций документооборота в корпоративных системах обработки информации. Диаграммы потоков данных являются основным средством моделирования функциональных требований к проектируемой системе.

### 2.5.1 Моделирование процесса A23 «отправки задачи от сервера»



Рисунок 7 - Моделирование процесса A23

## **2.6 Анализ результатов функционального моделирования в методике IDEF**

Функциональное моделирование системы "Система Управления беспилотными устройствами" ЭТМ с применением методики IDEF и DFD позволило комплексно анализировать и структурировать бизнес-процессы, связанные с управлением и мониторингом беспилотных аппаратов на складах. Создание контекстной диаграммы IDEF0 (A0):

Диаграмма представила систему как единый функциональный блок с основными входами, выходами, механизмами и управлениями, позволяя увидеть взаимодействие с внешней средой и основные потоки информации.

Декомпозиция на подсистемы:

Диаграммы декомпозиции (A1, A2, A3 и далее) детализировали каждый из основных компонентов системы, раскрывая внутреннюю структуру и специфические функции, такие как диспетчеризация, мониторинг состояния беспилотников, и логистика маршрутов.

Применение IDEF3 и DFD:

DFD (диаграммы потоков данных) сфокусировались на хранении, обработке и передаче данных в системе, обеспечивая понимание информационных потоков и их безопасности.

Применение методик IDEF и DFD позволило не только глубоко понять структуру и функционирование "Системы Управления беспилотными устройствами" ЭТМ, но и выявить ключевые точки для оптимизации и улучшения. Интеграция этих подходов дала возможность создать четкую и функционально полную модель системы, которая может быть использована для последующей разработки, внедрения и масштабирования системы. Эти модели также обеспечивают необходимую основу для разработки технической документации и материалов для пользователей и администраторов системы.



## 2.7 Объектное моделирование в методике UML

### 2.7.1 Диаграмма развертывания

Используется для визуализации физического распределения компонентов системы по устройствам или узлам сети. Это помогает понять, как компоненты взаимодействуют на уровне оборудования, а также как распределены ресурсы и зависимости.

Пользовательское приложение: Используется человеком для ввода данных о грузе и месте его доставки на складе.

Сервер обработки данных: Обработывает введенные данные, оптимизирует маршрут и выбирает подходящий беспилотный погрузчик для выполнения задачи. Также получает отчеты от погрузчика и формирует итоговые отчеты.

Беспилотный погрузчик: Выполняет задачи по перемещению грузов, поддерживает связь с сервером во время выполнения задачи и отправляет отчет о выполнении.

База данных отчетов: Хранит отчеты, сформированные сервером обработки данных.



Рисунок 8 - Диаграмма развертывания

## 2.7.2 Общий план склада ЭТМ

Модель склада для визуального представления.

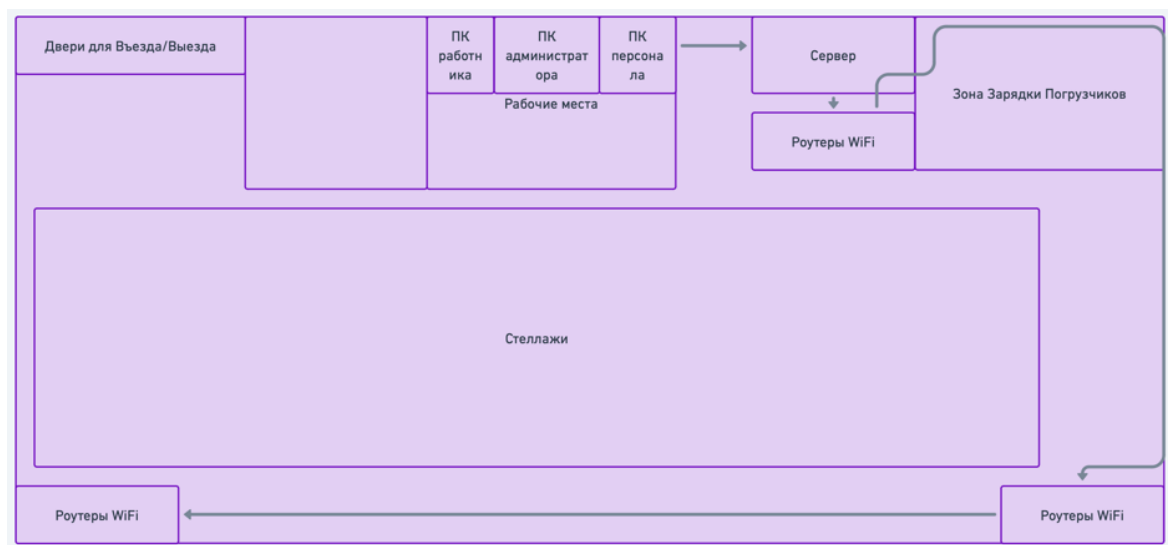


Рисунок 9 - Общий план склада ЭТМ

### 2.7.3 Диаграмма пакетов

Предназначена для структурирования элементов модели в группы, обычно используется для организации пространства имен в крупных системах. Диаграмма пакетов помогает управлять зависимостями между частями системы, упрощая понимание и обновление архитектуры.

Пользовательское приложение: Содержит интерфейс пользователя и логику ввода данных.

Сервер обработки данных: Включает серверную логику, базу данных склада, модуль оптимизации маршрутов и модуль обработки команд.

Беспилотный погрузчик: Состоит из системы навигации, контроллера управления, модуля связи и системы мониторинга.

База данных отчетов: Содержит хранилище данных, модуль обработки отчетов и интерфейс доступа.



Рисунок 10 - Диаграмма пакетов

## 2.7.4 Диаграмма классов

Отображает структуру системы на уровне классов, их атрибутов, методов и взаимосвязей между ними. Это ключевой инструмент объектно-ориентированного проектирования, позволяющий детализировать как данные, так и поведение системы.

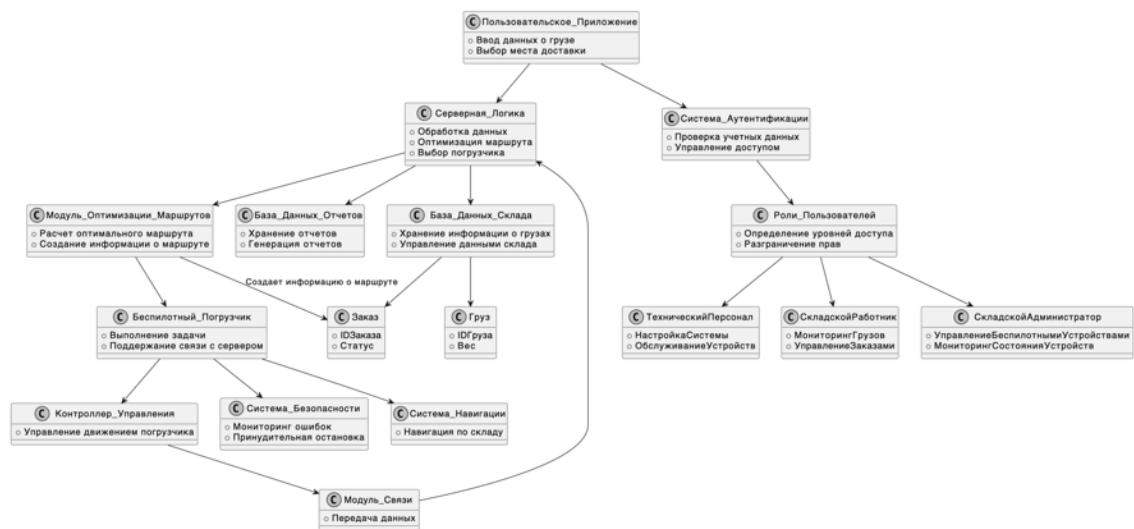


Рисунок 11 - Диаграмма классов

## 2.7.5 Диаграмма компонентов

Описывает организацию и взаимодействие высокоуровневых компонентов системы, включая библиотеки, пакеты и службы. Диаграмма компонентов часто используется для планирования системных интеграций и управления зависимостями.



Рисунок 12 - Диаграмма компонентов

## 2.7.6 Диаграмма деятельности

Используется для моделирования бизнес-процессов или рабочих процедур, показывая поток управления от действия к действию. Она помогает анализировать рабочие процессы, определять узкие места и улучшать процедуры.



Рисунок 13 - Диаграмма деятельности

### 2.7.7 Диаграмма прецедентов

Представляет взаимодействия между актерами и системой, выделяя функциональные требования в форме прецедентов (сценариев использования). Эта диаграмма помогает разработчикам и клиентам лучше понять, какие возможности предоставляет система.

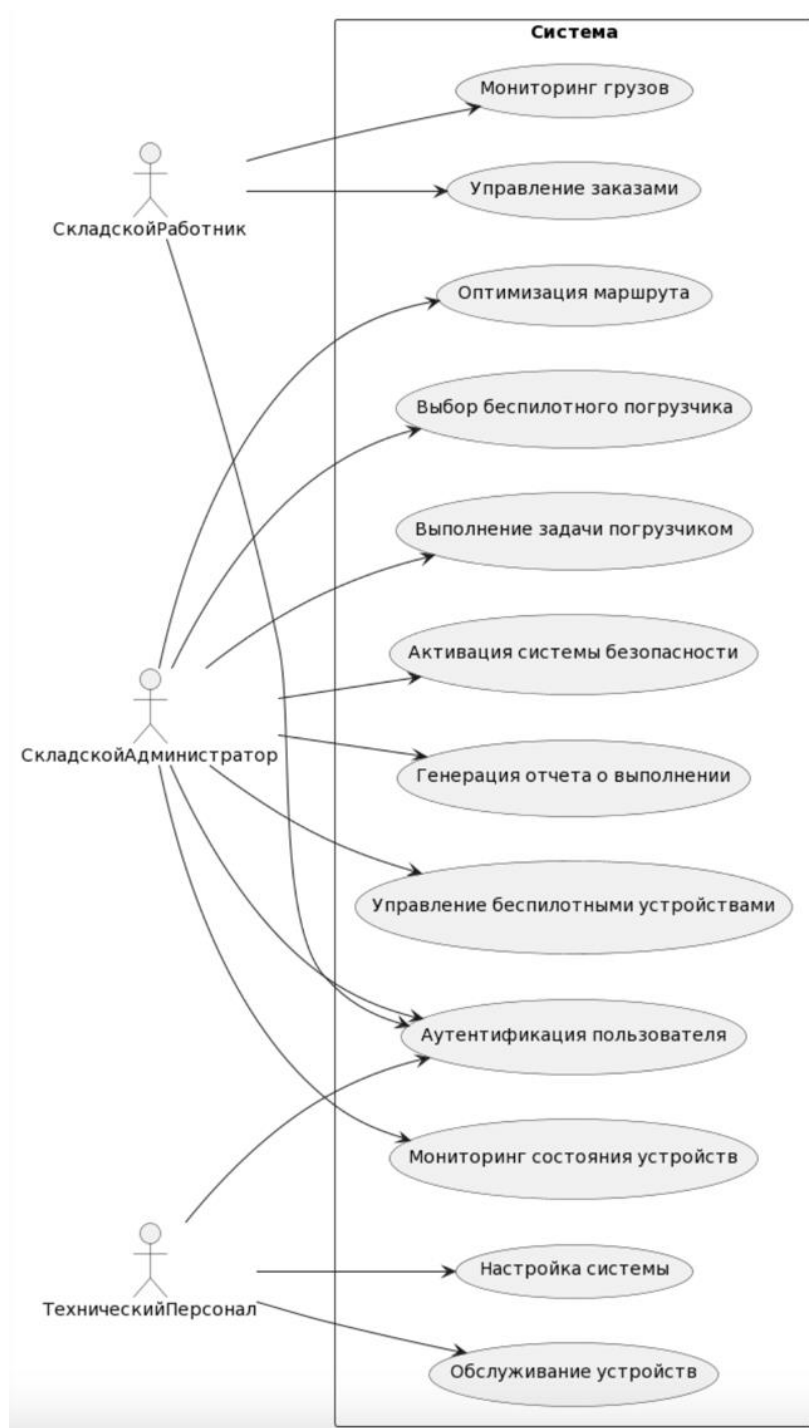


Рисунок 14 - Диаграмма прецедентов

## 2.7.8 Диаграмма состояний

Отображает изменения состояния объектов в системе в ответ на события. Эта диаграмма особенно полезна для проектирования реактивных систем, где поведение объектов тесно связано с изменением их состояний.

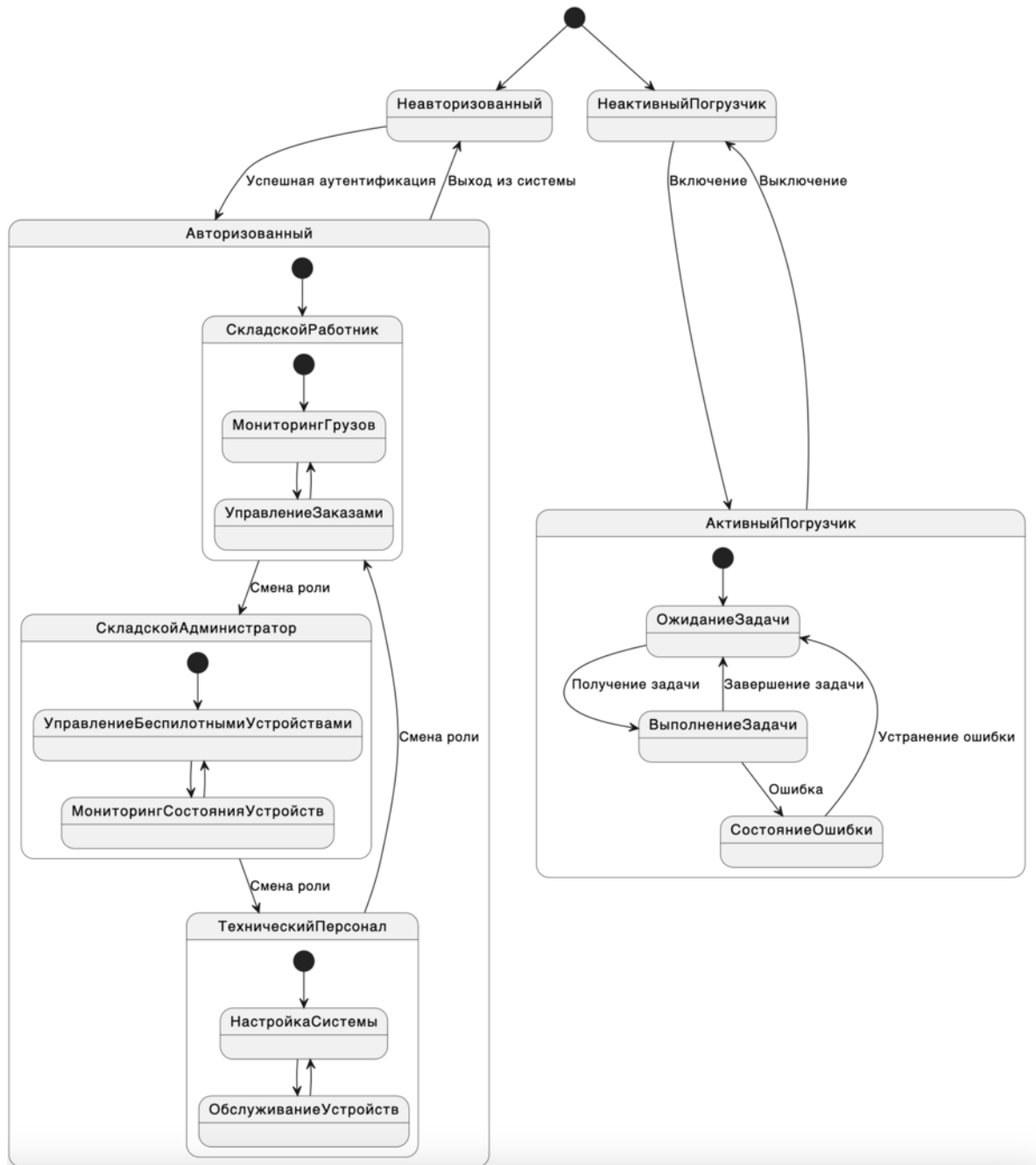


Рисунок 15 - Диаграмма состояний



## 2.7.9 Диаграмма последовательности

Иллюстрирует взаимодействие объектов во времени, показывая, как процессы и объекты взаимодействуют через отправку и получение сообщений. Особенно ценна при анализе конкретных рабочих процессов и при трассировке проблем в функционировании системы.

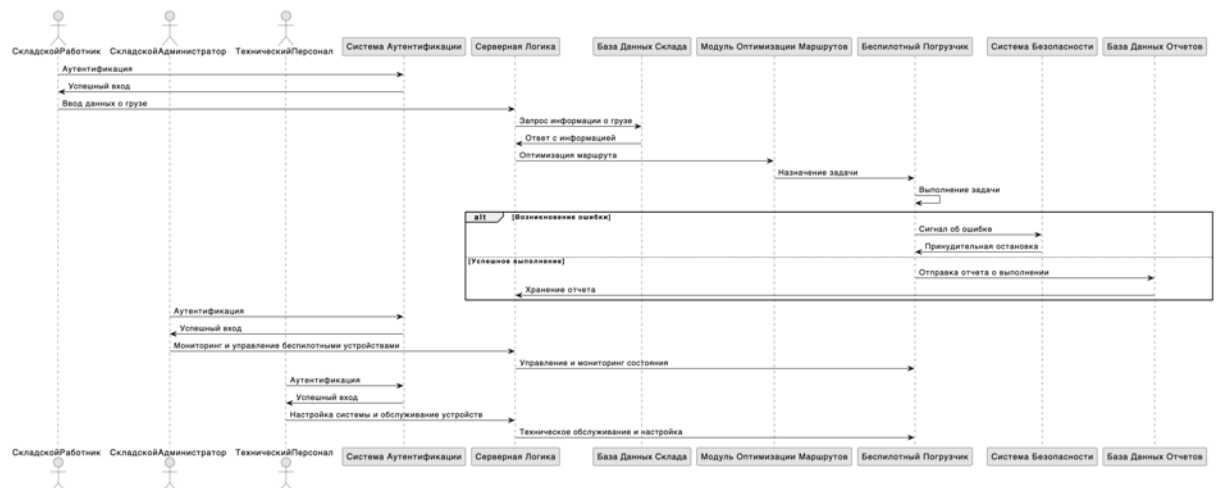


Рисунок 16 - Диаграмма последовательности

## 2.8 Словарь данных

Эта таблица отражает ключевые аспекты системы управления, включая типы данных и форматы, используемые для управления беспилотными устройствами, администрирования системы, и создания отчетов.

