МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное автономное образовательное учреждение высшего образования

«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ АЭРОКОСМИЧЕСКОГО ПРИБОРОСТРОЕНИЯ»

КАФЕДРА КОМПЬЮТЕРНЫХ ТЕХНОЛОГИЙ И ПРОГРАММНОЙ ИНЖЕНЕРИИ

КУРСОВОЙ ПРОЕКТ

ЗАЩИЩЕН С ОЦЕНКОЙ РУКОВОДИТЕЛЬ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Старший преподаватель |  |  |  | Т. И. Белая |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |

|  |
| --- |
| ПОЯСНИТЕЛЬНАЯ ЗАПИСКА К КУРСОВОМУ ПРОЕКТУ |
| Разработка информационной системы управления беспилотными устройствами для охраны территорий |
| по курсу: Проектирование программных систем |

РАБОТУ ВЫПОЛНИЛ

|  |  |  |  |
| --- | --- | --- | --- |
| СТУДЕНТ ГР. № | 4133 |  | Ковалев Д.В |
|  | подпись, дата |  | инициалы, фамилия |

Санкт-Петербург 2024

## РЕФЕРАТ

Отчет 69 с., 18 рис., 2 прил.

АВТОМАТИЗИРОВАННАЯ СИСТЕМА УПРАВЛЕНИЯ И АНАЛИЗА ДАННЫХ, ОХРАНЫ ТЕРРИТОРИЙ, БАЗА ДАННЫХ, ФУНКЦИИ, МОДЕЛИРОВАНИЕ

Объектом исследования является процесс управления охранными мероприятиями, включающий функции управления, мониторинга и анализа данных с различных датчиков, установленных на БПЛА.

Цель работы – разработка автоматизированной информационной системы, которая будет оптимизировать деятельность охранных мероприятий на территориях.

В процессе работы было проведено исследование предметной области, которое позволило выяснить требования и границы информационной системы.

В результате работы была создана автоматизированная система для управления и анализа данных с БПЛА, обеспечивающая повышение эффективности и продуктивности охранных мероприятий.

# СОДЕРЖАНИЕ

[РЕФЕРАТ 2](#_Toc167780761)

[СОДЕРЖАНИЕ 3](#_Toc167780762)

[ВВЕДЕНИЕ 6](#_Toc167780763)

[1. ЦЕЛИ И НАЗНАЧЕНИЕ СОЗДАНИЯ АВТОМАТИЗИРОВАННОЙ 7](#_Toc167780764)

[1.1 Назначение программы 7](#_Toc167780765)

[1.2 Бизнес-функции, для автоматизации которых предназначена система 7](#_Toc167780766)

[2. ПРОЕКТИРОВАНИЕ И КОМПЛЕКСНОЕ МОДЕЛИРОВАНИЕ 8](#_Toc167780767)

[2.1 Характеристика и структура программы 8](#_Toc167780768)

[2.2 Описание основных особенностей программы 8](#_Toc167780769)

[2.3 Основные пользователи системы 9](#_Toc167780770)

[2.4 Функциональное моделирование в методике IDEF0 10](#_Toc167780771)

[2.4.1 Контекстная диаграмма 10](#_Toc167780772)

[2.4.2 Декомпозиция контекстной диаграммы 11](#_Toc167780773)

[2.4.3 Декомпозированная диаграмма второго уровня ветки A1 12](#_Toc167780774)

[2.4.4 Декомпозированная диаграмма второго уровня ветки A2 13](#_Toc167780775)

[2.4.5 Декомпозированная диаграмма второго уровня ветки A3 14](#_Toc167780776)

[2.5 Моделирование в методике DFD 15](#_Toc167780777)

[2.6 Объектное моделирование в методике UML 16](#_Toc167780778)

[2.6.1 Диаграмма вариантов использования 16](#_Toc167780779)

[2.6.2 Диаграмма компонентов 17](#_Toc167780780)

[2.6.3 Диаграмма пакетов 18](#_Toc167780781)

[2.6.4 Диаграмма размещения 19](#_Toc167780782)

[2.7 Проектирование базы данных 20](#_Toc167780783)

[2.7.1 Словарь данных 21](#_Toc167780784)

[3. ТРЕБОВАНИЯ К ИНФОРМАЦИОННОЙ СИСТЕМЕ 22](#_Toc167780785)

[3.1 Требования к структуре АС в целом 22](#_Toc167780786)

[3.2 Функциональные требования к системе 22](#_Toc167780787)

[3.2.1 Просмотр, редактирование и удаление полей и их датчиков 22](#_Toc167780788)

[3.2.2 Добавление данных с датчиков 23](#_Toc167780789)

[3.2.3 Просмотр и редактирование данных с датчиков 25](#_Toc167780790)

[3.2.4 Анализ данных 26](#_Toc167780791)

[4. РАЗРАБОТКА АВТОМАТИЗИРОВАННОЙ СИСТЕМЫ 27](#_Toc167780792)

[4.1 Базы данных 27](#_Toc167780793)

[4.2 Приложение сервер 27](#_Toc167780794)

[4.3 Приложение клиент 30](#_Toc167780795)

[4.4 Разработка интерфейса 31](#_Toc167780796)

[4.4.2 Добавление данных с датчиков 32](#_Toc167780797)

[4.4.3 Просмотр и редактирование данных с датчиков 33](#_Toc167780798)

[4.4.4 Анализ данных 34](#_Toc167780799)

[ЗАКЛЮЧЕНИЕ 35](#_Toc167780800)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 37](#_Toc167780801)

[ПРИЛОЖЕНИЕ А 38](#_Toc167780802)

[ПРИЛОЖЕНИЕ Б 54](#_Toc167780821)

## ВВЕДЕНИЕ

В современных условиях цифровизации различных отраслей экономики, необходимость в эффективных системах управления и анализа данных становится все более очевидной. Это касается и сферы охраны, где внедрение информационных систем позволяет значительно повысить продуктивность и эффективность охранных мероприятий. Курсовой проект посвящен разработке информационной системы управления и анализа данных с БПЛА, что является актуальным направлением в сфере охраны территорий.

