Оглавление

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.
- Инициирует 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**) инициируется 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.