Наименование	Описание
Имя пользователя	Текстовый тип, длина до 255 символов
Номер стеллажа	Текстовый тип, формат буква (R/G/B) за которой следуют 3 цифры
Идентификатор беспилотника	Текстовый тип, до 10 символов, уникальное название беспилотника
Дата и время операции	Тип дата и время, формат: DD.MM.YYYY HH  , используется для отчетов
Статус беспилотника	Строковый тип, значения: в работе, неактивен, на обслуживании
Тип операции	Текстовый тип, длина до 100 символов, описание задачи беспилотника
Логин	Текстовый тип, длина до 255 символов
Пароль	Текстовый тип, длина до 255 символов, включает спецсимволы и цифры

Импорт номеров стеллажей из файла	Формат файла: текстовый, содержит строки с номерами стеллажей
Отчёт о деятельности	PDF-формат, содержит данные о всех операциях за выбранный период

### **3. ТРЕБОВАНИЯ К ИНФОРМАЦИОННОЙ СИСТЕМЕ**

#### **3.1 Требования к структуре АС в целом**

Информационная система "Система Управления беспилотными устройствами" ЭТМ использует клиент-серверную архитектуру. В данной архитектуре клиент (складской работник или администратор) взаимодействует с сервером для выполнения операций, таких как управление беспилотниками, отслеживание их местоположения и статуса, а также получение отчетов.

Архитектура информационной системы включает в себя следующие компоненты:

- Клиенты: приложения, установленные на рабочих станциях пользователей, отправляющие запросы к серверу. Эти приложения позволяют пользователям управлять беспилотниками, просматривать интерактивную карту склада и получать отчеты.

- Сервер: мощный компьютер в локальной сети либо хостинг, который обрабатывают запросы от клиентов и предоставляет необходимые данные и функциональность. Сервер отвечает за обработку данных, хранение информации о беспилотниках, стеллажах и выполненных операциях.

- Хранение данных: В системе используется централизованное хранение данных на сервере, где хранится вся критическая информация, включая данные о стеллажах, их координатах, а также подробности о беспилотниках, такие как их статусы и идентификационные данные. Это обеспечивает быстрый доступ к информации, необходимой для оперативного управления складом и мониторинга задач. Отчеты, генерируемые системой, автоматически сохраняются на рабочие станции пользователей для последующего просмотра и анализа, что позволяет администраторам и складским работникам оценивать продуктивность работы и принимать обоснованные управленческие решения на основе актуальных данных.

– Сеть: локальная вычислительная сеть (LAN) соединяет клиенты и серверы, обеспечивая высокоскоростную передачу данных и команд между устройствами. Это позволяет обеспечить мгновенное взаимодействие пользователей с беспилотниками и мониторинг текущего состояния склада.

Эта структура системы предназначена для обеспечения эффективного и безопасного управления беспилотными устройствами в условиях складской логистики, максимизируя производительность и минимизируя возможность ошибок и простоев.

## **3.2 Функциональные требования к системе**

### **3.2.1 Загрузка и запуск программы**

После запуска системы управления беспилотными устройствами, на экране пользователя открывается экранная форма регистрации, как показано на рисунке 17. На этом экране работнику необходимо ввести логин и пароль, предоставленные его начальством или IT-отделом, поскольку доступ к системе возможен только после предварительной регистрации пользователя в системе.

При вводе верных учетных данных и подтверждении через кнопку «Войти», система авторизует пользователя согласно его роли, после чего открывается главное окно приложения, содержание которого адаптируется под права и возможности пользователя.

В случае ввода неверных данных, система сохраняет экран регистрации активным, предоставляя пользователю возможность корректировки учетных данных. Этот процесс повышает удобство использования системы и способствует обеспечению безопасности.



## Вход в систему управления беспилотными устройствами

Имя пользователя:

Пароль:

Войти

Рисунок 17 – форма авторизации

### 3.2.2 Начало работы с системой

После успешной авторизации в системе "Система Управления беспилотными устройствами", пользователь попадает на главный экран, который включает в себя несколько ключевых элементов интерфейса, предоставляя комплексные инструменты для управления и мониторинга беспилотников на складе. На рисунке 18 представлен вид основного экрана системы, который доступен пользователю после входа в систему.

Основные элементы интерфейса:

- Заголовок страницы: отображает логотип компании и название системы, а также текущую роль пользователя, что позволяет легко идентифицировать уровень доступа к функциям системы.
- Интерактивная карта склада: Центральный элемент интерфейса, где визуализированы зоны склада, стеллажи с указанием их номеров, и текущее расположение беспилотников. Карта позволяет пользователю быстро оценить распределение ресурсов и эффективность текущей конфигурации склада.
- Кнопки управления:
- Отправить беспилотник: Пользователь может ввести номер стеллажа и отправить беспилотник для выполнения задачи, что автоматизирует процесс сбора и доставки товаров.
- Импорт файла стеллажей: позволяет загрузить список стеллажей, к которым необходимо направить беспилотники, облегчая планирование и оптимизацию рабочих процессов на складе.
- Дополнительные меню:
- Лог событий: Раскрывающееся меню, в котором отображаются все события, произошедшие на складе, включая действия беспилотников и системных сообщений.

- Список беспилотников: предоставляет детализированную информацию о каждом беспилотнике.
- Кнопки аварийного управления:
- Аварийная остановка: позволяет немедленно остановить все операции беспилотников в случае чрезвычайной ситуации, обеспечивая безопасность операций на складе.
- Восстановление работы: используется для возобновления работы системы после аварийной остановки или других системных сбоев.

Этот пользовательский интерфейс спроектирован таким образом, чтобы максимально упростить управление сложными процессами на складе, предоставляя пользователям все необходимые инструменты для эффективной работы с беспилотными устройствами. Все элементы интерфейса интуитивно понятны и доступны, что минимизирует время на обучение и интеграцию новых пользователей в систему.

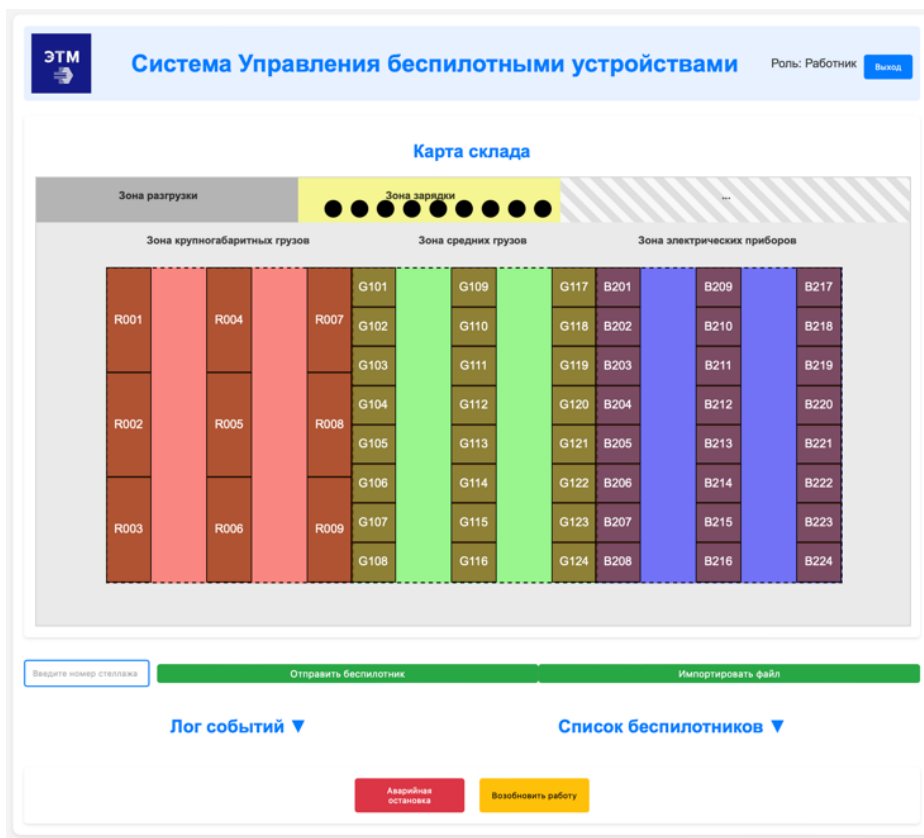


Рисунок 18 – Общий вид приложения



### **3.2.3 Автоматизированная отправка беспилотника к стеллажу**

Этот удобный и интуитивно понятный функционал позволяет пользователям мгновенно направить беспилотный погрузчик к конкретному стеллажу для забора груза. Процесс проиллюстрирован на рисунках 19-21.

Алгоритм работы:

1. **Выбор стеллажа:** Пользователь вводит название необходимого стеллажа в специальное поле ввода на интерфейсе программы.
2. **Отправка беспилотника:** после ввода информации пользователь активирует функцию, нажимая на кнопку "Отправить беспилотник". Система автоматически принимает запрос и незамедлительно отправляет беспилотник в указанное место.
3. **Автоматическое движение:** Беспилотный погрузчик самостоятельно следует к выбранному стеллажу по оптимальному маршруту, используя данные о местоположении и доступности.
4. **Поднятие груза и доставка:** По прибытии к стеллажу дрон автоматически выполняет процесс загрузки и включает режим доставки. Он переносит груз к зоне разгрузки, где груз размещается на нужном месте.
5. **Возврат к зоне зарядки:** после завершения задачи беспилотник возвращается в зону зарядки, где он подзаряжается в ожидании следующего задания.

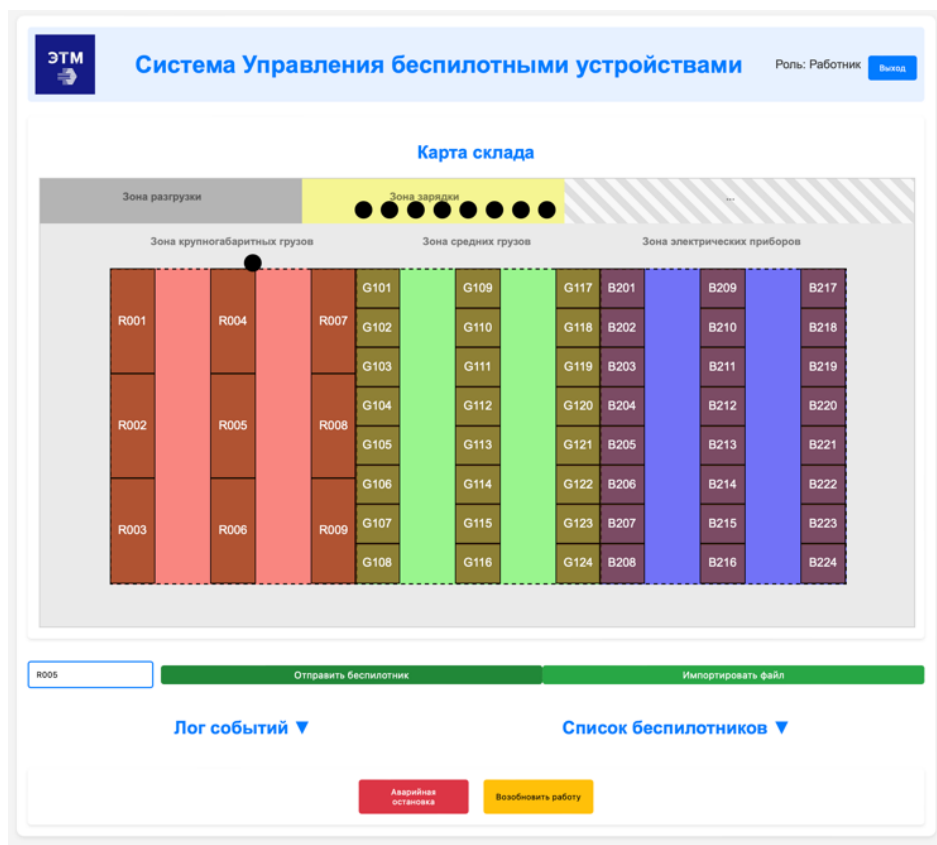


Рисунок 19 – ввод стеллажа и активация функции

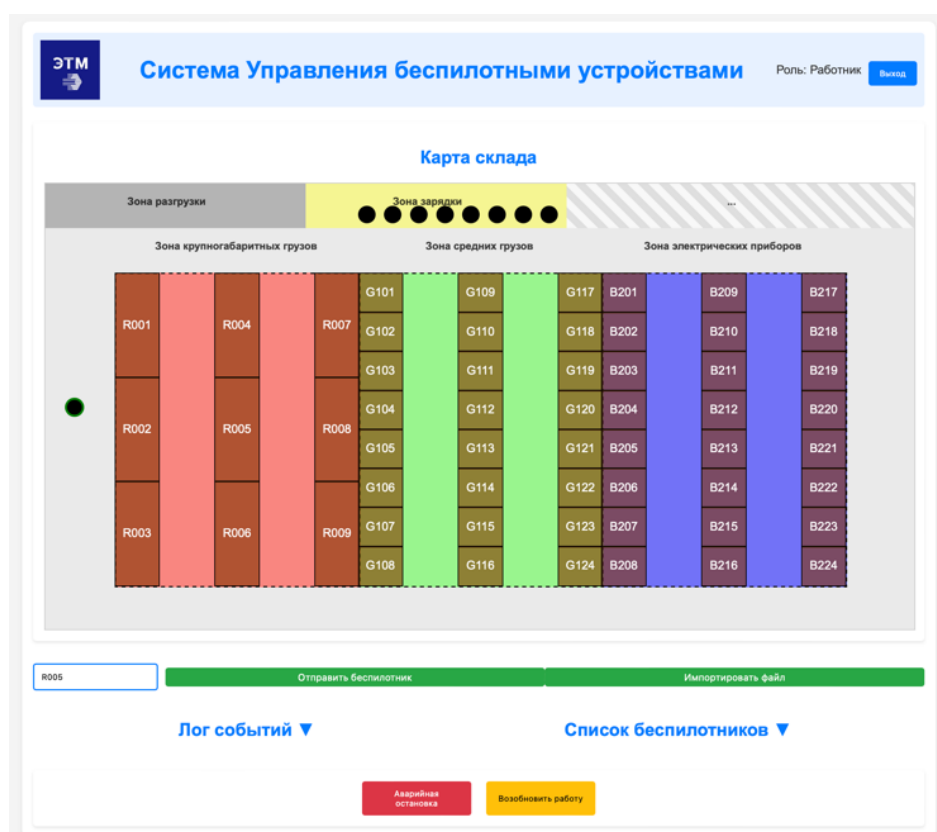


Рисунок 20 – груз взят (изменение контура беспилотника)

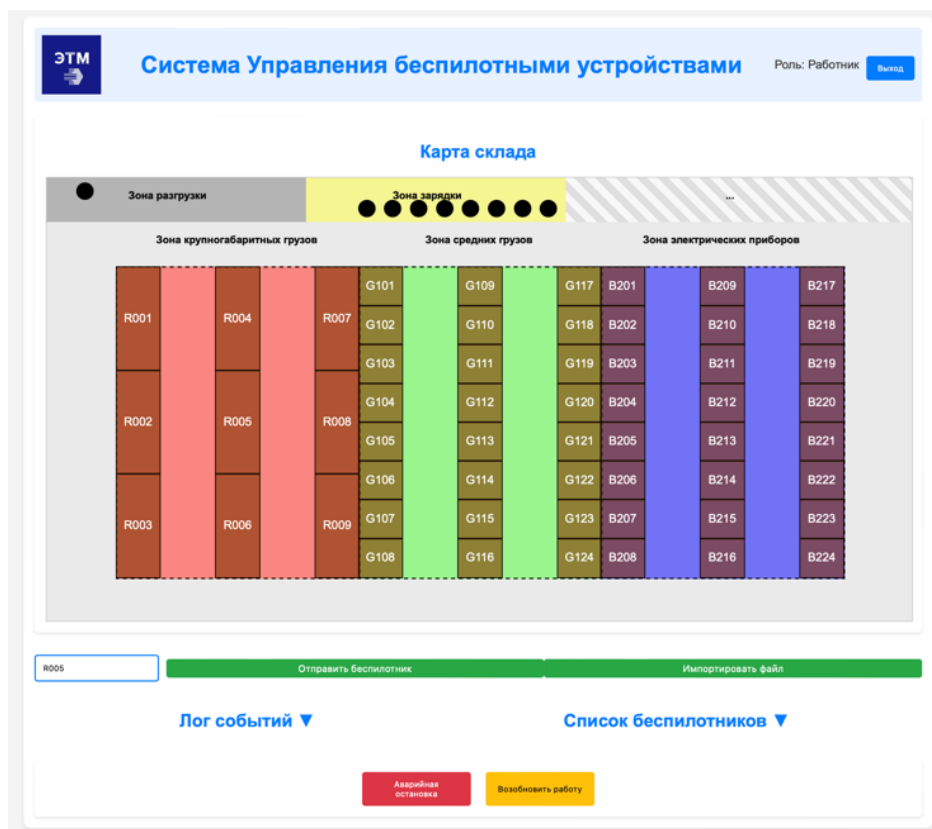


Рисунок 21 – груз перенесен в зону разгрузки (изменение контура беспилотника)

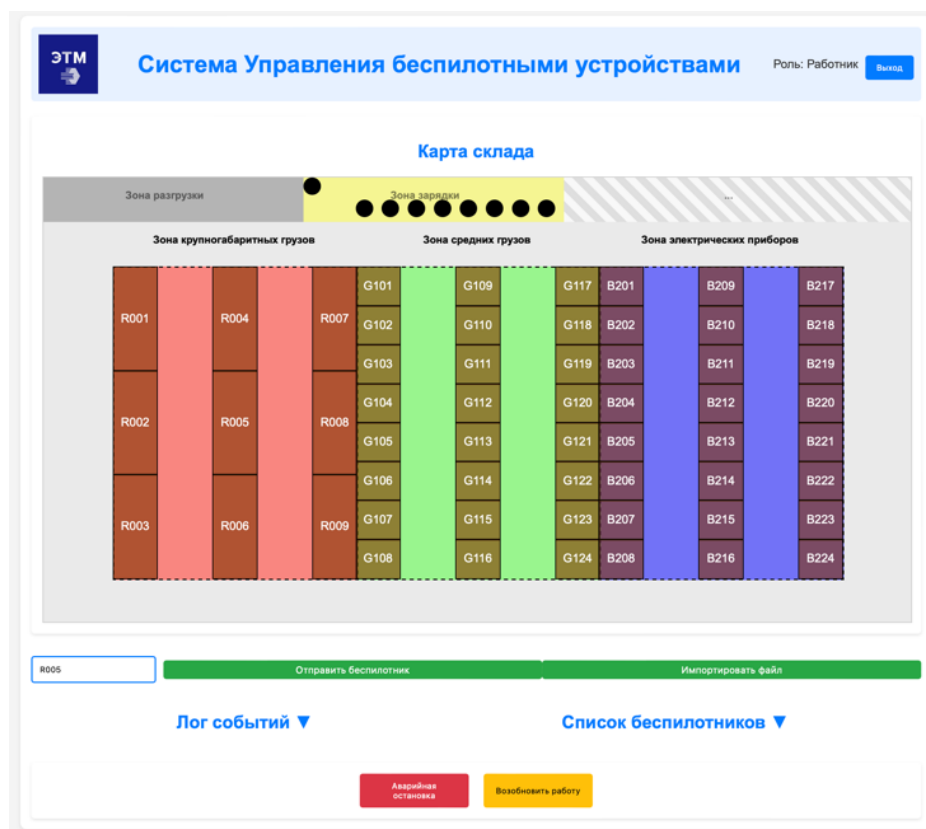


Рисунок 22 – беспилотник вернулся на свое место зарядки

### 3.2.4 Ручное управление беспилотником

Этот функционал предоставляет пользователю возможность прямого контроля за беспилотным погрузчиком, что особенно полезно в ситуациях, требующих точного и мгновенного реагирования. Процесс проиллюстрирован на рисунках 23-25.