Цель данного проекта – разработка автоматизированной информационной системы, которая позволит охранным агентствам эффективно собирать, хранить и анализировать данные, полученные с различных датчиков, установленных на БПЛА. Такая система обеспечит возможность ретроспективного мониторинга территорий, что позволит оперативно реагировать на изменения условий и принимать обоснованные решения по управлению охранными мероприятиями.

## ЦЕЛИ И НАЗНАЧЕНИЕ СОЗДАНИЯ АВТОМАТИЗИРОВАННОЙ

**СИСТЕМЫ**

## Назначение программы

Разрабатываемая информационная система предназначена для автоматизации процессов управления, мониторинга и анализа данных с БПЛА. Она позволяет получать данные с различных оптических источников, таких как камеры, датчики и хранить эти данные в централизованной базе данных. Анализ собранных данных позволяет принимать обоснованные решения по управлению охранными мероприятиями, такими как планирование, патрулирование и обнаружение опасностей.

## Бизнес-функции, для автоматизации которых предназначена система

Функциональность системы позволяет выполнять следующие операции:

* Хранение исторических данных: централизованное хранение данных о БПЛА и патрулях для последующего анализа и отчетности.
* Анализ данных: предоставление инструментов для анализа данных, таких как сводные таблицы, что позволяет выявлять тенденции и аномалии в изменениях на территориях.
* Управление БПЛА и секторами: добавление, редактирование и удаление информации о секторах и БПЛА на территориях через удобный интерфейс администратора.

## ПРОЕКТИРОВАНИЕ И КОМПЛЕКСНОЕ МОДЕЛИРОВАНИЕ

**СИСТЕМЫ**

## Характеристика и структура программы

Информационная система для управления и анализа данных с БПЛА была разработана с целью повышения эффективности управления охранных мероприятий за счет автоматизации управления, мониторинга и анализа данных, с применением различных БПЛА, развёрнутых на территориях.

Структура программы включает следующие основные компоненты:

* Декстопная часть: реализована на языке программирования С++ с использованием фреймворка QT. В декстопной части реализованы основные функции, отвечающие за обработку данных, взаимодействие с базой данных и предоставление функционала для интерфейса приложения.
* База данных: используется PostgreSQL 15, которая обеспечивает надежное хранение и быстрый доступ к данным. База данных структурирована таким образом, чтобы эффективно сохранять и обрабатывать данные системы, а также поддерживать систему авторизацию пользователей.
* GUI: веб-интерфейс, реализованный с использованием HTML, CSS и JavaScript. Клиентская часть обеспечивает удобный доступ к функционалу системы для пользователей.

## Описание основных особенностей программы

Основные особенности программы включают:

* + Монолитная архитектура: все компоненты системы объединены в единое приложение, что упрощает разработку и развертывание системы.
  + Централизованное хранение данных: все данные с сенсоров хранятся в единой базе данных MongoDB, что обеспечивает быстрый доступ и надежность хранения.
  + Инструменты анализа данных: система включает функционал для построения графиков и диаграмм, что позволяет пользователям проводить анализ собранных данных, выявлять тенденции и аномалии.
  + Управление полями и датчиками: пользователи системы могут добавлять, редактировать и удалять информацию о полях и установленных на них датчиках, что обеспечивает гибкость и адаптивность системы под конкретные нужды предприятия.

## Основные пользователи системы

Основными пользователями системы являются:

* Агрономы: используют систему для мониторинга состояния полей, анализа данных с сенсоров и принятия решений по управлению агротехнологическими процессами. Агрономы имеют доступ к данным, отчетам и аналитическим инструментам системы.
* Технический персонал: занимается установкой и обслуживанием датчиков на полях. Технический персонал использует систему для проверки состояния датчиков, замены и настройки оборудования.
* Руководители агропредприятия: используют систему для получения сводных отчетов о состоянии полей и эффективности агротехнологических мероприятий. Руководители могут принимать стратегические решения на основе данных, предоставляемых системой.

Система разработана таким образом, чтобы каждый пользователь имел доступ только к необходимым ему функциям, что обеспечивает безопасность и удобство использования.

## Функциональное моделирование в методике IDEF0

IDEF0 используется для создания функциональной модели, отображающей структуру и функции системы, а также потоки информации и материальных объектов, связывающие эти функции