Алгоритм работы:

1. Выбор беспилотника: В списке беспилотников пользователь выбирает нужный погрузчик. При выборе, активный беспилотник подсвечивается оранжевым, что позволяет легко определить текущий выбор.

2. Управление с помощью клавиатуры: после выбора погрузчика пользователь может прямо на интерфейсе программы управлять им с помощью стрелок на клавиатуре. Вверх, вниз, влево и вправо - каждое движение точно отражается на поведении беспилотника. Для завершения управления следует нажать «Снять выбор».

3. Ограничения движения: важно отметить, что при ручном управлении каждый беспилотник имеет свои ограничения по перемещению. Это обеспечивает безопасность и предотвращает попадание в запрещенные зоны. Каждому погрузчику присвоена своя зона и диапазон разрешенного движения:

- Погрузчики 1, 2, 3 могут находиться в красной зоне и проездах.
- Погрузчики 4, 5, 6 могут находиться в зеленой зоне и проездах.
- Погрузчики 7, 8, 9 могут находиться в синей зоне и проездах.

Преимущества ручного управления:

- Гибкость: Пользователь имеет полный контроль над движением беспилотника, что позволяет реагировать на нестандартные ситуации и решать задачи в реальном времени.

– Точность: Ручное управление обеспечивает максимально точное позиционирование погрузчика, что особенно важно при выполнении задач, требующих высокой точности.

– Удобство: интуитивно понятный интерфейс и простые команды управления делают этот процесс максимально удобным и доступным для пользователей.

Этот функционал дополняет автоматизированные процессы управления беспилотными погрузчиками, обеспечивая более гибкий и адаптивный подход к выполнению задач.

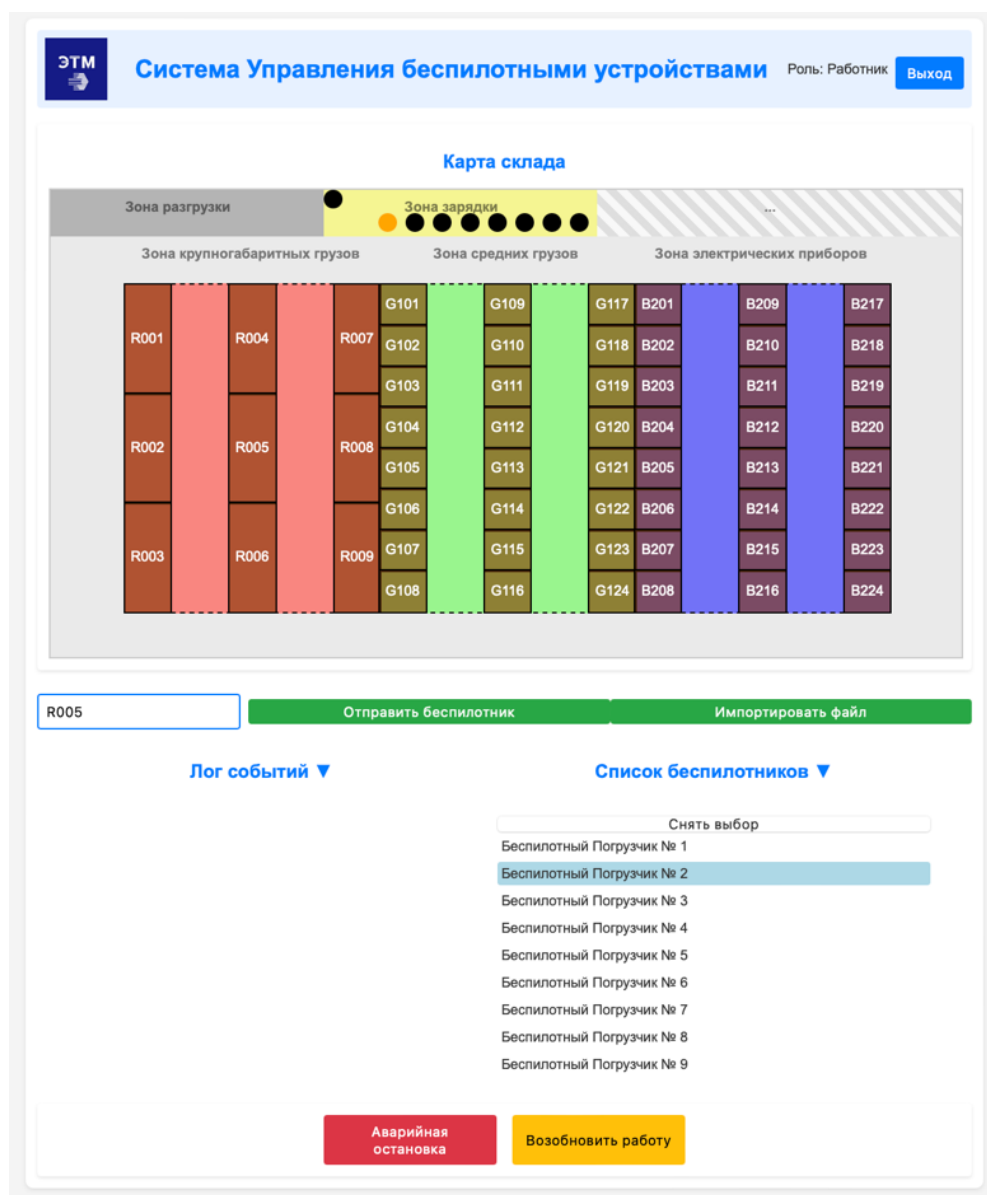


Рисунок 23 – выбор беспилотника

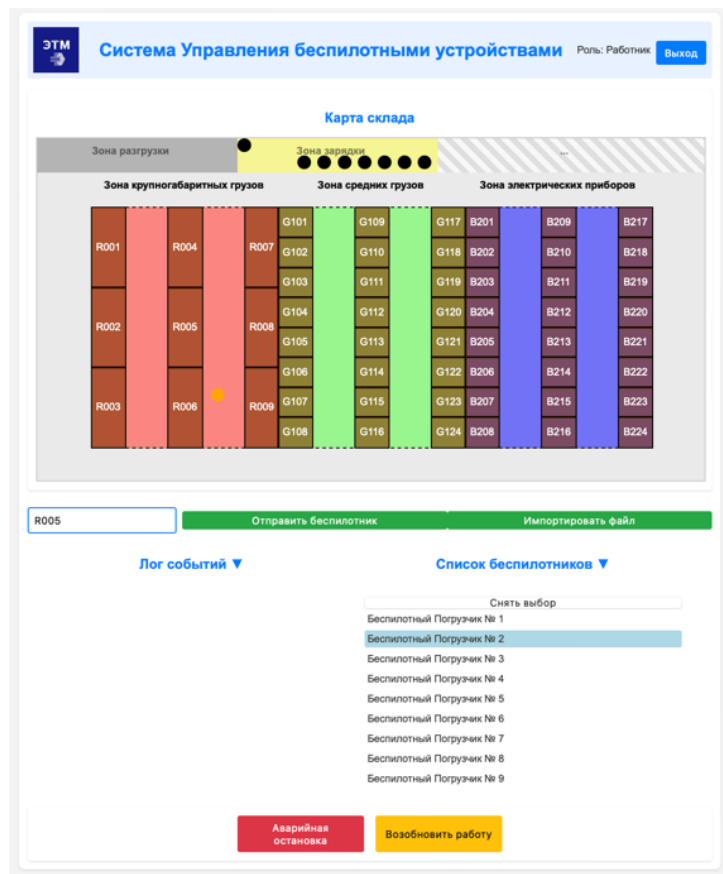


Рисунок 24 – ручное перемещение беспилотника

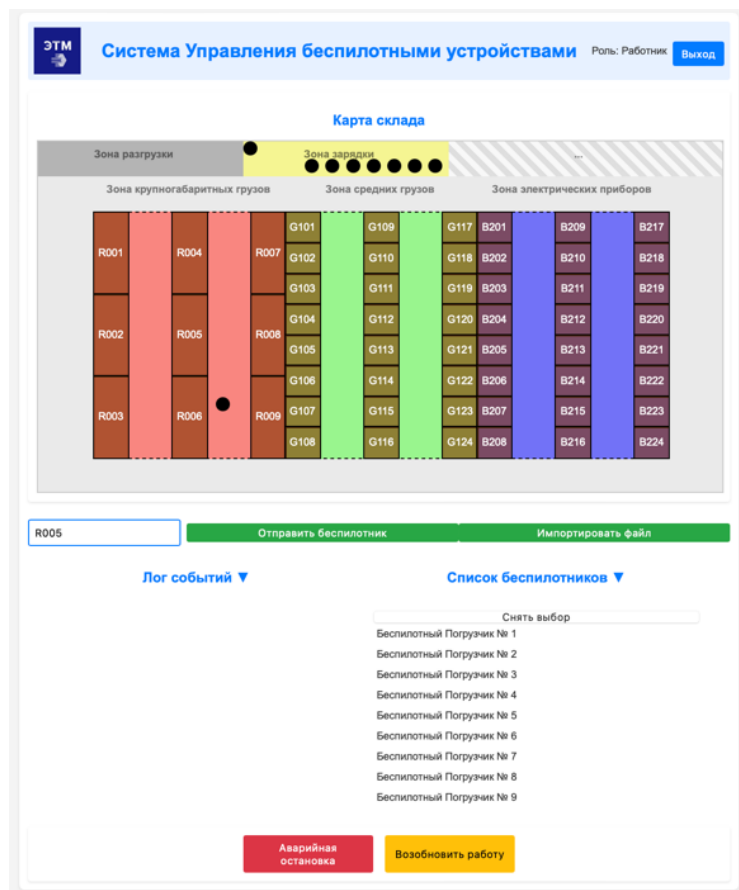


Рисунок 25 – беспилотник перемещен, выбор снят

### **3.2.5 Автоматизация работы склада: импорт файла с задачами**

Этот ключевой функционал позволяет максимально упростить и ускорить процесс управления складом, автоматизируя назначение задач для беспилотных погрузчиков. Процесс проиллюстрирован на рисунках 26-28.

Алгоритм работы:

1. Импорт файла с задачами: для начала процесса автоматизации необходимо нажать на кнопку "Импорт файла" на интерфейсе программы. Это запускает процесс импорта задач из внешнего источника.
2. Выбор текстового документа: после нажатия кнопки пользователю предлагается выбрать текстовый документ, в котором задачи представлены в виде списка названий стеллажей, к которым нужно подъехать беспилотным погрузчиком.
3. Автоматическое назначение задач: после успешного импорта файла система начинает автоматически назначать задачи беспилотным погрузчикам в соответствии с указанными в файле стеллажами. Каждый погрузчик получает свою задачу и автоматически отправляется выполнять ее.

Преимущества автоматизации с помощью импорта файла:

- Эффективность: Автоматизированный процесс импорта и назначения задач позволяет существенно сократить время, затрачиваемое на планирование и распределение задач.
- Точность: Использование файлов для передачи информации о задачах исключает вероятность ошибок при их вводе вручную.
- Гибкость: Пользователи могут легко адаптировать процесс импорта и назначения задач под изменяющиеся условия работы склада.

Этот функционал значительно упрощает процесс управления складом, обеспечивая более эффективное и гибкое распределение задач между беспилотными погрузчиками.

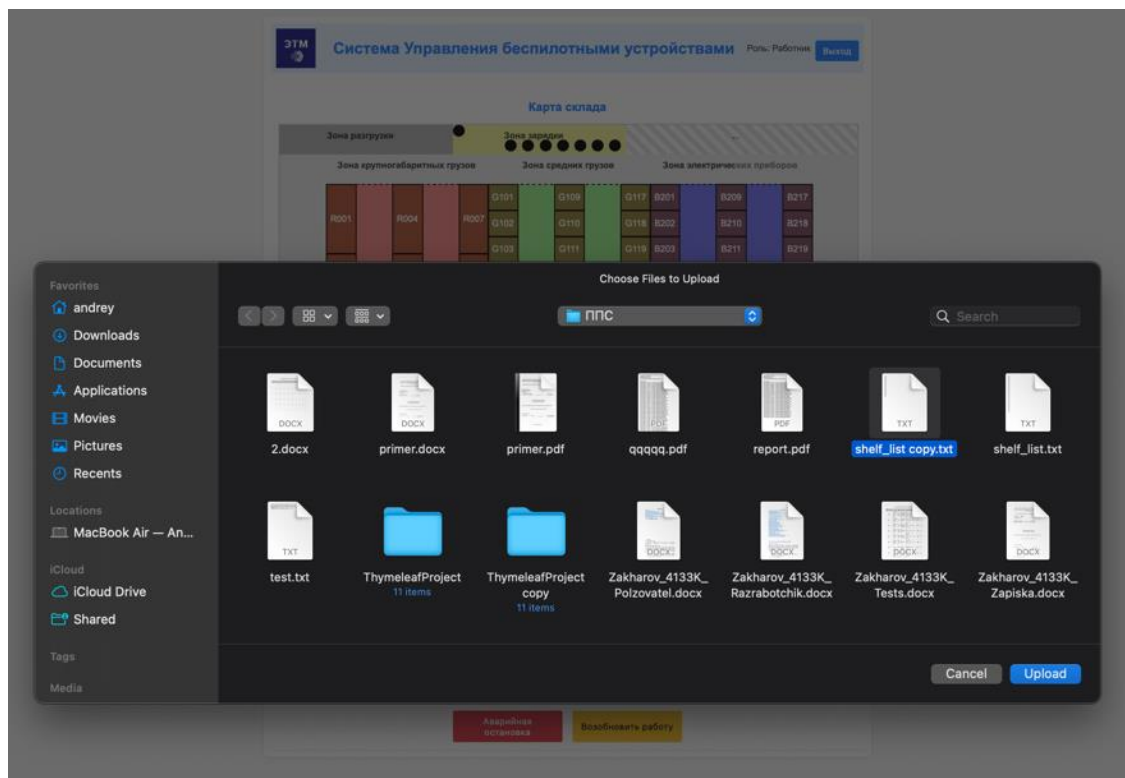


Рисунок 26 – при нажатии на Импорт файла открывается проводник, необходимо выбрать нужный файл и нажать Upload/Выгрузить



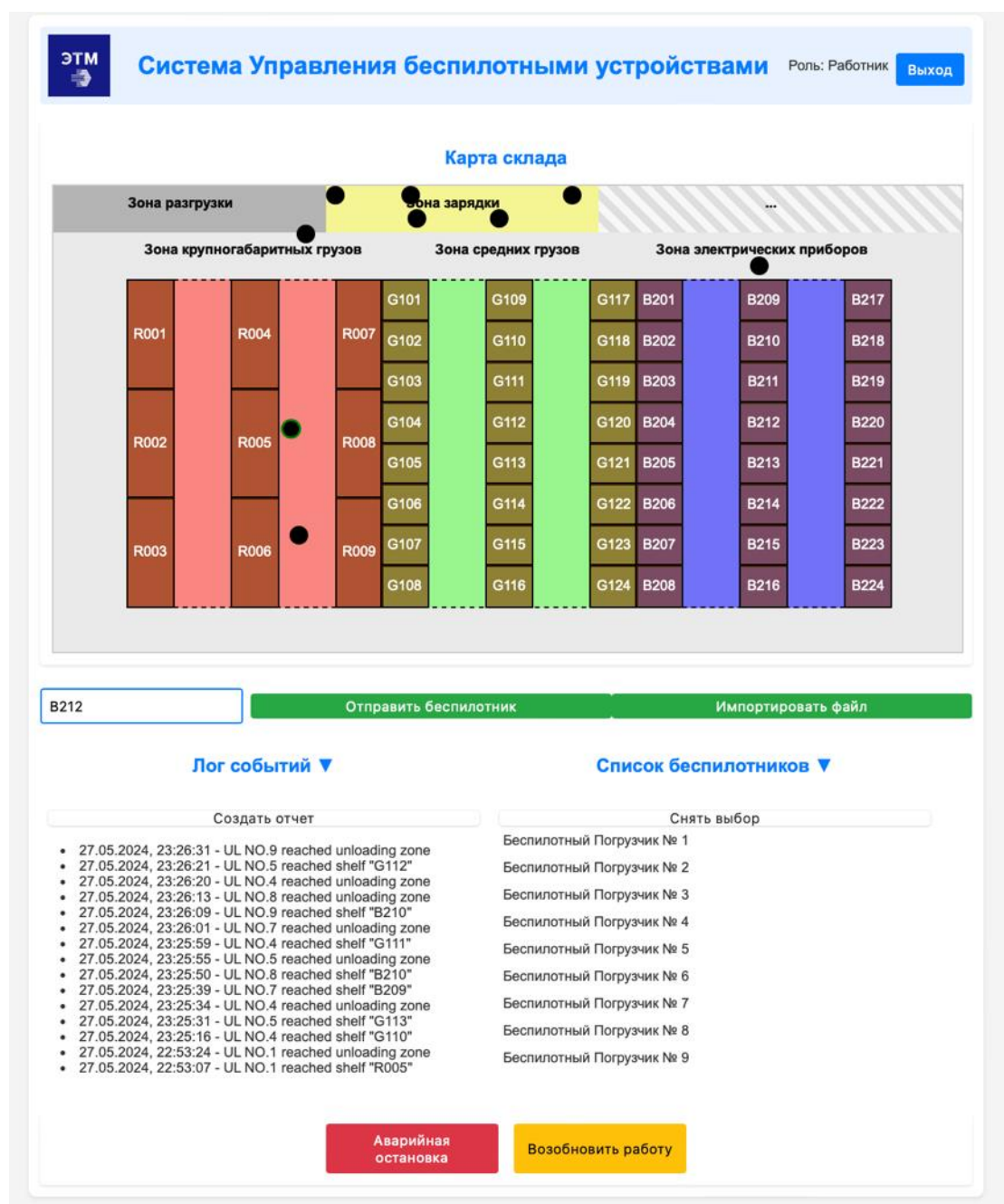
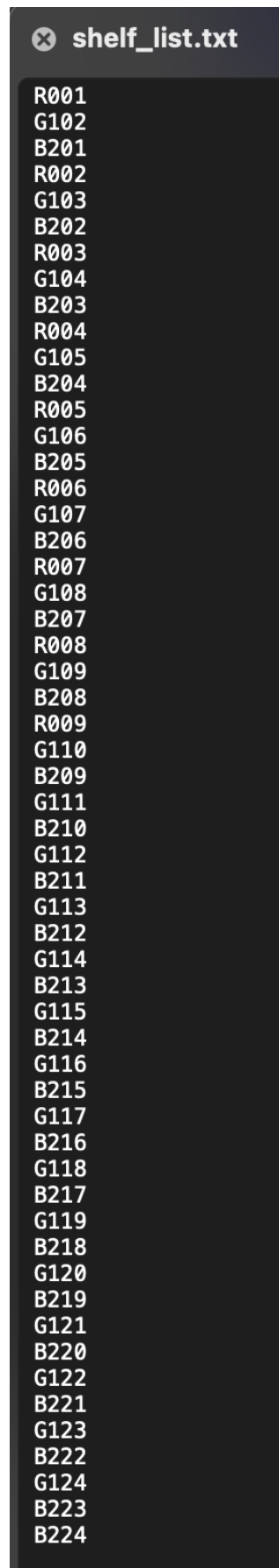


Рисунок 27 – беспилотники автоматически заезжают за грузом в нужных стеллажах

A screenshot of a text editor window titled 'shelf\_list.txt'. The window contains a list of codes arranged in pairs, alternating between 'R' and 'G' prefixes followed by a three-digit number. The list starts with R001 and G102, and ends with B224. The text is white on a dark background.

```
shelf_list.txt
R001
G102
B201
R002
G103
B202
R003
G104
B203
R004
G105
B204
R005
G106
B205
R006
G107
B206
R007
G108
B207
R008
G109
B208
R009
G110
B209
G111
B210
G112
B211
G113
B212
G114
B213
G115
B214
G116
B215
G117
B216
G118
B217
G119
B218
G120
B219
G121
B220
G122
B221
G123
B222
G124
B223
B224
```

Рисунок 28 – пример файла для импорта

### **3.2.6 Аварийная остановка работы системы**

Этот важный функционал предназначен для немедленной остановки всех беспилотных погрузчиков в случае возникновения чрезвычайной ситуации или необходимости прекращения работы. Процесс проиллюстрирован на рисунке 29.

Алгоритм работы:

1. Аварийная остановка: при необходимости прекратить работу всех беспилотных погрузчиков, пользователь нажимает на кнопку "Аварийной остановки" на интерфейсе программы. Это активирует процесс немедленной остановки всех погрузчиков.

2. Остановка движения: после нажатия на кнопку все беспилотники моментально останавливаются и начинают мигать красным по контуру, что явно сигнализирует о том, что система находится в режиме аварийной остановки.

3. Возможность ручного перемещения: В этом режиме каждый беспилотный погрузчик можно перемещать вручную, если это необходимо для безопасного восстановления работы системы или выполнения других операций.

Преимущества аварийной остановки:

– Безопасность: Возможность мгновенной остановки всех беспилотных погрузчиков в случае чрезвычайной ситуации способствует обеспечению безопасности персонала и оборудования.

– Контроль: Аварийная остановка предоставляет оператору полный контроль над ситуацией, позволяя быстро принять меры по устранению проблемы или обеспечению безопасности.

– Гибкость: Возможность ручного перемещения беспилотных погрузчиков в режиме аварийной остановки позволяет оператору легко и эффективно управлять процессом восстановления работы системы.

Этот функционал играет ключевую роль в обеспечении безопасности и контроля работы системы управления беспилотными погрузчиками, обеспечивая оперативное реагирование на чрезвычайные ситуации.

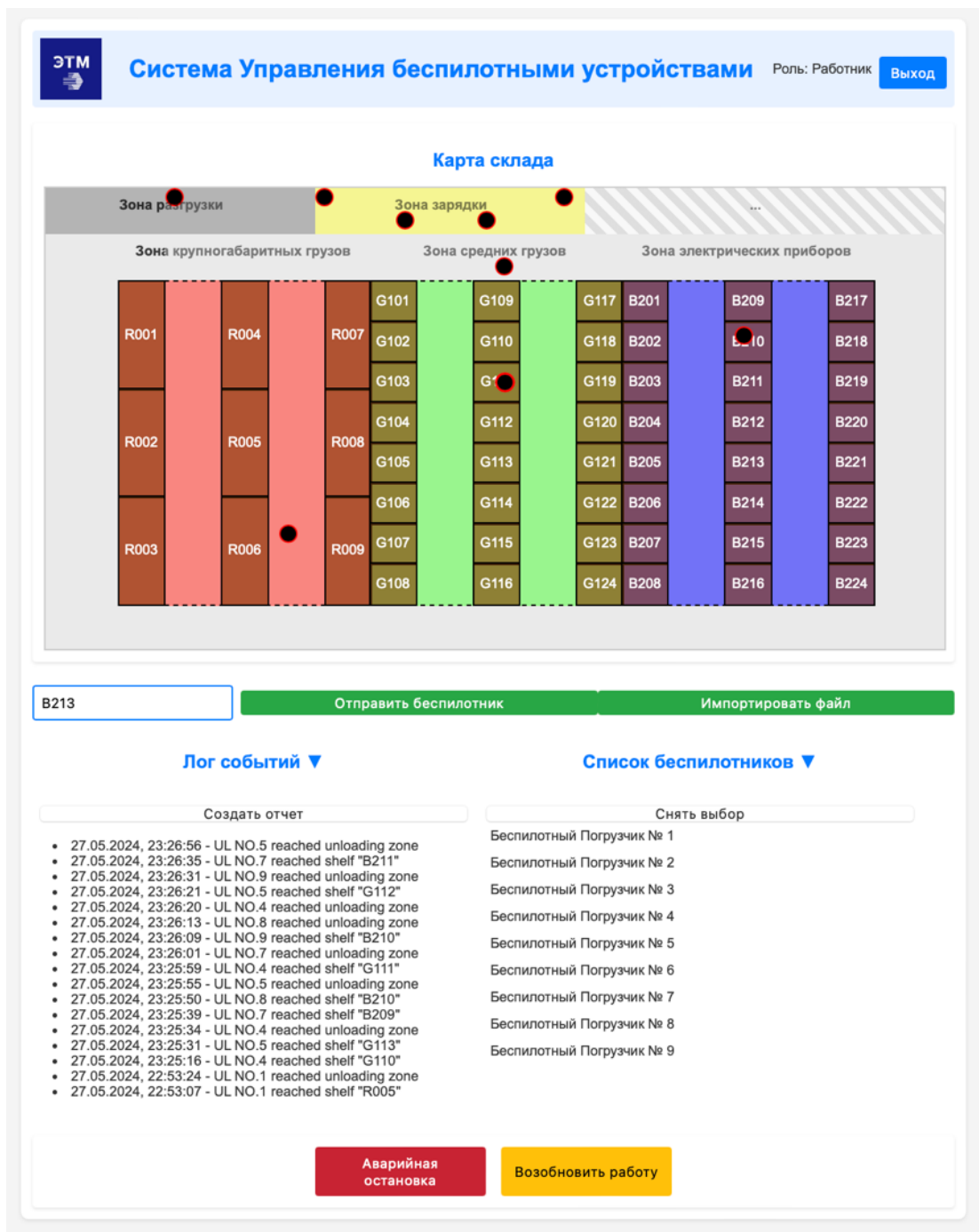


Рисунок 29 – активирован режим аварийной остановки

### 3.2.7 Возобновление работы системы

Этот этап важен для восстановления нормальной работы системы после ее аварийной остановки. Процесс проиллюстрирован на рисунке 30.

Алгоритм работы:

1. Возобновление работы: для начала процесса восстановления работы системы необходимо нажать на кнопку "Возобновление работы" на интерфейсе программы. Это активирует процесс возобновления всех операций беспилотных погрузчиков.

2. Продолжение работы: после нажатия кнопки все беспилотные погрузчики автоматически возобновляют свою работу. Они продолжают движение к заданным точкам и выполняют назначенные им задачи.

3. Нормализация работы: после процесса возобновления работы системы все операции и функционал возвращаются к нормальному режиму. Система готова к продолжению работы без прерываний.

Преимущества возобновления работы системы:

- Эффективность: Процесс возобновления работы позволяет минимизировать простой оборудования и восстановить работоспособность системы в кратчайшие сроки.
- Континуитет: Возобновление работы не приводит к потере данных или прерыванию операций. Все беспилотные погрузчики продолжают свою работу с того момента, на котором они были остановлены.
- Простота: Процесс возобновления работы осуществляется одним нажатием кнопки, что делает его легким и быстрым.

Этот функционал обеспечивает безопасное и эффективное возобновление работы системы после аварийной остановки, минимизируя простои и обеспечивая непрерывность операций на складе.

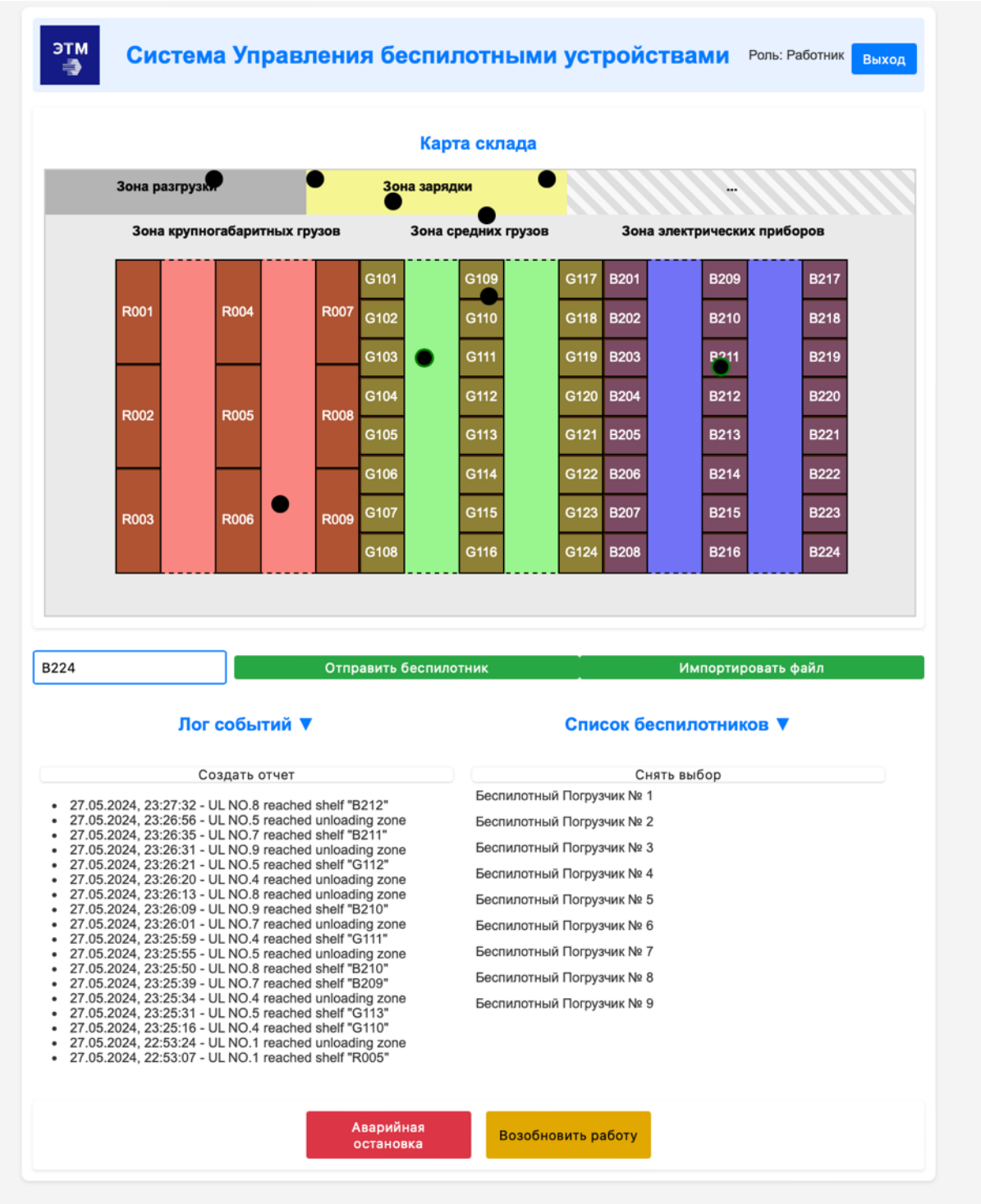


Рисунок 30 – все беспилотники успешно возобновили работу

### 3.2.8 Создание отчёта

Этот этап важен для документирования работы системы и анализа ее производительности. Процесс проиллюстрирован на рисунках 26-28.

Алгоритм работы:

1. Проверка наличия записей в логах: для создания отчета необходимо, чтобы в логах системы была хотя бы одна запись. Если логи пусты, отчет не будет создан.
2. Инициирование создания отчета: Пользователь нажимает на кнопку "Создать отчет" на интерфейсе программы. Это запускает процесс создания отчета.
3. Сохранение отчета: после нажатия кнопки отчет сохраняется на компьютер.
4. Содержание отчета: В отчете содержится следующая информация:
  - Промежуток времени, за который был создан отчет.
  - Количество работающих беспилотных погрузчиков в указанный промежуток времени.
  - Количество завершенных задач за тот же промежуток времени.
  - Логи, содержащие детальную информацию о действиях и событиях, произошедших в системе за указанный промежуток времени.

Преимущества создания отчета:

- Документация работы: Создание отчета позволяет вести детальную документацию о работе системы, что полезно для анализа и оптимизации процессов.

– Анализ производительности: Информация, содержащаяся в отчете, помогает анализировать производительность системы, выявлять проблемные места и принимать меры по их улучшению.

– Отслеживание действий: Логи в отчете позволяют отслеживать все действия и события, происходящие в системе, что важно для контроля за ее работой и решения возникающих проблем.

Этот функционал обеспечивает полную прозрачность работы системы управления беспилотными погрузчиками и обеспечивает возможность документирования и анализа ее работы.

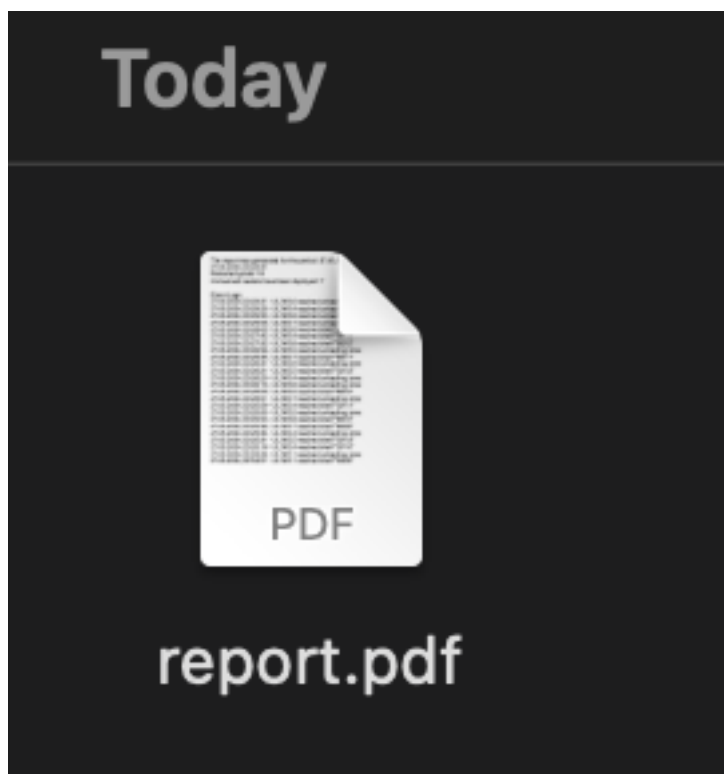


Рисунок 31 – при сохранении отчета скачивается такой отчет



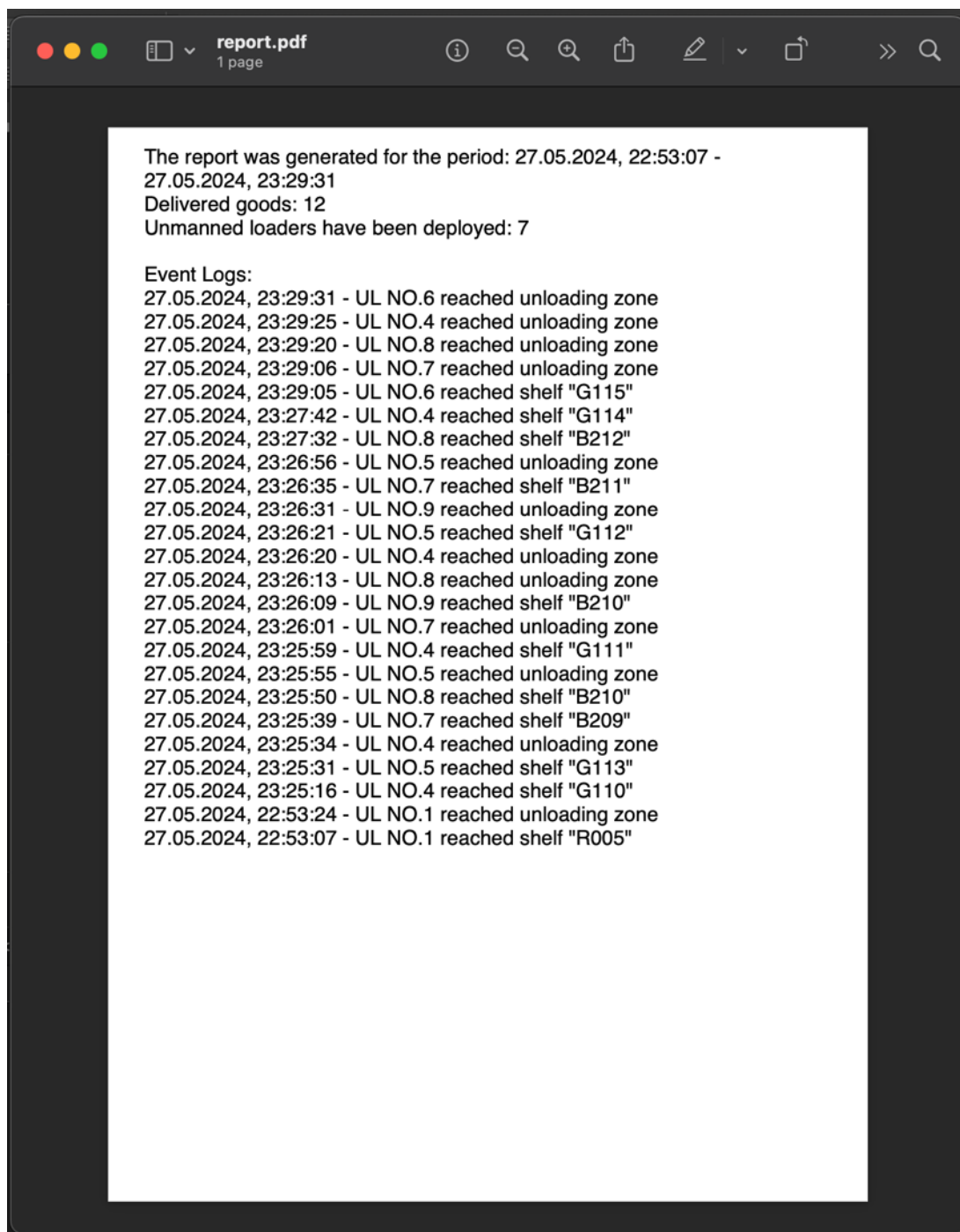


Рисунок 32 – содержание скачанного отчета

## Лог событий ▼

Создать отчет

- 27.05.2024, 23:29:31 - UL NO.6 reached unloading zone
- 27.05.2024, 23:29:25 - UL NO.4 reached unloading zone
- 27.05.2024, 23:29:20 - UL NO.8 reached unloading zone
- 27.05.2024, 23:29:06 - UL NO.7 reached unloading zone
- 27.05.2024, 23:29:05 - UL NO.6 reached shelf "G115"
- 27.05.2024, 23:27:42 - UL NO.4 reached shelf "G114"
- 27.05.2024, 23:27:32 - UL NO.8 reached shelf "B212"
- 27.05.2024, 23:26:56 - UL NO.5 reached unloading zone
- 27.05.2024, 23:26:35 - UL NO.7 reached shelf "B211"
- 27.05.2024, 23:26:31 - UL NO.9 reached unloading zone
- 27.05.2024, 23:26:21 - UL NO.5 reached shelf "G112"
- 27.05.2024, 23:26:20 - UL NO.4 reached unloading zone
- 27.05.2024, 23:26:13 - UL NO.8 reached unloading zone
- 27.05.2024, 23:26:09 - UL NO.9 reached shelf "B210"
- 27.05.2024, 23:26:01 - UL NO.7 reached unloading zone
- 27.05.2024, 23:25:59 - UL NO.4 reached shelf "G111"
- 27.05.2024, 23:25:55 - UL NO.5 reached unloading zone
- 27.05.2024, 23:25:50 - UL NO.8 reached shelf "B210"
- 27.05.2024, 23:25:39 - UL NO.7 reached shelf "B209"
- 27.05.2024, 23:25:34 - UL NO.4 reached unloading zone
- 27.05.2024, 23:25:31 - UL NO.5 reached shelf "G113"
- 27.05.2024, 23:25:16 - UL NO.4 reached shelf "G110"
- 27.05.2024, 22:53:24 - UL NO.1 reached unloading zone
- 27.05.2024, 22:53:07 - UL NO.1 reached shelf "R005"

Рисунок 33 – содержание отчета совпадает с текущими логами – всё верно отрабатывает

## 4. РАЗРАБОТКА АВТОМАТИЗИРОВАННОЙ СИСТЕМЫ

### 4.1 Серверная часть системы

Серверная часть приложения представляет собой основной компонент, ответственный за обработку запросов от клиентов и управление данными, необходимыми для функционирования системы управления беспилотными погрузчиками.