## Контекстная диаграмма

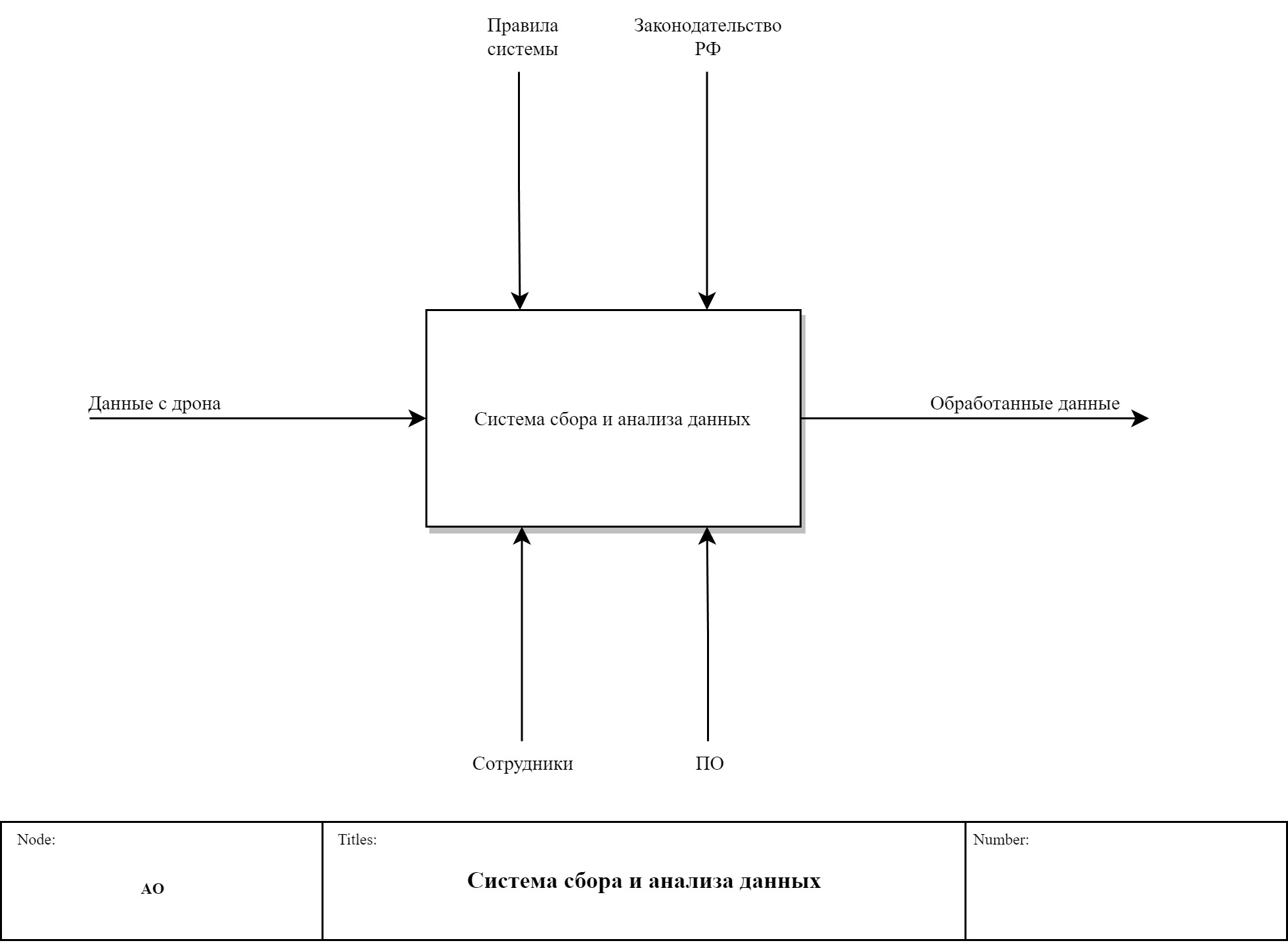


Рисунок 1 - Контекстная диаграмма

## Декомпозиция контекстной диаграммы

Декомпозиция контекстной диаграммы в методологии IDEF0 — это процесс детализации и разделения высокого уровня представления системы на более мелкие и управляемые части. Контекстная диаграмма (или A0 диаграмма) в IDEF0 показывает основную функцию системы, взаимодействие системы с внешними объектами (входы, выходы, механизмы и управления), но не вдаётся в детали внутренней структуры системы. В данном случае мы декомпозировали систему на 3 основных задачи.