#### Общее описание

Класс `ThymeleafProjectApplication` является точкой входа в приложение. Он использует аннотацию `@SpringBootApplication`, что делает его запускаемым классом и обеспечивает автоматическую конфигурацию Spring Boot.

#### Основные функции и возможности

- Управление дронами: Класс `DroneController` обрабатывает запросы, связанные с управлением дронами на складе. Он содержит методы для получения списка всех дронов, добавления нового дрона и удаления существующего.
- Аутентификация и авторизация: Класс `WebSecurityConfig` конфигурирует безопасность приложения, определяя правила доступа для различных URL-адресов и настройки аутентификации пользователей.

#### Интеграция с другими компонентами

- Использование Thymeleaf: для создания динамических HTML-страниц используется шаблон Thymeleaf. Это обеспечивает удобное и гибкое отображение данных на стороне сервера.
- Использование Spring Security: для обеспечения безопасности приложения и защиты от несанкционированного доступа используется Spring Security. Это позволяет контролировать доступ к различным URL-адресам и выполнение различных операций в приложении.

#### Дополнительные компоненты

- Классы моделей: В приложении используется класс DroneObj, представляющий модель дрона со свойствами, такими как идентификатор, статус и текущая задача.

- Контроллеры и сервисы: Классы DroneController и LoginController являются контроллерами, обрабатывающими HTTP-запросы от клиентов. Они взаимодействуют с соответствующими сервисами для выполнения бизнес-логики приложения.

Этот функционал обеспечивает основу работы серверной части приложения и обеспечивает его функциональность и безопасность.

#### **4.1.1 Описание класса DroneController**

##### Назначение

DroneController является контроллером для управления дронами в системе. Он обрабатывает запросы от клиентов, связанные с дронами, и выполняет соответствующие операции.

##### Аннотации

- @Controller: Обозначает класс как компонент Spring, управляющий запросами на веб-страницы.
- @RequestMapping("/drones"): Указывает базовый путь для всех методов контроллера.

##### Поля класса

- private List<DroneObj> drones: Список объектов типа DroneObj, представляющий дроны в системе. Инициализируется пустым списком.

##### Методы

1. getAllDrones(Model model): Обработывает GET-запросы для получения списка всех дронов.

- Параметры:
  - `Model model`: Модель, используемая для передачи данных в представление.
  - Действия:
  - Добавляет список дронов в модель.
  - Возвращает имя представления "drones".
2. `addDrone(@ModelAttribute DroneObj drone)`: Обработывает POST-запросы для добавления нового дрона.
- Параметры:
  - `@ModelAttribute DroneObj drone`: Данные нового дрона, переданные из формы.
  - Действия:
  - Добавляет новый дрон в список дронов.
  - Перенаправляет на страницу со списком дронов.
3. `deleteDrone(@PathVariable int id)`: Обработывает POST-запросы для удаления дрона по его идентификатору.
- Параметры:
  - `@PathVariable int id`: Идентификатор удаляемого дрона, переданный в URL.
  - Действия:
  - Удаляет дрон с указанным идентификатором из списка дронов.
  - Перенаправляет на страницу со списком дронов.

Зависимости

- `import org.springframework.beans.factory.annotation.Autowired;` Для автоматического внедрения зависимостей.
- `import org.springframework.security.core.annotation.AuthenticationPrincipal;` Для работы с аутентификацией текущего пользователя.
- `import org.springframework.stereotype.Controller;` Для обозначения класса как контроллера.
- `import org.springframework.ui.Model;` Для работы с моделью данных в Spring MVC.
- `import org.springframework.web.bind.annotation.*;` Для использования аннотаций контроллера.
- `import java.util.ArrayList;` Для работы с коллекцией дронов.
- `import java.util.List;` Для работы с коллекциями дронов.
- `import com.example.ThymeleafProject.DroneObj;` Для использования класса `DroneObj`.

Этот класс играет ключевую роль в обработке запросов, связанных с дронами, и управлении данными о них в системе.

#### **4.1.2 Описание класса `DroneObj`**

##### **Назначение**

`DroneObj` представляет собой модель дрона в системе. Этот класс содержит информацию о каждом дроне, включая его идентификатор, текущий статус и выполняемую задачу.

##### **Поля класса**

- `private int droneID;` Идентификатор дрона.
- `private String status;` Текущий статус дрона.

- `private String currentTask`: Текущая задача, выполняемая дроном.

#### Конструктор

- `public DroneObj(int droneID, String status, String currentTask)`:

Конструктор класса, который принимает и инициализирует значения идентификатора дрона, статуса и текущей задачи.

#### Методы доступа (геттеры и сеттеры)

1. `public int getDroneID()`: Возвращает идентификатор дрона.
2. `public void setDroneID(int droneID)`: Устанавливает новое значение идентификатора дрона.
3. `public String getStatus()`: Возвращает текущий статус дрона.
4. `public void setStatus(String status)`: Устанавливает новый статус дрона.
5. `public String getCurrentTask()`: Возвращает текущую задачу дрона.
6. `public void setCurrentTask(String currentTask)`: Устанавливает новую текущую задачу для дрона.

#### Методы

- `@Override public String toString()`: Переопределенный метод `toString()`, который возвращает строковое представление объекта `DroneObj`, включая идентификатор, статус и текущую задачу.

#### Зависимости

Отсутствуют.

Этот класс предоставляет удобный способ хранения информации о дронах в системе и управления их данными. Он используется в контроллере и других компонентах приложения для обработки и представления информации о дронах.

### 4.1.3 Описание класса LoginController

#### Назначение

LoginController отвечает за обработку запросов, связанных с аутентификацией пользователей в системе и их выходом из системы.

#### Методы

1. `public String login()`: Обработывает GET-запросы на эндпоинт `/login`. Возвращает строку `"login"`, что указывает на представление, отображающее страницу входа в систему.

2. `public String logout(HttpServletRequest request) throws ServletException`: Обработывает POST-запросы на эндпоинт `/logout`. Выполняет выход пользователя из системы, вызывая метод `logout()` для объекта `HttpServletRequest`, и перенаправляет пользователя на страницу входа в систему.

#### Аннотации

– `@Controller`: Обозначает класс как контроллер Spring MVC, который обрабатывает HTTP-запросы.

– `@GetMapping("/login")`: Указывает, что метод `login()` обрабатывает HTTP GET-запросы на адресе `/login`.

– `@PostMapping("/logout")`: Указывает, что метод `logout()` обрабатывает HTTP POST-запросы на адресе `/logout`.

#### Параметры методов

– `HttpServletRequest request`: Параметр метода `logout()`, представляющий HTTP-запрос, который будет использоваться для выполнения выхода пользователя из системы.



## Исключения

- `ServletException`: Выбрасывается методом `logout()` при возникновении ошибки при выполнении выхода пользователя из системы.

## Зависимости

- Нет дополнительных зависимостей, кроме стандартных библиотек и фреймворка Spring.

Этот класс обеспечивает обработку запросов для аутентификации и выхода пользователей из системы, обеспечивая взаимодействие с веб-интерфейсом.

### 4.1.4 Описание класса `ThymeleafProjectApplication`

#### Назначение

Класс `ThymeleafProjectApplication` представляет точку входа в приложение Spring Boot. Он содержит конфигурации, бины и настройки, необходимые для запуска и работы приложения.

#### Методы

1. `public static void main(String[] args)`: Метод `main` является точкой входа в приложение. Он запускает Spring Boot-приложение.

#### Аннотации

- `@SpringBootApplication`: Аннотация, объединяющая `@Configuration`, `@EnableAutoConfiguration` и `@ComponentScan`. Она указывает на то, что данный класс является конфигурационным, автоматически настраивает приложение и сканирует компоненты в пакете, где находится класс `ThymeleafProjectApplication` и его подпакетах.

- `@Bean`: Аннотация, которая указывает на то, что метод создает и возвращает новый экземпляр объекта, который будет управлять Spring.

– `@Configuration`: Аннотация, которая указывает на то, что класс содержит один или более методов, помеченных аннотацией `@Bean`, которые могут быть использованы для настройки и настройки Spring IoC контейнера.

– `@Override`: Переопределение метода родительского класса.

#### Методы и их назначение

1. `public MessageSource messageSource()`: Метод создает и конфигурирует `MessageSource`, который используется для загрузки сообщений из файлов свойств.

2. `public LocaleResolver localeResolver()`: Метод создает и настраивает `LocaleResolver`, который используется для определения текущей локали приложения.

3. `public ServletWebServerFactory servletContainer()`: Метод создает и настраивает фабрику `ServletWebServerFactory`, которая используется для настройки контейнера сервлетов Tomcat.

4. `@Bean public List<DroneObj> drones()`: Метод создает и возвращает новый список объектов `DroneObj`, который используется в приложении для хранения информации о беспилотниках.

#### Вложенный класс AppConfig

1. `@Bean public List<DroneObj> drones()`: Метод создает и возвращает новый список объектов `DroneObj`. Этот класс создан внутри `ThymeleafProjectApplication` для управления бинами и настроек в приложении.

#### Зависимости

– `org.springframework.boot.SpringApplication`: Класс, предоставляющий удобные методы для запуска Spring Boot-приложения.

- `org.springframework.boot.autoconfigure.SpringBootApplication`:  
Аннотация, объединяющая несколько других аннотаций для настройки Spring Boot-приложения.
- `org.springframework.context.MessageSource`: Интерфейс, представляющий источник сообщений для интернационализации.
- `org.springframework.web.servlet.LocaleResolver`: Интерфейс, представляющий стратегию для определения текущей локали в приложении.
- `org.springframework.boot.web.embedded.tomcat.TomcatServletWebServerFactory`: Фабрика, используемая для настройки встроенного сервера Tomcat в Spring Boot.
- `jakarta.servlet.ServletException`: Исключение, возникающее при сервлетных ошибках и проблемах.
- `org.apache.catalina.Context`: Контекст, представляющий веб-приложение в сервере Tomcat.
- `org.apache.tomcat.util.descriptor.web.FilterDef`: Определение фильтра для конфигурации сервера Tomcat.
- `org.apache.tomcat.util.descriptor.web.FilterMap`: Отображение URL-адресов на фильтры для конфигурации сервера Tomcat.
- `org.springframework.web.filter.CharacterEncodingFilter`: Фильтр, используемый для установки кодировки символов для HTTP-запросов и ответов.
- `java.util.List`: Интерфейс, представляющий последовательность элементов определенного типа в списке.

Этот класс содержит основные настройки и конфигурации для запуска приложения Spring Boot, включая настройку сервера, обработку сообщений и определение локали приложения.

## 4.1.5 Описание класса WebSecurityConfig

### Назначение

Класс WebSecurityConfig отвечает за настройку безопасности веб-приложения с использованием Spring Security. Он определяет конфигурацию прав доступа, шифрование паролей, аутентификацию пользователей и управление сессиями входа и выхода.

### Методы и аннотации

1. @Configuration: Аннотация, указывающая, что класс является конфигурационным компонентом Spring.
2. @EnableWebSecurity: Аннотация, которая включает поддержку безопасности веб-приложения на основе Spring Security.
3. @Bean: Аннотация, позволяющая определить метод как бин Spring.

### Методы и их назначение

1. `public SecurityFilterChain filterChain(HttpSecurity http) throws Exception`: Метод, который настраивает цепочку фильтров безопасности для обработки HTTP-запросов. В данном методе определяются правила доступа к различным URL-адресам, аутентификация пользователей и настройка страниц входа и выхода из системы.
2. `public PasswordEncoder passwordEncoder()`: Метод, который создает и возвращает объект для шифрования паролей пользователей. В данном случае используется BCryptPasswordEncoder.
3. `public InMemoryUserDetailsManager userDetailsService()`: Метод, который создает и возвращает менеджер пользователей в памяти. В этом методе создаются пользователи с их именами, паролями и ролями. Для каждого пользователя пароль шифруется с помощью PasswordEncoder.

### Зависимости

- `org.springframework.context.annotation.Bean`: Аннотация, указывающая на то, что метод является фабрикой бина Spring.
- `org.springframework.context.annotation.Configuration`: Аннотация, указывающая, что класс является конфигурационным компонентом Spring.
- `org.springframework.security.config.annotation.web.builders.HttpSecurity`: Класс, предоставляющий API для настройки защиты на основе HTTP-запросов.
- `org.springframework.security.config.annotation.web.configuration.EnableWebSecurity`: Аннотация, которая включает поддержку безопасности веб-приложения на основе Spring Security.
- `org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder`: Класс, используемый для шифрования паролей с использованием алгоритма BCrypt.
- `org.springframework.security.core.userdetails.UserDetails`: Интерфейс, представляющий информацию о пользователе, необходимую для аутентификации и авторизации.
- `org.springframework.security.provisioning.InMemoryUserDetailsManager`: Класс, предоставляющий реализацию `UserDetailsService` для управления пользователями в памяти.
- `org.springframework.security.core.userdetails.User`: Класс, предоставляющий реализацию интерфейса `UserDetails` для представления информации о пользователе.

Этот класс предоставляет настройки безопасности для веб-приложения, включая правила доступа, шифрование паролей и управление пользователями в памяти.

## 4.2 Клиентская часть системы

Клиентская часть приложения представляет собой интерфейс, с которым взаимодействуют пользователи для управления беспилотными погрузчиками и отслеживания их действий на складе. Включает в себя HTML-страницы, стили CSS и JavaScript для обеспечения интерактивности и динамичного отображения данных.

### Общее описание

Клиентская часть приложения предоставляет удобный пользовательский интерфейс для работы с системой управления беспилотными погрузчиками. Включает в себя различные разделы для отображения карты склада, списка дронов, лога событий и управления аварийными ситуациями.

### HTML-страницы

- index.html: Главная страница, на которой отображается карта склада, список дронов, лог событий и элементы управления аварийными ситуациями.
- login.html: Страница входа в систему управления беспилотными погрузчиками. Предоставляет форму для ввода имени пользователя и пароля.

### Стили CSS

- style.css: Содержит основные стили для элементов интерфейса, такие как шрифты, цвета, отступы и размеры.
- map.css: Дополнительные стили для карты склада, такие как цвета зон, рамки стеллажей и метки.

### JavaScript

- map.js: Отвечает за взаимодействие с картой склада, обработку кликов по зонам, отображение дронов и их перемещение.

- login.js: Обеспечивает валидацию формы входа и отправку данных на сервер для аутентификации.

Взаимодействие с серверной частью

- HTML-формы отправляют запросы на сервер для аутентификации и выполнения операций управления дронами.

- JavaScript скрипты могут обрабатывать ответы от сервера и динамически обновлять содержимое страницы без перезагрузки.

Интеграция с серверной частью

- Для отображения динамических данных на страницах используется шаблон Thymeleaf, который позволяет интегрировать данные, полученные с сервера, напрямую в HTML-разметку.

- Взаимодействие с сервером осуществляется через HTTP-запросы, отправляемые с помощью JavaScript, и обработку ответов от сервера для обновления содержимого страницы.

Дополнительные компоненты

- Файлы изображений: логотипы
- Другие сторонние библиотеки: например, jspdf используется для генерации PDF-отчетов, что может быть полезно для пользователей при анализе данных и отчетности.

#### **4.2.1 Описание интерфейса Drones.html**

Назначение: представляет собой главную страницу системы управления беспилотными устройствами.

Используемые технологии:

- Thymeleaf для шаблонизации и встраивания выражений Spring Security.

- JavaScript библиотека jsPDF для работы с PDF-файлами.
- CSS для стилизации страницы.

Структура:

- Шапка страницы с логотипом и информацией о пользователе.
- Раздел с картой склада.
- Раздел управления беспилотными устройствами.
- Раздел для вывода лога событий.
- Раздел со списком беспилотных устройств.
- Раздел для управления аварийной остановкой устройств.