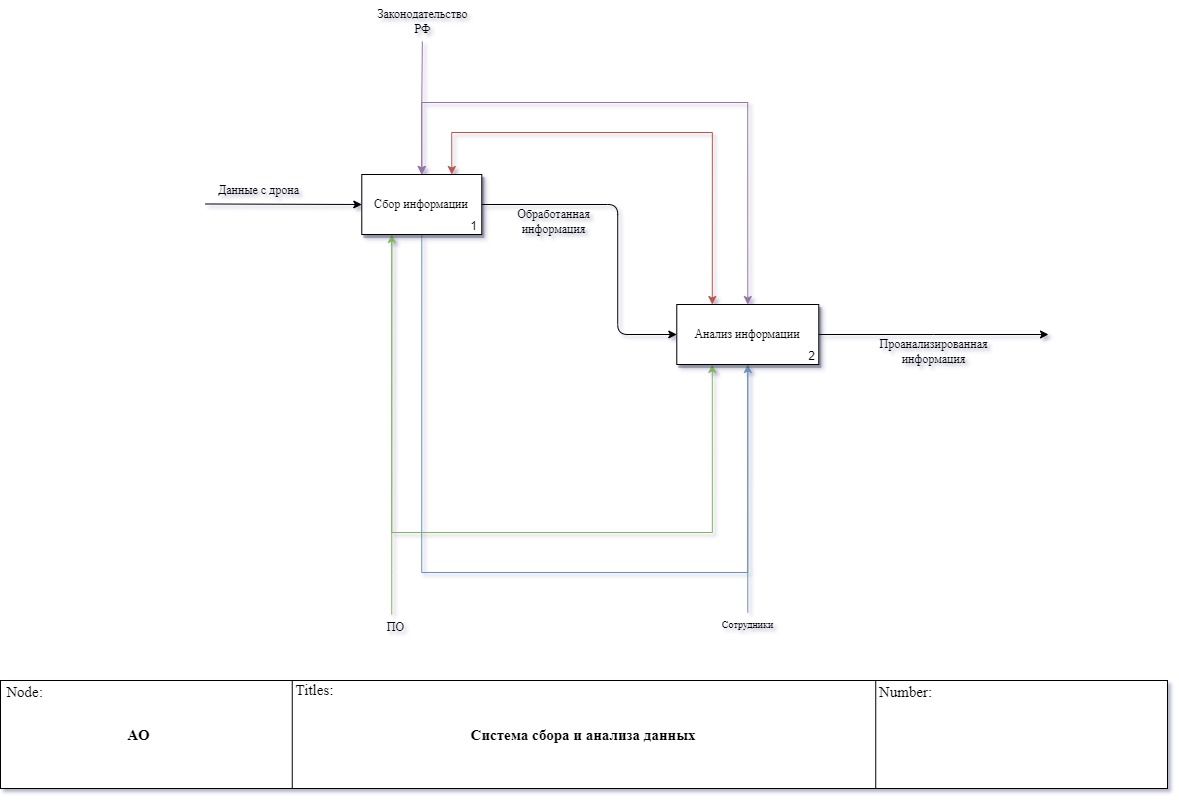


Рисунок 2 – Декомпозированная контекстная диаграмма

## Декомпозированная диаграмма второго уровня ветки A1

Декомпозиция задачи «сбор информации» на 3 подзадачи.

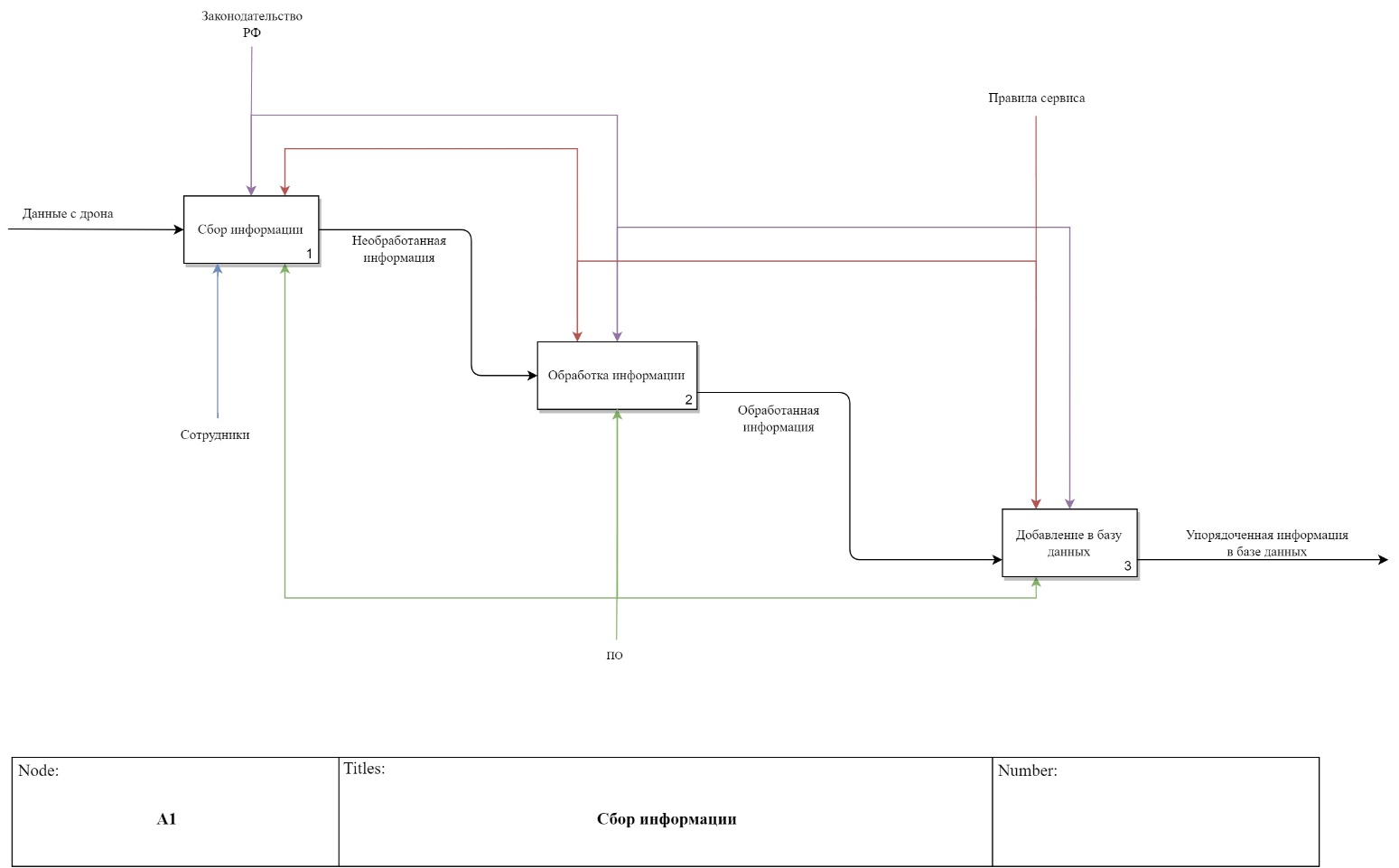


Рисунок 3 - Декомпозированная диаграмма второго уровня ветки A1

## Декомпозированная диаграмма второго уровня ветки A2

Декомпозиция задачи «анализ информации» на 2 подзадачи.

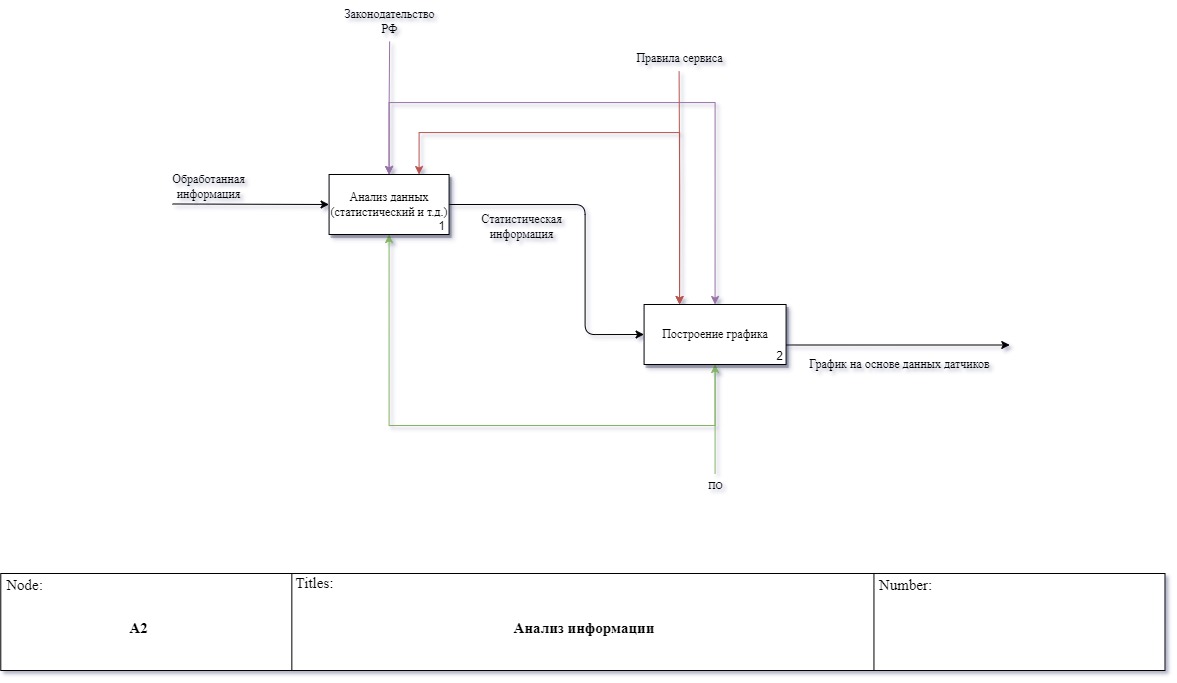


Рисунок 4 - Декомпозированная диаграмма второго уровня ветки A1

## Моделирование в методике DFD

DFD диаграммы в отличии 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.

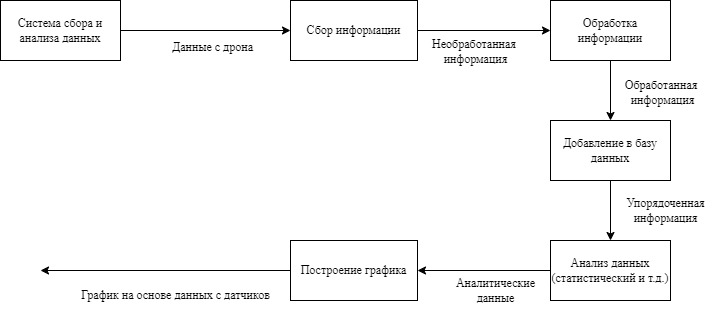


Рисунок 5 – Диаграмма потоковых данных

## Объектное моделирование в методике UML

## 2.6.1 Диаграмма вариантов использования

Диаграмма наглядно демонстрирует, как различные пользователи и системы взаимодействуют с системой, что помогает понять общую функциональность системы и выявить основные сценарии использования для дальнейшего проектирования и тестирования.

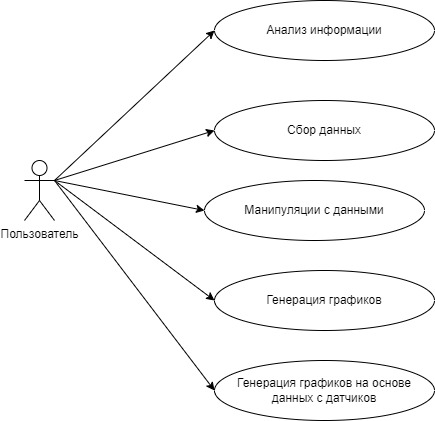


Рисунок 6 - Диаграмма вариантов использования

## 2.6.2 Диаграмма компонентов

Диаграмма компонентов UML используется для визуализации и понимания структуры программной системы на уровне высокоуровневых компонентов и их взаимосвязей. Она отображает различные программные компоненты, такие как модули, библиотеки, и их интерфейсы, а также зависимости между ними. Эта диаграмма помогает определить, как различные части системы взаимодействуют друг с другом, выявить потенциальные проблемы с интеграцией и зависимостями, и спланировать модульность и повторное использование компонентов. Благодаря диаграммам компонентов можно эффективно управлять сложностью системы, улучшать её модульность и способствовать ясному разделению ответственности между различными частями проекта.