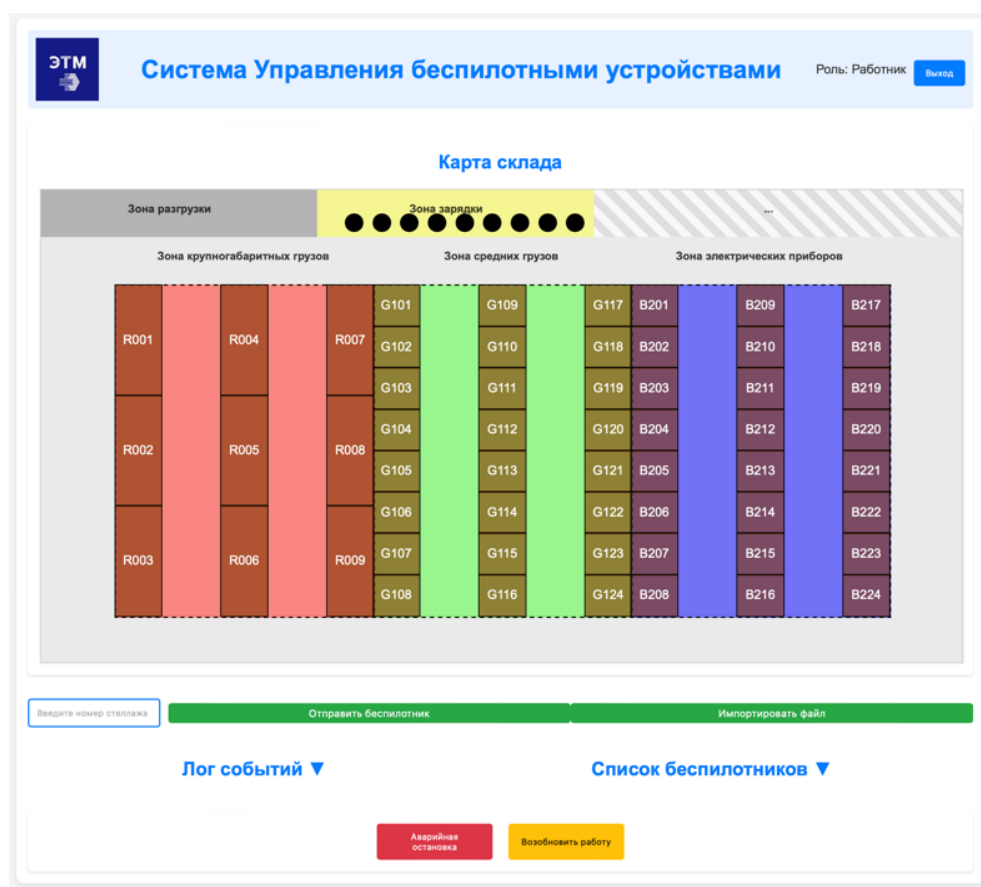


Рисунок 34 – главная страница системы



#### 4.2.2 Описание интерфейса Login.html

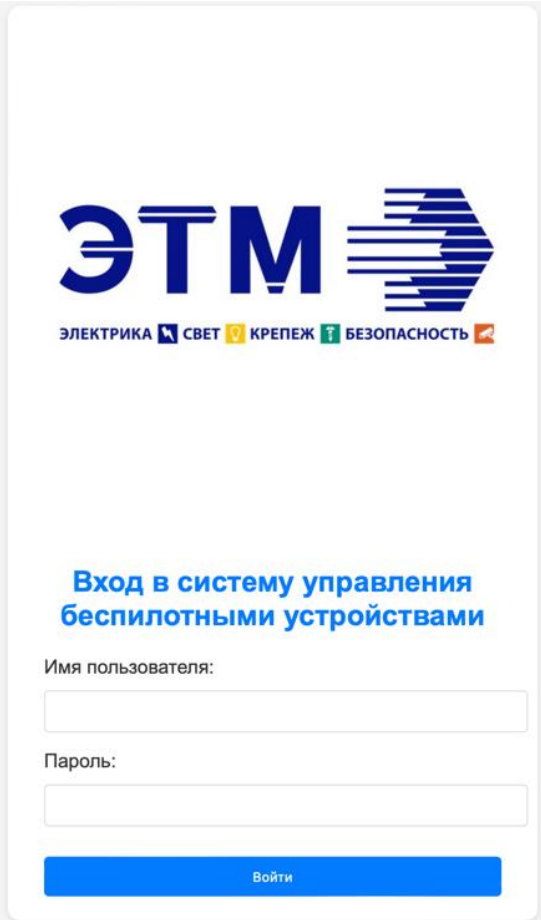
Назначение: Страница входа в систему управления беспилотными устройствами.

Используемые технологии:

- HTML для разметки структуры страницы.
- CSS для стилизации элементов формы.

Структура:

- Форма для ввода данных пользователя (имя пользователя и пароль).
- Кнопка "Войти" для отправки данных на сервер для аутентификации.



The screenshot shows a login page with a white background and a light gray border. At the top center is the logo for "ЭТМ" (ETM) in blue, followed by a stylized blue arrow pointing right. Below the logo is a horizontal line of text: "ЭЛЕКТРИКА" (Electrician) with a blue icon, "СВЕТ" (Light) with a yellow icon, "КРЕПЕЖ" (Fastening) with a green icon, and "БЕЗОПАСНОСТЬ" (Safety) with a red icon. Below this is the title "Вход в систему управления беспилотными устройствами" (Login to the unmanned aircraft system management system) in blue. Under the title are two input fields: "Имя пользователя:" (Username:) and "Пароль:" (Password:). At the bottom is a blue button with the text "Войти" (Login).

Рисунок 35 – Окно авторизации

### 4.2.3 Описания функций Map.js

JavaScript файл содержит логику для управления интерактивной картой склада с беспилотниками, обрабатывает события пользователя и управляет динамическими элементами интерфейса. Ниже приведено описание основных функций в файле:

Основные функции:

1. `initializeMap()`
  - Назначение: Инициализация карты склада.
  - Входные данные: Нет.
  - Выходные данные: Нет.
  - Действия: создает зоны на карте, включая зоны разгрузки, зарядки, офиса и стеллажи для хранения товаров.
2. `createZones(map)`
  - Назначение: Создание зон на карте.
  - Входные данные: `map` (`HTMLElement`) — элемент карты, к которому будут добавлены зоны.
  - Выходные данные: Нет.
  - Действия: динамически создает и стилизует элементы для каждой зоны (красная, зеленая, синяя).
3. `createShelves(map)`
  - Назначение: Создание стеллажей в каждой зоне.
  - Входные данные: `map` (`HTMLElement`).
  - Выходные данные: Нет.

- Действия: распределяет стеллажи в зонах, вычисляет их позицию и размер.

#### 4. `toggleSection(element)`

- Назначение: Переключение видимости разделов интерфейса.
- Входные данные: `element` (`HTMLElement`) — элемент, активирующий функцию.
- Выходные данные: Нет.
- Действия: изменяет класс `'open'` у родительского элемента для показа или скрытия содержимого.

#### 5. `createChargingZone(map)`, `createUnloadingZone(map)`, `createOfficeZone(map)`

- Назначение: Создание специализированных зон на карте.
- Входные данные: `map` (`HTMLElement`).
- Выходные данные: Нет.
- Действия: создает и стилизует зоны зарядки, разгрузки и офиса.

#### 6. `createZoneLabels(map)`

- Назначение: Добавление меток к зонам.
- Входные данные: `map` (`HTMLElement`).
- Выходные данные: Нет.
- Действия: создает и добавляет метки для каждой зоны, обеспечивая наглядное обозначение зон на карте.

#### 7. `sendDroneToShelf()`

- Назначение: Отправка дрона к стеллажу.

- Входные данные: нет (использует глобальные переменные для состояния).

- Выходные данные: Нет.

- Действия: обрабатывает логику отправки дрона, включая проверку и поиск подходящего дрона и расчет маршрута.

#### 8. moveDrone(keyCode)

- Назначение: Управление перемещением дрона по карте.

- Входные данные: keyCode (Number) — код клавиши.

- Выходные данные: Нет.

- Действия: перемещает дрон в зависимости от нажатой клавиши (вверх, вниз, влево, вправо), проверяет допустимость перемещения.

#### 9. addToLog(droneId, eventDescription, eventTime)

- Назначение: Добавление событий в лог.

- Входные данные: droneId (String), eventDescription (String), eventTime (Date, опционально).

- Выходные данные: Нет.

- Действия: регистрирует события, связанные с дронами, в лог событий, помогая отслеживать операции на складе.

#### 10. createReport()

- Назначение: Создание отчета о деятельности на складе.

- Входные данные: нет (использует данные из лога).

- Выходные данные: PDF-отчет.

- Действия: генерирует PDF-отчет, содержащий информацию о выполненных заданиях и используемых дронах.

#### 11. `initializeDrones()`

- Назначение: Инициализация беспилотников на карте.
- Входные данные: Нет.
- Выходные данные: Нет.
- Действия: создает представления беспилотников и добавляет их на карту склада. Также инициализирует список элементов управления для каждого дрона.

#### 12. `createDrone(index)`

- Назначение: Создание элемента беспилотника.
- Входные данные: `index (Number)` — индекс для идентификации дрона.
- Выходные данные: `HTMLElement` — созданный элемент дрона.
- Действия: создает и стилизует элемент дрона, устанавливает его начальное положение на карте.

#### 13. `createListItem(index, droneId)`

- Назначение: Создание элемента списка для управления дроном.
- Входные данные: `index (Number)`, `droneId (String)` — идентификатор дрона.
- Выходные данные: `HTMLElement` — созданный элемент списка.
- Действия: создает элемент списка, который отображает информацию о дроне и обеспечивает интерактивное управление выбором дрона.

#### 14. `selectDrone(droneId)`

- Назначение: Выбор и активация дрона.

- Входные данные: `droneId (String)` — идентификатор дрона.
- Выходные данные: Нет.
- Действия: активирует выбранный дрон, выделяя его на интерфейсе.

#### 15. `deselectAllDrones()`

- Назначение: Снятие выбора со всех дронов.
- Входные данные: Нет.
- Выходные данные: Нет.
- Действия: удаляет выделение и активацию со всех дронов, очищает выбор пользователя.

#### 16. `isWithinBounds(x, y, bounds)`

- Назначение: Проверка нахождения координат в заданных границах.
- Входные данные: `x (Number)`, `y (Number)`, `bounds (Object)` — границы для проверки.
- Выходные данные: `Boolean` — результат проверки.
- Действия: определяет, находятся ли заданные координаты внутри указанных границ.

#### 17. `isCollisionWithShelf(droneId, x, y)`

- Назначение: Проверка на столкновение дрона со стеллажом.
- Входные данные: `droneId (String)`, `x (Number)`, `y (Number)`.
- Выходные данные: `Boolean` — результат проверки.
- Действия: определяет, произошло ли столкновение дрона со стеллажами в его зоне.

#### 18. `isAllowedToMove(droneId, x, y)`

- Назначение: Проверка возможности перемещения дрона.
- Входные данные: `droneId (String)`, `x (Number)`, `y (Number)`.
- Выходные данные: `Boolean` — разрешено ли перемещение.
- Действия: проверяет, не выходит ли дрон за границы карты, зон или стеллажей.

19. `moveDroneAlongPath(droneId, path, shelfName, onComplete)`

- Назначение: Перемещение дрона по заданному пути.
- Входные данные: `droneId (String)`, `path (Array of Arrays)`, `shelfName (String)`, `onComplete (Function)` — колбэк по завершению движения.
- Выходные данные: Нет.
- Действия: перемещает дрон по маршруту, отображая его перемещение на карте, и вызывает функцию `onComplete` по завершении.

20. `performTaskAfterDelivery(droneId, shelfName)`

- Назначение: Выполнение задачи после доставки на стеллаж.
- Входные данные: `droneId (String)`, `shelfName (String)`.
- Выходные данные: Нет.
- Действия: обозначает успешное завершение задачи изменением визуального состояния дрона и записывает событие в лог.

21. `moveDrone(keyCode)`

- Назначение: обрабатывает управление дроном с клавиатуры.
- Входные данные: `keyCode (Number)` — код клавиши.
- Выходные данные: Нет.

- Действия: изменяет позицию выбранного дрона в зависимости от нажатой клавиши (влево, вправо, вверх, вниз), если это разрешено функцией `isAllowedToMove`.

## 22. `sendDroneToShelf()`

- Назначение: инициирует процесс отправки дрона к стеллажу.
- Входные данные: нет (читает значение из DOM).
- Выходные данные: Нет.
- Действия: проверяет наличие активного процесса отправки, выбирает дрона и маршрут для доставки к указанному стеллажу, а также запускает движение дрона по маршруту.

## 23. `handleFileSelect(event)`

- Назначение: обрабатывает выбор файла пользователем.
- Входные данные: `event (Event)` — событие выбора файла.
- Выходные данные: Нет.
- Действия: читает содержимое файла, разбивает на строки (представляющие стеллажи), и инициирует автоматическую отправку дронов по этим стеллажам.

## 24. `startSendingDrones()`

- Назначение: запускает интервальную отправку дронов на стеллажи.
- Входные данные: Нет.
- Выходные данные: Нет.
- Действия: осуществляет периодическую проверку очереди стеллажей и отправляет дроны на стеллажи из очереди, пока очередь не опустеет.



25. `addToLog(droneId, eventDescription, eventTime)`

- Назначение: добавляет запись в журнал событий.
- Входные данные: `droneId` (String), `eventDescription` (String), `eventTime` (Date, опционально).
- Выходные данные: Нет.
- Действия: записывает в журнал информацию о событиях, связанных с дронами, с указанием времени события.

Эти функции охватывают управление дронами, обработку событий в интерфейсе пользователя и взаимодействие с картой склада. Они важны для обеспечения функциональности системы управления беспилотными устройствами, что позволяет пользователям эффективно выполнять задачи по перемещению грузов и управлению дронами.

## ЗАКЛЮЧЕНИЕ

В ходе разработки системы управления беспилотными устройствами для компании ЭТМ были выполнены все ключевые этапы проекта, начиная от анализа предметной области и заканчивая реализацией и тестированием системы. Это позволило создать полнофункциональное программное обеспечение, способное управлять беспилотниками для автоматизации складских операций.

Система включает в себя не только функции управления перемещением беспилотников, но и возможности для мониторинга их статуса, создания отчетов и взаимодействия с интерактивной картой склада. Это обеспечивает операторам полный контроль над процессами на складе и позволяет эффективно управлять распределением задач между беспилотными устройствами.

Основные достоинства системы:

- Автоматизация процессов: система сокращает время на выполнение рутинных операций, таких как перемещение грузов, благодаря автоматизированному контролю дронов.
- Повышенная оперативность: мгновенный доступ к данным о местоположении и статусе дронов позволяет оперативно реагировать на изменения в рабочем процессе.
- Интуитивно понятный интерфейс: система разработана с учетом удобства пользователей, что облегчает ежедневную работу и обучение новых операторов.
- Гибкость и масштабируемость: благодаря модульной архитектуре и использованию современных технологий, система легко адаптируется к изменяющимся требованиям и масштабу предприятия.

Программа работает на локальном сервере и доступна через защищенное соединение, обеспечивая надежность и безопасность передачи данных.

Реализация системы на платформе Mac OS и требование использования IntelliJ IDEA для разработки подчеркивают современный подход к созданию и поддержке программного обеспечения.

Дальнейшее развитие системы может включать в себя интеграцию с дополнительными складскими системами, улучшение алгоритмов маршрутизации дронов, а также расширение функционала по созданию и анализу отчетов для обеспечения более глубокого анализа данных.

В целом, проект демонстрирует успешную реализацию инновационного решения для автоматизации складских процессов, что делает его важным шагом в цифровизации и оптимизации работы компании ЭТМ.

## СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Baeldung. "Spring Boot and Microservices." Онлайн-ресурс, предлагающий учебные пособия и курсы по Spring Boot и микросервисной архитектуре.
2. Oracle. "Java Platform Standard Ed. 8 Documentation." Официальная документация по Java SE, предоставляющая обширные руководства.
3. Mozilla Developer Network. "JavaScript Guide." Онлайн-руководство по JavaScript от Mozilla, содержащее сведения по всем аспектам языка.
4. Хорст Клейн "Automated Guided Vehicle Systems: a Primer with Practical Applications.". Это руководство по автоматизированным системам управления беспилотными транспортными средствами.
5. Сьюзан Лейбл "Warehouse Management: A Complete Guide for Retailers.". Книга представляет собой подробное руководство по управлению складом, включая использование автоматизированных систем для повышения эффективности.
6. Грегор Хопе, Бобби Вульф "Enterprise Integration Patterns: Designing, Building, and Deploying Messaging Solutions.". Книга описывает шаблоны проектирования и практики интеграции предприятий, которые могут быть применены при разработке систем на базе Spring и Java.
7. Брайан Гетц и др "Java Concurrency in Practice.". Эта книга — основное руководство по многопоточности и параллельному программированию в Java, что крайне важно для разработки серверных приложений и систем управления беспилотниками.
8. Информационные материалы о компании ЭТМ. Официальный сайт компании, содержащий данные о миссии, стратегии и ключевых направлениях деятельности.
9. Ларри Ульман "Modern JavaScript: Develop and Design.". Книга предоставляет обзор современного JavaScript, включая основные концепции, которые могут быть использованы при разработке клиентских и серверных приложений.

10. Майкл Гроовер "Industrial Robotics: Technology, Programming, and Applications.". Освещает ключевые аспекты промышленной робототехники, включая автоматизированные беспилотные системы.

11. Uwe Bracht, Dieter Spath, Thomas Schaeffer "Design and Control of Automated Guided Vehicle Systems: A Case Study Approach.". Предлагает подробный анализ разработки и контроля систем автоматически управляемых транспортных средств на примерах реальных проектов.

12. "Automated Vehicles for Material Handling." Практическое руководство по внедрению и эксплуатации беспилотных транспортных средств в материально-техническом обеспечении.

13. "Smart Warehousing: Advanced Technologies for Warehousing and Supply Chain Management." Обзор передовых технологий в складском хозяйстве и управлении цепочками поставок, включая беспилотные погрузчики и автоматизацию.

**ПРИЛОЖЕНИЕ А**  
**«Системный отчёт»**

The report was generated for the period: \_\_. \_\_.\_\_\_\_, \_\_:\_\_:\_\_ - \_\_.\_\_.\_\_\_\_,  
\_\_:\_\_:\_\_

Delivered goods: \_\_\_\_

Unmanned loaders have been deployed: \_\_

Event Logs:

## **ПРИЛОЖЕНИЕ Б**

### **ИНФОРМАЦИОННАЯ СИСТЕМА УПРАВЛЕНИЯ БЕСПИЛОТНЫМИ УСТРОЙСТВАМИ В СКЛАДСКОМ ПОМЕЩЕНИИ КОМПАНИИ ЭТМ**

**Руководство разработчика**

**Санкт-Петербург 2024**

## СОДЕРЖАНИЕ

<b><u>1. Назначение и условия применения программы</u></b> .....	89
<b><u>1.1 Назначение программы</u></b> .....	89
<b><u>1.2 Функции, выполняемые программой</u></b> .....	89
<b><u>1.3 Условия, необходимые для выполнения программы</u></b> .....	90
1.3.1. Объем оперативной памяти .....	90
1.3.2. Требования к составу периферийных устройств .....	90
1.3.3. Требования к параметрам периферийных устройств .....	90
1.3.4. Требования к программному обеспечению .....	91
1.3.5. Требования к персоналу (программисту) .....	91
<b><u>2. Характеристика программы</u></b> .....	92
<b><u>2.1 Описание основных характеристик программы</u></b> .....	92
2.1.1. Режим работы программы .....	92
<b><u>3. Обращение к программе</u></b> .....	92
<b><u>3.1 Инициализация карты склада</u></b> .....	92
<b><u>3.2 Выбор и отправка дронов на задание</u></b> .....	93
<b><u>3.3 Управление перемещением дрона</u></b> .....	94
<b><u>3.4 Создание стеллажей на карте</u></b> .....	95
<b><u>3.5 Проверка разрешения на движение дрона</u></b> .....	96
<b><u>3.6 Отправка дронов на выполнение заданий</u></b> .....	97
<b><u>3.7 Работа с файлами заданий</u></b> .....	98
<b><u>3.8 Управление статусом дронов</u></b> .....	98
<b><u>3.9 Анимация перемещения дрона</u></b> .....	99
<b><u>3.10 Остановка и возобновление работы всех дронов</u></b> .....	100
<b><u>3.11 Отправка дронов по расписанию из файла</u></b> .....	101
<b><u>3.12 Логирование событий</u></b> .....	101
<b><u>3.13 Создание отчетов</u></b> .....	102
<b><u>3.14 Общая логика работы программы:</u></b> .....	103
<b><u>4. Входные и выходные данные</u></b> .....	104
<b><u>4.1 Организация используемой входной информации</u></b> .....	104
<b><u>4.2 Организация используемой выходной информации</u></b> .....	105



## **1. Назначение и условия применения программы**

### **1.1 Назначение программы**

Программа "Система Управления беспилотными устройствами" предназначена для сотрудников компании ЭТМ. Она позволяет управлять беспилотниками для выполнения задач по перемещению грузов. В программе можно управлять беспилотником самостоятельно или отправить его к стеллажу за грузом, чтобы он доставил его в разгрузочную зону.