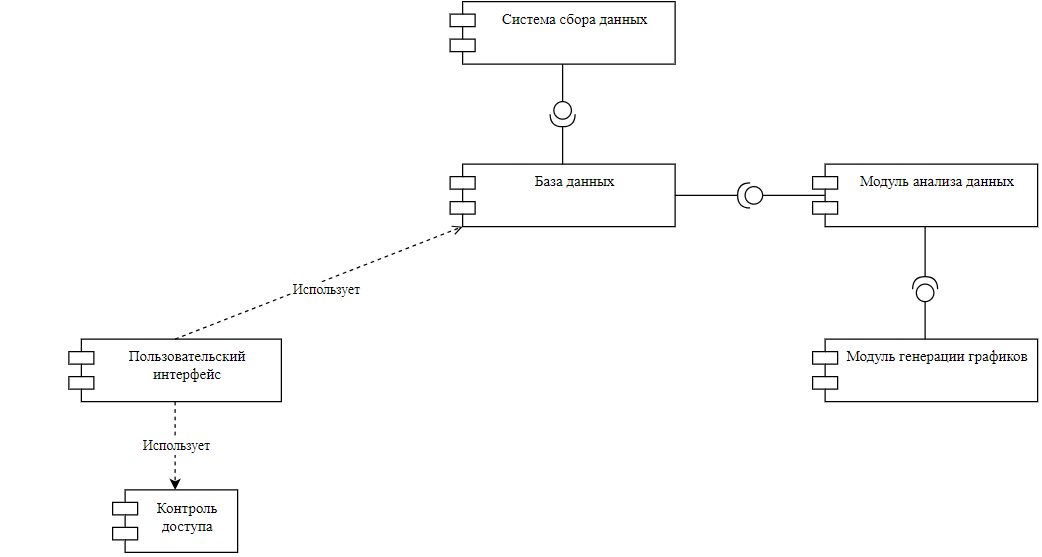


Рисунок 7 - Диаграмма компонентов

## 2.6.3 Диаграмма пакетов

Диаграмма пакетов UML используется для организации и группировки различных элементов модели, в логически связанные группы, облегчая понимание и управление сложными системами. Эта диаграмма помогает визуализировать структуру системы на более высоком уровне абстракции, показывая, как пакеты взаимодействуют друг с другом через зависимости и отношения.

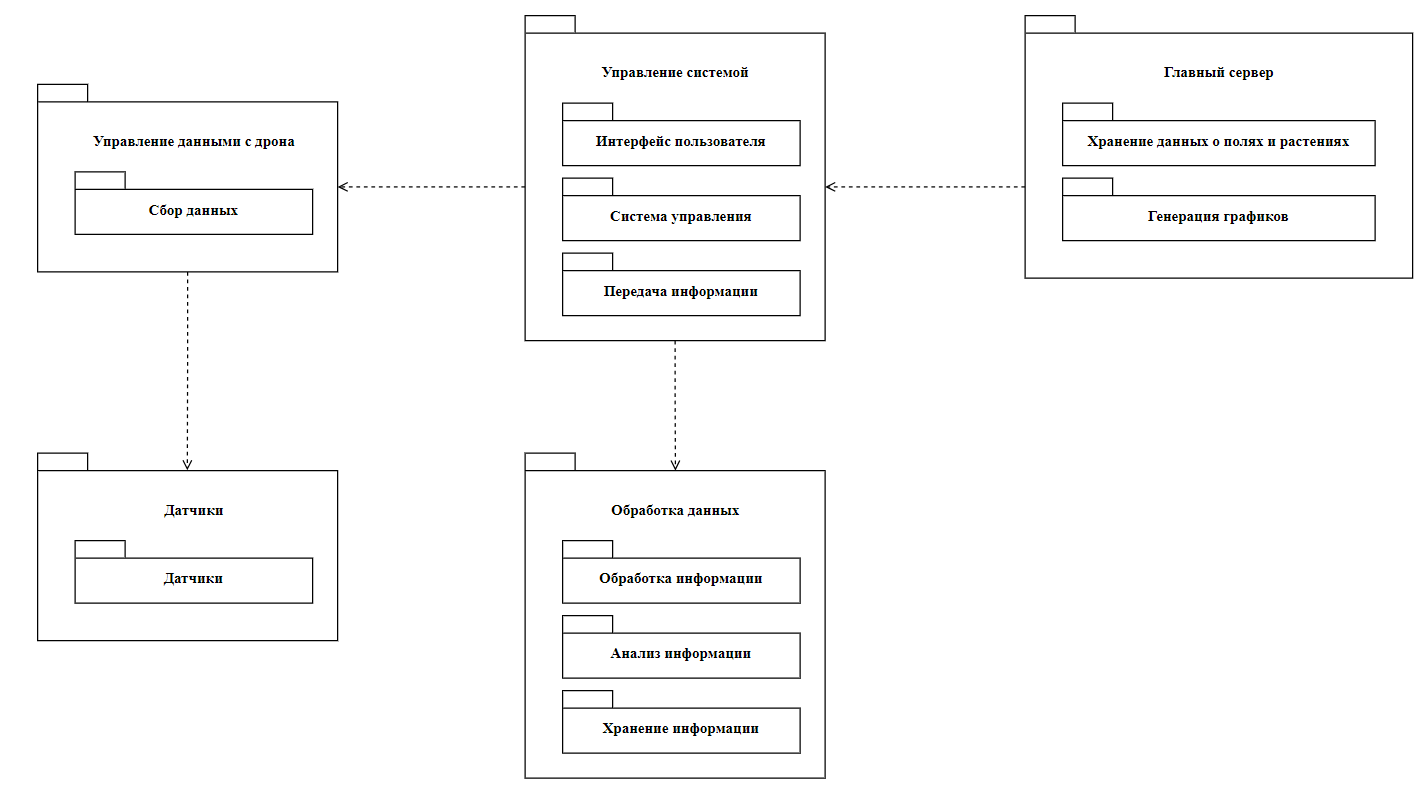


Рисунок 8 - Диаграмма пакетов

## 2.6.4 Диаграмма размещения

Диаграмма размещения UML используется для моделирования физического развёртывания артефактов программного обеспечения на узлах аппаратного обеспечения. Она показывает, как программные компоненты и сервисы распределены по физическим устройствам, таким как серверы, компьютеры и другие аппаратные узлы, а также описывает связи и взаимодействия между ними. Эта диаграмма полезна для визуализации и анализа архитектуры системы с точки зрения её физической реализации, что помогает в планировании развертывания, оценке производительности, надежности и масштабируемости системы.

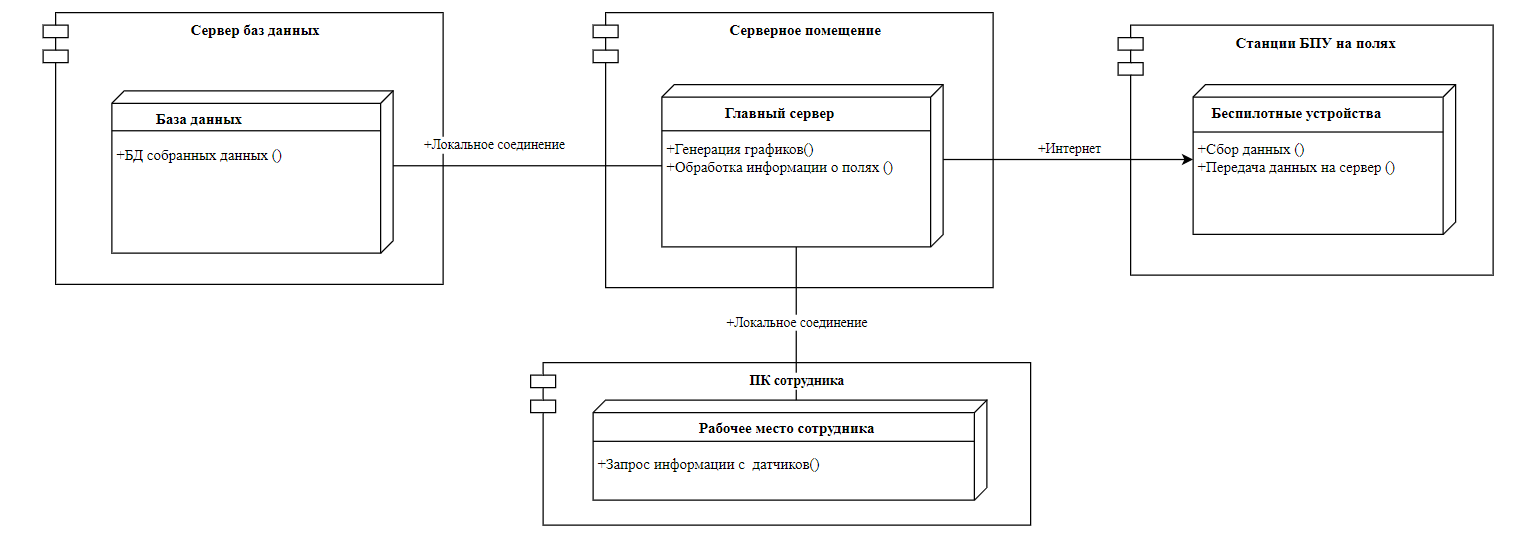


Рисунок 9 - Диаграмма размещения

## Проектирование базы данных

**2.7.1 Логическая модель данных**

Логическая модель базы данных используется для представления структуры данных системы на концептуальном уровне, независимом от конкретной СУБД. Логическая модель помогает четко определить требования к данным и их структуру, обеспечить целостность и нормализацию данных, а также устранить избыточность.

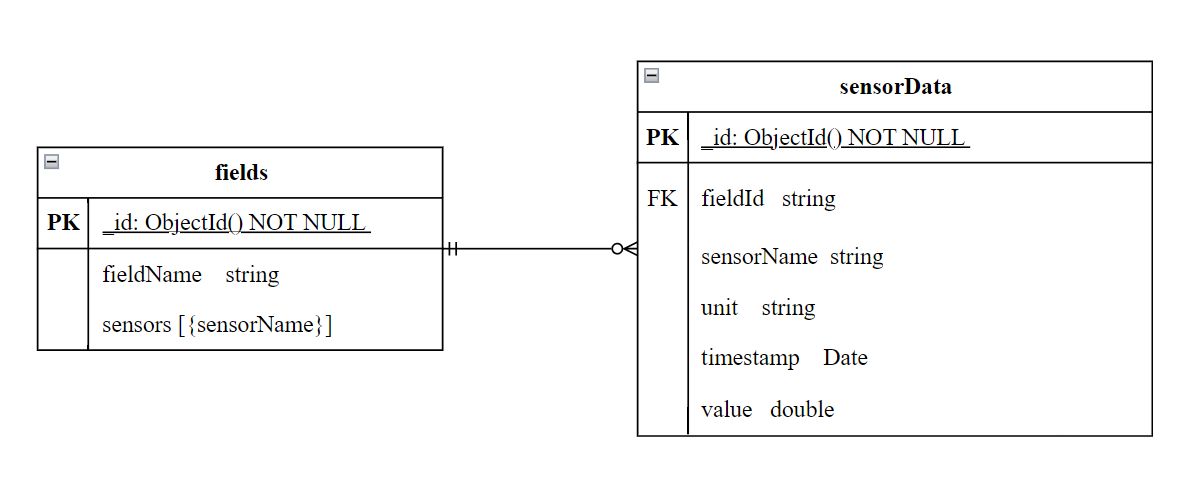


Рисунок 10 - Модель базы данных

## 2.7.1 Словарь данных

|  |  |
| --- | --- |
| **Наименование** | **Описание** |
| id | Текстовый тип, длина до 255 символов. Уникальный идентификатор поля. |
| fieldName | Текстовый тип, длина до 255 символов. Название поля. |
| sensors | Список объектов типа Sensor, каждый из которых содержит название датчика. |
| id | Текстовый тип, длина до 255 символов. Уникальный идентификатор записи данных. |
| fieldId | Текстовый тип, длина до 255 символов. Идентификатор поля. |
| sensorName | Текстовый тип, длина до 255 символов. Название датчика. |
| unit | Текстовый тип, длина до 255 символов. Единица измерения. |
| timestamp | Тип дата, формат день, месяц, год, арабскими цифрами, разделенными точкой — 01.01.2020. |
| value | Числовой тип, до 10 цифр. Значение, собранное датчиком. |

## ТРЕБОВАНИЯ К ИНФОРМАЦИОННОЙ СИСТЕМЕ

## Требования к структуре АС в целом

Информационная система для хранения и анализа данных с полей должна обеспечивать эффективное и надежное выполнение всех необходимых функций. В данном разделе описаны основные требования к структуре системы, необходимой для ее корректного функционирования.