### **1.2 Функции, выполняемые программой**

1. Управление беспилотниками:
  - Самостоятельное управление оператором.
  - Автоматическое задание маршрута к стеллажу за грузом и доставка в разгрузочную зону.
2. Отслеживание беспилотников:
  - Просмотр местоположения беспилотников на интерактивной карте.
  - Отображение статуса беспилотников (в работе или нет).
3. Создание отчетов:
  - Формирование отчетов о работе беспилотников.
  - Сохранение отчетов в PDF-формате.
4. Работа с интерактивной картой:
  - Визуализация маршрутов и зон действия беспилотников.
  - Просмотр текущего положения беспилотников в режиме реального времени.
5. Управление загрузкой и разгрузкой:
  - Задание точек загрузки и разгрузки на карте.
  - Мониторинг выполнения задач по загрузке и разгрузке.

### **1.3 Условия, необходимые для выполнения программы**

Для обеспечения надлежащей работы системы управления беспилотными устройствами важно соблюдение определенных технических требований. Эти требования помогут гарантировать стабильность и эффективность системы.

#### **1.3.1. Объем оперативной памяти**

Для корректной работы системы требуется минимум 8 ГБ оперативной памяти. Рекомендуется использовать 16 ГБ или более, особенно если предполагается одновременная работа с множеством беспилотников и обработка больших объемов данных.

#### **1.3.2. Требования к составу периферийных устройств**

Для взаимодействия с системой необходимы следующие периферийные устройства:

- Клавиатура и мышь для управления интерфейсом.
- Монитор с разрешением не менее 1920x1080 для четкого отображения информации.
- Наличие веб-камеры и микрофона может быть необходимо для проведения видеоконференций и совещаний в режиме онлайн.

#### **1.3.3. Требования к параметрам периферийных устройств**

- Монитор должен поддерживать разрешение Full HD (1920x1080) или выше для корректного отображения графического интерфейса системы.
- Клавиатура и мышь должны быть исправны и подходить для интенсивного использования.

### **1.3.4. Требования к программному обеспечению**

Для работы системы требуется:

- Операционная система macOS 14 Sonoma или новее.
- Браузер Safari последней версии для доступа к веб-интерфейсу системы.
- Установленное программное обеспечение IntelliJ IDEA версии 2023.2.2 или новее для возможности программных разработок и тестирования.
- Наличие актуальных обновлений безопасности и патчей для всех используемых программ.

### **1.3.5. Требования к персоналу (программисту)**

Для работы с системой программист или технический специалист должен обладать следующими квалификациями:

- Знание языка программирования Java и опыт работы с фреймворком Spring, так как система использует эти технологии.
- Опыт работы с базами данных и умение обращаться с серверными приложениями.
- Знание основ работы с операционными системами, особенно macOS, на уровне администратора.
- Навыки работы с системами контроля версий, например, Git для управления версиями программного кода.

Соблюдение этих требований обеспечит стабильную и эффективную работу системы, а также упростит процесс разработки и поддержки.

## 2. Характеристика программы

### 2.1 Описание основных характеристик программы

Система управления беспилотными устройствами представляет собой комплексное программное решение, разработанное для мониторинга и управления беспилотными погрузчиками на предприятиях. Она позволяет установить логические схемы работы системы, обеспечивает двустороннее взаимодействие оператора с системой и автоматизирует процесс выявления и устранения неполадок.

#### 2.1.1. Режим работы программы

Программа работает в основном режиме:

Основной режим работы, когда программа выполняется на рабочем сервере с установленной операционной системой и активным интернет-соединением.

## 3. Обращение к программе

### 3.1 Инициализация карты склада

**Функция:** `initializeMap()`

**Назначение:** Инициализация карты склада с созданием зон разгрузки, зарядки, офиса, а также стеллажей для хранения товаров.

**Логика работы:** Функция `initializeMap()` отвечает за создание начального вида карты склада. Она вызывает ряд подфункций для добавления различных элементов на карту:

- **`createZones(map)`** - создает основные зоны на карте: красная, зеленая и синяя зоны для различных категорий товаров.
- **`createShelves(map)`** - создает стеллажи в каждой зоне.

- **createChargingZone(map)** - создает зону зарядки дронов.
- **createUnloadingZone(map)** - создает зону разгрузки товаров.
- **createOfficeZone(map)** - создает офисную зону.
- **createZoneLabels(map)** - добавляет метки к созданным зонам для улучшения ориентации на карте.

- **Типы данных:**
- **map: HTMLElement** — Элемент DOM, который используется как контейнер для карты склада.

Каждая из этих функций подробно структурирует и визуализирует соответствующие секции карты, что позволяет пользователям системы эффективно навигировать и управлять задачами беспилотников.

### 3.2 Выбор и отправка дронов на задание

#### Функция: **sendDroneToShelf()**

**Назначение:** Обработка логики отправки дрона к стеллажу для выполнения задания по перемещению товаров.

**Логика работы:** Функция **sendDroneToShelf()** активируется при взаимодействии пользователя с интерфейсом отправки дрона (кнопка отправки). Процесс включает следующие шаги:

1. Проверка, идет ли уже отправка другого дрона (через флаг **sendingInProgress**).
2. Получение номера стеллажа из пользовательского ввода и его координат через **getShelfCoordinates()**.
3. Выбор подходящего дрона для выполнения задачи с помощью **selectDroneForShelf()**.
4. Расчет маршрута до стеллажа через **calculatePath()**.

5. Последовательное перемещение дрона по заданному маршруту с помощью **moveDroneAlongPath()**, которая также обрабатывает логику прибытия дрона на место и последующие действия.

- **Параметры функции sendDroneToShelf():**
- **Входные данные:** Не принимает параметры напрямую, использует глобальные переменные и состояния.
- **Возвращаемое значение:** Не возвращает значения, изменяет состояние системы и DOM.

Функция охватывает полный цикл задачи для дрона, начиная от начала движения и заканчивая выполнением задания, включая обработку ошибок и исключительных ситуаций, таких как отсутствие стеллажа или дрона.

### 3.3 Управление перемещением дрона

#### Функция: **moveDrone()**

**Назначение:** Управление перемещением дрона в зависимости от нажатий клавиш пользователя.

**Логика работы:** Функция **moveDrone(keyCode)** вызывается при нажатии клавиш управления (стрелки клавиатуры). Функция проверяет, допустимо ли движение дрона в заданном направлении с помощью **isAllowedToMove()**, которая учитывает границы карты, зоны столкновений и специфические зоны, такие как зоны зарядки и разгрузки. Если движение допустимо, обновляет позицию дрона на карте.

- **Параметры функции moveDrone(keyCode):**
- **keyCode: Number** — Целочисленное значение, представляющее нажатую клавишу.
- **Возвращаемое значение:** Не возвращает значения, но обновляет стили CSS для позиционирования элемента на карте.

Эта функция критически важна для интерактивной части системы, позволяя операторам в реальном времени контролировать и корректировать маршруты дронов в зависимости от изменяющихся условий склада.

### 3.4 Создание стеллажей на карте

#### Функция: `createShelves(map)`

**Назначение:** Добавление стеллажей в определенные зоны на карте склада.

**Логика работы:** Функция `createShelves(map)` отвечает за создание и размещение стеллажей в трех различных зонах (красной, зеленой и синей) на карте склада. Процесс включает:

- Определение количества стеллажей для каждой зоны.
- Расчет положения каждого стеллажа в процентном соотношении от размеров зоны.
- Создание HTML-элементов для каждого стеллажа и их стилизация.
- Размещение стеллажей в соответствующих зонах на карте.
- **Параметры функции `createShelves(map)`:**
- **`map`: `HTMLElement`** — Элемент DOM карты, на которой располагаются стеллажи.
- **Возвращаемое значение:** Не возвращает значения, прямо влияет на DOM, добавляя элементы стеллажей.

Функция использует данные о зонах и количестве стеллажей, чтобы динамически сгенерировать необходимые элементы и правильно их позиционировать, обеспечивая точное отображение физической структуры склада в интерфейсе пользователя.

### 3.5 Проверка разрешения на движение дрона

**Функция:** `isAllowedToMove(droneId, x, y)`

**Назначение:** Определение, разрешено ли дрону перемещаться в заданную позицию на карте.

**Логика работы:** Функция `isAllowedToMove(droneId, x, y)` используется для проверки, может ли дрон переместиться в указанную позицию, учитывая ряд ограничений:

- Проверка нахождения дрона в пределах границ карты.
- Проверка столкновения со стеллажами или другими препятствиями.
- Учет зон, специфичных для дрона (например, зона зарядки и разгрузки).
- **Параметры функции `isAllowedToMove(droneId, x, y)`:**
  - **`droneId: String`** — идентификатор дрона.
  - **`x, y: Number`** — координаты, к которым дрон пытается переместиться.
  - **Возвращаемое значение: `Boolean`** — возвращает `true`, если движение возможно, и `false`, если нет.

Эта функция критична для предотвращения ошибок управления и обеспечения безопасности дронов на складе, предотвращая столкновения и другие потенциальные аварийные ситуации.



### 3.6 Отправка дронов на выполнение заданий

**Функция:** `moveDroneAlongPath(droneId, path, shelfName, onComplete)`

**Назначение:** Управление движением дрона вдоль заданного маршрута до стеллажа.

**Логика работы:** Функция `moveDroneAlongPath(droneId, path, shelfName, onComplete)` отвечает за анимацию перемещения дрона по карте склада. Процесс включает:

- Постепенное обновление позиции дрона на карте в соответствии с маршрутом.
- Отслеживание завершения перемещения дрона.
- Вызов callback-функции `onComplete`, когда дрон достигает конечной точки маршрута.
- **Параметры функции `moveDroneAlongPath(droneId, path, shelfName, onComplete)`:**
  - **`droneId: String`** — идентификатор дрона.
  - **`path: Array of Arrays`** — массив координат (массивов), описывающих путь.
  - **`shelfName: String`** — имя стеллажа, цель перемещения.
  - **`onComplete: Function`** — функция обратного вызова, которая выполняется по завершении перемещения.
- **Возвращаемое значение:** Не возвращает значения, реализует асинхронное перемещение.

Функция обеспечивает плавное и точное перемещение дронов на складе, поддерживая реалистичное взаимодействие с пользователем и системой.

### 3.7 Работа с файлами заданий

**Функция:** `handleFileSelect(event)`

**Назначение:** Обработка выбора файла с заданиями для дронов.

**Логика работы:** Функция `handleFileSelect(event)` активируется при выборе файла пользователем. Она читает содержимое файла, разделяет его на отдельные задания и инициирует процесс отправки дронов по заданным маршрутам. Процесс включает:

- Чтение и анализ содержимого файла.
- Разбивка содержимого на индивидуальные задания.
- Инициация последовательной отправки дронов для выполнения заданий.
- **Параметры функции `handleFileSelect(event)`:**
- **event: Event** — объект события, содержащий данные о выбранном файле.
- **Возвращаемое значение:** Не возвращает значения, запускает процесс чтения и обработки файла.

Эта функция позволяет автоматизировать процесс задания маршрутов для дронов, упрощая масштабное управление операциями на складе.

### 3.8 Управление статусом дронов

**Функция:** `selectDroneForShelf(shelfNumber)`

**Назначение:** Выбор доступного дрона для выполнения задания на указанном стеллаже.

**Логика работы:** Функция `selectDroneForShelf(shelfNumber)` определяет, какой из дронов доступен для выполнения задания в соответствии

с их текущим статусом и зоной, к которой относится стеллаж. Процесс включает:

- Определение зоны стеллажа по его номеру.
- Поиск дрона, который не занят (свободен) и находится в нужной зоне.
- Возврат идентификатора первого подходящего дрона.
- **Параметры функции `selectDroneForShelf(shelfNumber)`:**
- **`shelfNumber: String`** — номер стеллажа, для которого нужно выбрать дрона.
- **Возвращаемое значение: `String` или `null`** — возвращает идентификатор дрона или `null`, если подходящего дрона нет.

Эта функция критична для распределения заданий между дронами, обеспечивая их эффективное использование без перекрытия заданий.

### 3.9 Анимация перемещения дрона

**Функция: `moveDroneAlongPath(droneId, path, shelfName, onComplete)`**

**Назначение:** Плавное перемещение дрона по заданному пути.

**Логика работы:** Эта функция используется для анимированного перемещения дрона вдоль вычисленного маршрута. Включает в себя:

- Постепенное обновление позиции дрона на карте в соответствии с путевыми точками.
- При достижении конечной точки маршрута вызывается функция **`onComplete`**, которая может обрабатывать последующие задачи, например, разгрузку товара.

– **Техническое уточнение:** функция **moveDroneAlongPath()** использует **setTimeout** для анимации, что требует управления состоянием анимации и потенциального взаимодействия с пользовательским интерфейсом в реальном времени.

Функция обеспечивает визуально плавное передвижение дронов, улучшая интерактивность и пользовательский опыт системы управления складом.

### 3.10 Остановка и возобновление работы всех дронов

**Функция:** **stopAllDrones()**

**Назначение:** Временная остановка всех дронов и их анимаций.

**Логика работы:** Функция **stopAllDrones()** используется для остановки всех операций дронов в случае необходимости, например, для обслуживания или в экстренных ситуациях. Она:

- Устанавливает флаг **dronesPaused** в **true**.
- Добавляет визуальный эффект (мигание) для указания на статус остановки.

Функция **resumeAllDrones()** затем используется для возобновления работы, сбрасывая флаг и убирая визуальные эффекты. Эти функции обеспечивают управление потоком работы дронов, что критически важно для управления ресурсами и безопасности.

### 3.11 Отправка дронов по расписанию из файла

**Функция:** `handleFileSelect(event)`

**Назначение:** Чтение и выполнение заданий для дронов из загруженного файла.

**Логика работы:** Функция `handleFileSelect(event)` активируется при выборе файла с заданиями. Она:

- Читает файл.
- Извлекает задания в виде списка номеров стеллажей.
- Иницирует процесс последовательной отправки дронов на стеллажи, используя функцию `sendDroneToShelf()`.

Эта функция позволяет автоматизировать процессы планирования и распределения заданий для дронов, улучшая операционную эффективность складских операций.

### 3.12 Логирование событий

**Функция:** `addToLog(droneId, eventDescription, eventTime)`

**Назначение:** Добавление записей о событиях, связанных с дронами, в журнал событий.

**Логика работы:** Функция `addToLog()` используется для записи важных событий, таких как достижение стеллажа или возвращение в зону разгрузки, в журнал. Эта функция принимает идентификатор дрона, описание события и время события, добавляя запись в начало списка журнала на веб-странице. Это позволяет операторам системы в реальном времени видеть все активности и

быстро реагировать на возникающие проблемы или изменения состояния дронов.

– **Параметры функции `addToLog(droneId, eventDescription, eventTime)`:**

– **`droneId: String`** — идентификатор дрона, событие которого регистрируется.

– **`eventDescription: String`** — текстовое описание события.

– **`eventTime: Date`** — время события, объект `Date`.

– **Возвращаемое значение:** Не возвращает значения, добавляет элемент в список журнала.

### 3.13 Создание отчетов

**Функция: `createReport()`**

**Назначение:** Генерация подробного отчета о всех задачах, выполненных дронами, включая даты, время и подробности выполнения.

**Логика работы:** Функция `createReport()` собирает все данные из журнала событий, обрабатывает их и форматирует в виде подробного отчета. Этот отчет может включать информацию о количестве доставок, использованных дронах и хронологию событий. Отчет затем может быть сохранен в формате PDF для архивации или распечатки, что полезно для анализа производительности системы и планирования последующих операций.

– **Параметры функции `createReport()`:**

– **Входные данные:** Использует данные из журнала событий.

– **Возвращаемое значение:** Не возвращает значения, генерирует PDF-отчет.

### 3.14 Общая логика работы программы:

Программа предназначена для управления автоматизированной системой склада с использованием беспилотных летательных аппаратов (дронов) для перемещения грузов между стеллажами, зонами зарядки, разгрузки и офисными зонами.

#### Логика работы:

1. **Инициализация карты склада:** При загрузке документа (**DOMContentLoaded**) иницируется функция **initializeMap()**, которая строит виртуальное представление склада с различными функциональными зонами: зоны для различных типов грузов (красная, зеленая, синяя), зону разгрузки, зону зарядки и офисную зону. Каждая зона имеет уникальные свойства и задачи.

2. **Размещение стеллажей и дронов:** **createShelves(map)** добавляет стеллажи в соответствующие зоны. Каждый стеллаж идентифицируется уникальным номером и координатами. **initializeDrones()** размещает дроны на карте и создает интерактивный список, позволяющий выбирать и управлять дронами.

#### 3. Управление дронами:

– **Выбор и отправка дронов:** Пользователи могут выбирать дроны и отправлять их к стеллажам для перемещения грузов. **sendDroneToShelf()** обрабатывает запросы на отправку, выбирая подходящего дрона и вычисляя маршрут передвижения.

– **Перемещение дронов:** Функции **moveDroneAlongPath()** и **calculatePath()** отвечают за навигацию дрона к заданной цели на карте, учитывая препятствия и ограничения зоны.

4. **Логирование и отчетность:** События, такие как достижение дроном стеллажа или зоны разгрузки, регистрируются в журнале событий с

помощью **addToLog()**. **createReport()** собирает данные из журнала для создания подробного отчета о выполненных операциях.

5. **Интерактивность и управление:** Пользовательский интерфейс позволяет в реальном времени взаимодействовать с системой, выбирая дроны и управляя их перемещением с помощью клавиатурных команд. Система поддерживает остановку и возобновление работы дронов через функции **stopAllDrones()** и **resumeAllDrones()**.

6. **Обработка файлов:** Пользователи могут загружать задания для дронов через файлы, которые обрабатываются функцией **handleFileSelect()**. Это позволяет автоматизировать процесс задания маршрутов и операций для множества дронов.

Эта общая логика работы программы обеспечивает эффективное управление ресурсами склада, оптимизацию маршрутов и повышение точности и скорости обработки заказов за счет автоматизации с использованием беспилотных аппаратов.

## 4. Входные и выходные данные

### 4.1 Организация используемой входной информации

Входная информация для системы управления беспилотными устройствами на складе включает:

- **Ввод с клавиатуры:** Управление дронами с помощью клавиш на клавиатуре для перемещения в разные направления.
- **Файлы заданий:** Пользователи могут загружать файлы с заданиями, которые содержат информацию о том, какие стеллажи дроны должны посетить для перемещения или учета товаров.
- **Сенсорные данные с дронов:** Получение данных о положении, состоянии батареи, и других параметрах дронов в реальном времени.



- **Данные с интерфейса пользователя:** Пользовательский ввод через графический интерфейс веб-приложения для задач, таких как выбор дронов, отправка на задания и мониторинг их выполнения.

## **4.2 Организация используемой выходной информации**

Выходная информация, генерируемая системой, обеспечивает:

- **Отправка команд дронам:** Передача управляющих сигналов дронам для выполнения специфических задач, таких как перемещение к стеллажу или возврат в зону зарядки.

- **Данные для пользовательского интерфейса:** Обновление информации на панели управления, отображение текущего статуса дронов, их местоположения на карте склада и статуса выполнения заданий.

- **Логирование событий:** Регистрация всех действий и событий в системе, которые включают время, тип операции, и идентификатор дрона, для возможности анализа и отладки.

- **Генерация отчетов:** Создание подробных отчетов о выполненных заданиях, которые могут включать данные о времени выполнения, задействованных дронах и успешности выполнения задач.

Эти пункты структурируют потоки данных в системе управления беспилотными устройствами, упорядочивая взаимодействие между пользователем, оборудованием и программным обеспечением, обеспечивая высокую производительность и точность операций на складе.

## **ПРИЛОЖЕНИЕ В**

### **ИНФОРМАЦИОННАЯ СИСТЕМА УПРАВЛЕНИЯ БЕСПИЛОТНЫМИ УСТРОЙСТВАМИ В СКЛАДСКОМ ПОМЕЩЕНИИ КОМПАНИИ ЭТМ**

#### **Руководство пользователя**

**Санкт-Петербург 2024**

## СОДЕРЖАНИЕ

<b>1. Введение .....</b>	<b>108</b>
<b>1.1. Область применения .....</b>	<b>108</b>
<b>1.2. Краткое описание возможностей .....</b>	<b>108</b>
<b>1.3. Уровень подготовки пользователя .....</b>	<b>109</b>
<b>2. Назначение и условия применения системы .....</b>	<b>110</b>
<b>3. Подготовка к работе.....</b>	<b>111</b>
<b>3.1 Порядок загрузки данных и программ .....</b>	<b>111</b>
<b>3.2. Порядок проверки работоспособности.....</b>	<b>112</b>
<b>4.Описание функций .....</b>	<b>114</b>
<b>5. Аварийные ситуации .....</b>	<b>117</b>
<b>6. Рекомендации по освоению.....</b>	<b>118</b>

## **1. Введение**

### **1.1. Область применения**

Требования настоящего документа применяются при проведении предварительного тестирования, эксплуатационного тестирования, эксплуатации.

### **1.2. Краткое описание возможностей**

Система управления беспилотными устройствами, разработанная для компании ЭТМ, предназначена для координации и управления беспилотными устройствами на территории предприятия. Основные пользователи системы — сотрудники компании, которые используют программу для автоматизации логистических и складских операций с помощью беспилотников. Система позволяет:

**Управление беспилотниками:** оператор может в ручном режиме управлять движением беспилотника или задать автоматический режим для выполнения задач, таких как транспортировка груза к стеллажам и доставка его до разгрузочной зоны.

**Мониторинг статуса и местоположения беспилотников:** в режиме реального времени на интерактивной карте отображается текущее местоположение каждого устройства и его статус (активен или нет).

**Управление заданиями:** возможность назначения задач беспилотникам и отслеживание их выполнения.

**Просмотр списка беспилотников:** доступ к актуальному списку всех беспилотных устройств в системе.

**Создание отчетов:** генерация отчетов по результатам работы беспилотников, включая детали выполненных задач и статусов устройств. Программа гарантирует высокий уровень защиты данных, работая как на отдельных компьютерах, так и в интернете. Защита информации

осуществляется с помощью специализированных инструментов безопасности (Java Spring Security). Это означает, что все данные надежно защищены от взлома и несанкционированного доступа, позволяя использовать программу с уверенностью в сохранности конфиденциальной информации.

### **1.3. Уровень подготовки пользователя**

Для эффективной работы с системой управления беспилотными устройствами требуется, чтобы пользователь имел определенные навыки и знания. Система разработана для операционной системы macOS 14 Sonoma, поэтому пользователи должны обладать следующими компетенциями:

**Опыт работы с операционной системой macOS 14 Sonoma:** пользователи должны быть знакомы с основными функциями и интерфейсом системы macOS, включая специфические особенности и обновления данной версии операционной системы.

**Навыки работы с браузером Safari:** умение эффективно использовать браузер Safari для доступа к веб-ресурсам, так как он является основным средством доступа к системе управления.

**Умение работать с PDF-документами:** пользователи должны уметь открывать, просматривать и возможно редактировать PDF-документы, так как отчеты и некоторые управленческие документы системы предоставляются в этом формате.

**Знание файловой системы операционной системы macOS:** основное внимание уделяется умению работать с файловым менеджером Finder для организации и управления файлами, что необходимо для эффективной работы с документацией и отчетами системы.

Эти навыки необходимы для того, чтобы пользователи могли полноценно взаимодействовать с системой, обеспечивать её стабильную и безопасную работу, а также эффективно решать поставленные перед ними задачи.

## **2. Назначение и условия применения системы**

Система управления беспилотными устройствами создана для того, чтобы сделать проще и удобнее работу сотрудников компании ЭТМ, которые занимаются управлением беспилотниками и обслуживанием складских операций. Эта система помогает организовать информацию о беспилотниках, их заданиях и местоположении, что ускоряет и улучшает работу всей компании.

Основные функции системы:

**Управление беспилотниками:** Система позволяет сотрудникам отправлять беспилотники за грузами и контролировать их маршруты на складе.

**Отслеживание:** Можно в любой момент проверить, где находится беспилотник и что он делает.

**Отчеты:** Система умеет создавать отчеты о работе беспилотников, что помогает анализировать производительность.

Система работает на компьютерах с операционной системой macOS 14 Sonoma, и для доступа к ней нужно использовать браузер Safari. Это означает, что пользователи должны уметь пользоваться компьютером с данной операционной системой и браузером для работы с системой.

Система предназначена для использования определенными сотрудниками с разными уровнями доступа:

**Работник:** имеет доступ ко всем функциям системы и отчетам.

**Админ:** имеет доступ ко всем функциям системы, отчетам, файлам системы, логам.

Эта система всегда доступна для двух упомянутых выше пользователей, что позволяет им в любой момент получить нужную информацию и управлять процессами на складе.

### **3. Подготовка к работе**

#### **3.1 Порядок загрузки данных и программ**

Чтобы система управления беспилотными устройствами начала работу, нужно выполнить несколько шагов по установке и настройке программного обеспечения:

1. **Установка необходимой программы:** На компьютере, который будет использоваться как сервер (основной компьютер системы), должна быть установлена программа IntelliJ IDEA версии 2023.2.2 или новее. Это специальное программное обеспечение, которое помогает системе правильно работать и обрабатывать данные.

2. **Подключение к сетям:** Все компьютеры, которые будут использоваться для работы с системой, должны быть подключены к интернету и к внутренней сети предприятия. Это необходимо для того, чтобы они могли обмениваться информацией между собой и с сервером.

3. **Размещение программы на сервере:**

– **Локальная установка:** Если используется свой собственный сервер (это может быть специальный компьютер в офисе), программа устанавливается непосредственно на него. Это позволяет контролировать работу системы непосредственно с рабочего места.

– **Использование хостинга:** Если нет возможности использовать свой сервер, можно воспользоваться услугами сторонней компании, которая предоставит место на своих серверах для вашей программы. Это называется хостингом.

4. **Доступ к программе:** После установки и настройки, программа будет доступна для использования по специальной ссылке в браузере. Для системы эта ссылка будет выглядеть так: <https://localhost:8088/drones>. Это адрес, по которому сотрудники смогут входить в систему и начинать работу.

Эти шаги помогут правильно запустить систему управления беспилотными устройствами и обеспечат её стабильную и безопасную работу.

### **3.2. Порядок проверки работоспособности**

Чтобы убедиться в том, что система управления беспилотными устройствами работает и доступна для использования, можно выполнить следующие простые шаги:

#### **Запуск браузера Safari:**

Найдите иконку браузера Safari на рабочем столе вашего компьютера или в доке (нижняя панель приложений на Mac).

Щелкните по иконке Safari один раз, чтобы открыть браузер.



Рисунок 36 – логотип браузера Safari

#### **Ввод адреса системы:**

Когда браузер откроется, найдите адресную строку в верхней части окна. Это длинное белое поле, где обычно написаны адреса сайтов.



Введите в эту строку следующий адрес: <https://localhost:8088/drones> и нажмите клавишу Enter на клавиатуре.

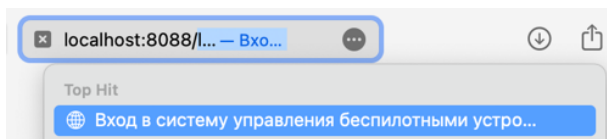


Рисунок 37 – ввод адреса

### **Авторизация в системе:**

После ввода адреса на экране появится страница для входа в систему. Здесь вам нужно будет ввести свой логин и пароль, которые вы должны получить от отдела IT компании.

После ввода данных нажмите кнопку «Войти».

A login form with two input fields. The first field is labeled 'Имя пользователя:' and contains the text 'user'. The second field is labeled 'Пароль:' and contains four dots. Below the fields is a blue button labeled 'Войти'.

Рисунок 38 – поля логина

### **Проверка доступности системы:**

Если после входа в систему перед вами открылась страница с интерактивной картой, где отображаются беспилотники и их местоположения, это значит, что система работает правильно.

Если вы видите карту и можете взаимодействовать с ней, значит, все в порядке.

В случае, если система не запускается или вы не можете войти, следует обратиться за помощью к технической поддержке вашей компании. Обычно для этого достаточно сообщить о проблеме в IT отдел.

## 4.Описание функций

В системе управления беспилотными устройствами доступны следующие функции, которыми могут пользоваться работники и администратор:

1. **Управление беспилотником вручную:**

– **Как активировать:** Откройте список всех беспилотников в системе. Это можно сделать, выбрав соответствующий пункт в меню.

– **Список беспилотников ▼**

Рисунок 39 – меню беспилотников

– **Что делать:** Выберите беспилотник, которым хотите управлять, нажав на его имя или номер в списке. После выбора используйте кнопки-стрелочки на клавиатуре для управления движением беспилотника.

**Список беспилотников ▼**

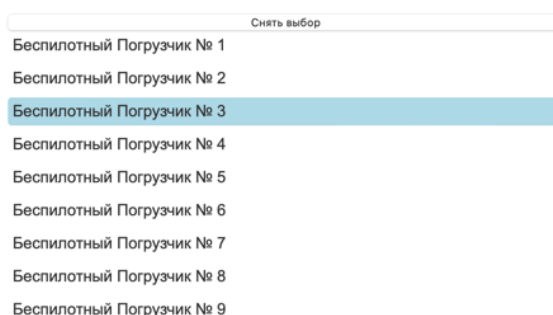


Рисунок 40 – раскрытый список беспилотников

– **Как завершить управление:** Для того чтобы перестать управлять беспилотником, нажмите кнопку, которая снимет выбор с этого беспилотника.



2. **Автоматическая отправка беспилотника за грузом:**

– **Как активировать:** Введите номер или название стеллажа в поле ввода, которое находится в системе управления.



Рисунок 41 – поле ввода стеллажа

– **Что делать:** После ввода номера стеллажа нажмите кнопку "Отправить беспилотник". Беспилотник автоматически поедет к указанному стеллажу, заберет нужный груз и доставит его до разгрузочной зоны, после чего вернется на зарядную станцию.

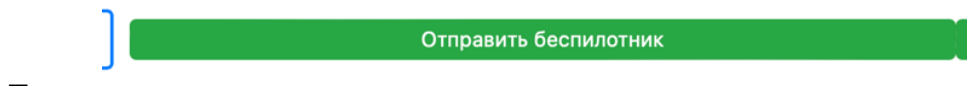


Рисунок 42 – кнопка отправки беспилотника

### 3. Создание отчета:

– **Как активировать:** Откройте список логов работы беспилотников, выбрав соответствующий пункт в меню.

**Лог событий ▼**

Рисунок 43 – кнопка лога событий

– **Что делать:** Просмотрите логи и нажмите кнопку "Создать отчет", чтобы система сформировала документ с данными о работе беспилотников за выбранный период.

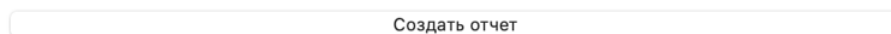


Рисунок 44 – кнопка создания отчета

### 4. Просмотр списка беспилотников:

– **Как активировать:** В главном меню системы выберите пункт "Список беспилотников".

**Список беспилотников ▼**

Рисунок 45 – кнопка списка беспилотников

– **Что делать:** Вам откроется список всех беспилотников с детальной информацией о каждом из них.

Беспилотный Погрузчик № 1  
Беспилотный Погрузчик № 2  
Беспилотный Погрузчик № 3  
Беспилотный Погрузчик № 4  
Беспилотный Погрузчик № 5  
Беспилотный Погрузчик № 6  
Беспилотный Погрузчик № 7  
Беспилотный Погрузчик № 8  
Беспилотный Погрузчик № 9

Рисунок 46 – список беспилотников

5. **Импорт списка стеллажей:**

- **Как активировать:** Нажмите кнопку "Импорт файла" в системе.

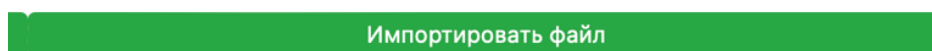


Рисунок 47 = кнопка импорта файлов

- **Что делать:** Выберите файл со списком стеллажей, который
- нужно загрузить в систему. После выбора файла нажмите "Открыть", чтобы файл был импортирован в систему.

6. **Аварийная остановка всех беспилотников:**

- **Как активировать и что делать:** В любой момент вы можете нажать кнопку "Аварийная остановка", которая немедленно остановит все беспилотники в случае необходимости.



Рисунок 48 – кнопка аварийной остановки

7. **Восстановление работы беспилотников после остановки:**

- **Как активировать и что делать:** После аварийной остановки, когда ситуация будет устранена, нажмите кнопку "Восстановление работы", чтобы все беспилотники возобновили свою деятельность.

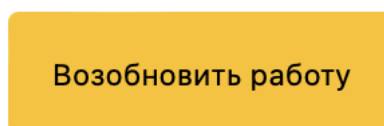


Рисунок 49 – кнопка возобновления работы

Эти функции сделаны максимально простыми и понятными, чтобы каждый сотрудник мог легко взаимодействовать с системой и эффективно использовать беспилотники в своей работе.

## **5. Аварийные ситуации**

При использовании системы управления беспилотными устройствами могут возникнуть различные технические неполадки. Ниже приведены наиболее распространенные проблемы и рекомендации по их решению:

### **1. Проблемы с интернет-соединением:**

- **Описание ошибки:** Нет доступа к интернету.
- **Возможные причины:** Проблема с вашим интернет-провайдером, отключен Wi-Fi или сетевой кабель.
- **Требуемые действия пользователя:** Проверьте, подключен ли ваш компьютер к интернету. Если используете Wi-Fi, убедитесь, что Wi-Fi включен и подключен к нужной сети. Попробуйте перезагрузить роутер. Если проблема не решается, обратитесь к вашему интернет-провайдеру или в техническую поддержку.

### **2. Сбой электропитания:**

- **Описание ошибки:** Неожиданное отключение электроэнергии.
- **Возможные причины:** Отключение электроэнергии в вашем районе или проблемы с электропроводкой.
- **Требуемые действия пользователя:** Если возможно, используйте источник бесперебойного питания (ИБП) для критически важных компьютеров. После восстановления электроснабжения перезагрузите ваш компьютер и проверьте, работает ли система корректно.

### **3. Проблемы с аутентификацией:**

- **Описание ошибки:** Ошибка при входе в систему.

- **Возможные причины:** Неверно введен логин или пароль.
- **Требуемые действия пользователя:** Убедитесь, что вы правильно вводите свои учетные данные. Если повторный ввод не помогает, обратитесь за помощью к администратору системы или к директору для восстановления доступа.

#### 4. **Сервер не найден:**

- **Описание ошибки:** Невозможно соединиться с сервером.
- **Возможные причины:** Проблемы с сетью, сервер временно недоступен или неправильно введен адрес.
- **Требуемые действия пользователя:** Проверьте, правильно ли вы ввели адрес системы в браузере. Попробуйте перезагрузить страницу. Если проблема сохраняется, проверьте ваше интернет-соединение или свяжитесь с технической поддержкой для проверки состояния сервера.

Эти рекомендации помогут вам быстро решить основные проблемы, которые могут возникнуть при работе с системой управления беспилотными устройствами. Если проблема не решается с помощью предложенных действий, всегда можно обратиться за помощью к технической поддержке вашей компании.

## **6. Рекомендации по освоению**

Для обеспечения эффективной работы с системой управления беспилотными устройствами предлагаем следующие рекомендации. Эти советы помогут вам лучше понять и быстрее освоить функционал системы, а также избежать распространенных ошибок в процессе её эксплуатации.

### **Описание контрольного примера**

Представим типичный сценарий использования системы: вы хотите отправить беспилотник за определенным грузом на склад и затем получить отчет о выполненной задаче.

#### 1. **Запуск системы:**

- Откройте браузер Safari и введите адрес системы <https://localhost:8088/drones> в адресную строку.

- Введите свои учетные данные для входа в систему.

## 2. Отправка беспилотника за грузом:

- Введите номер стеллажа, к которому должен подъехать беспилотник, в поле ввода.

- Нажмите кнопку "Отправить беспилотник".

## 3. Просмотр выполнения задачи:

- Обратите внимание на интерактивную карту.
- Убедитесь, что беспилотник успешно достиг стеллажа и возвращает груз на разгрузочную зону.

## 4. Создание отчета:

- Перейдите в раздел "Логи".
- Нажмите "Создать отчет" для получения подробного отчета о выполненной задаче.

## Правила запуска и выполнения

- **Подготовьте рабочее место:** Убедитесь, что ваш компьютер подключен к сети и что все необходимые программы установлены и функционируют корректно.

- **Проверьте свои доступы:** Перед началом работы убедитесь, что у вас есть права доступа для выполнения необходимых операций. Если у вас возникнут вопросы по доступам, обратитесь к администратору.

- **Следуйте инструкциям:** Внимательно следуйте инструкциям и рекомендациям, представленным в документации и на веб-сайте системы.

- **Практика:** Чем больше вы практикуетесь в использовании системы, тем более эффективным и уверенным пользователем вы станете. Не бойтесь экспериментировать с различными функциями в контролируемой среде.

Соблюдение этих рекомендаций позволит вам более продуктивно и безопасно использовать систему управления беспилотными устройствами, а также поможет избежать распространенных ошибок и недоразумений в процессе её эксплуатации.