## Функциональные требования к системе

## Просмотр, редактирование и удаление полей и их датчиков

Пользователь должен иметь возможность просматривать информацию о существующих полях и их датчиков, добавлять новые поля с датчиками, редактировать данные существующих полей и датчиков и удалять ненужные записи. На веб-странице должна быть представлена таблица с перечнем всех полей с их датчиками. Для добавления нового поля пользователь вводит название поля и название датчиков и нажимает кнопку "Добавить поле". Для редактирования существующего поля пользователь выбирает нужное поле в таблице и нажимает кнопку "Редактировать", после чего открывается форма с текущими данными, которые можно изменить. Для удаления поля пользователь выбирает его в таблице и нажимает кнопку "Удалить", после чего система запрашивает подтверждение действия и, при согласии, удаляет запись из базы данных.

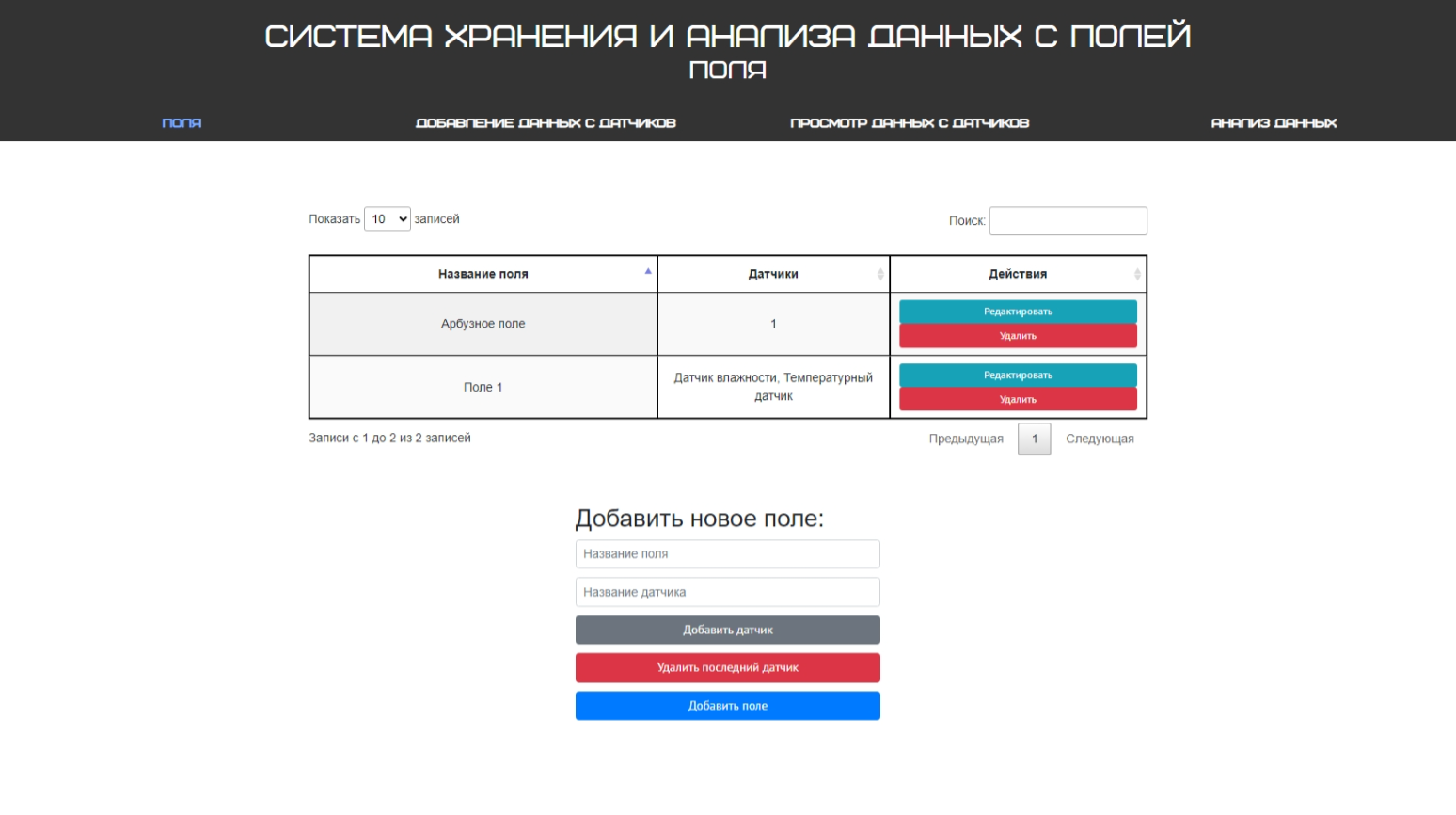


Рисунок 11 - Страница с полями и датчиками

## Добавление данных с датчиков

Пользователь должен иметь возможность вручную добавлять данные, полученные с датчиков, в систему. На соответствующей веб-странице предоставляется форма для ввода данных, где пользователь выбирает поле, выбирает название датчика, вводит единицу измерения, вводит временную метку и вводит значение. После заполнения всех необходимых полей и нажатия кнопки "Добавить данные", данные сохраняются в базе данных. Также присутствует возможность добавления данных с базу данных с помощью загрузки JSON файла. Для этого нужно нажать на кнопку «Выбор файла», выбрать файл, а после нажать кнопку «Загрузить из файла». Также присутствует возможность добавления данных с помощью специального поля на странице. В нем можно написать данные в формате JSON и после нажать кнопку «Загрузить».

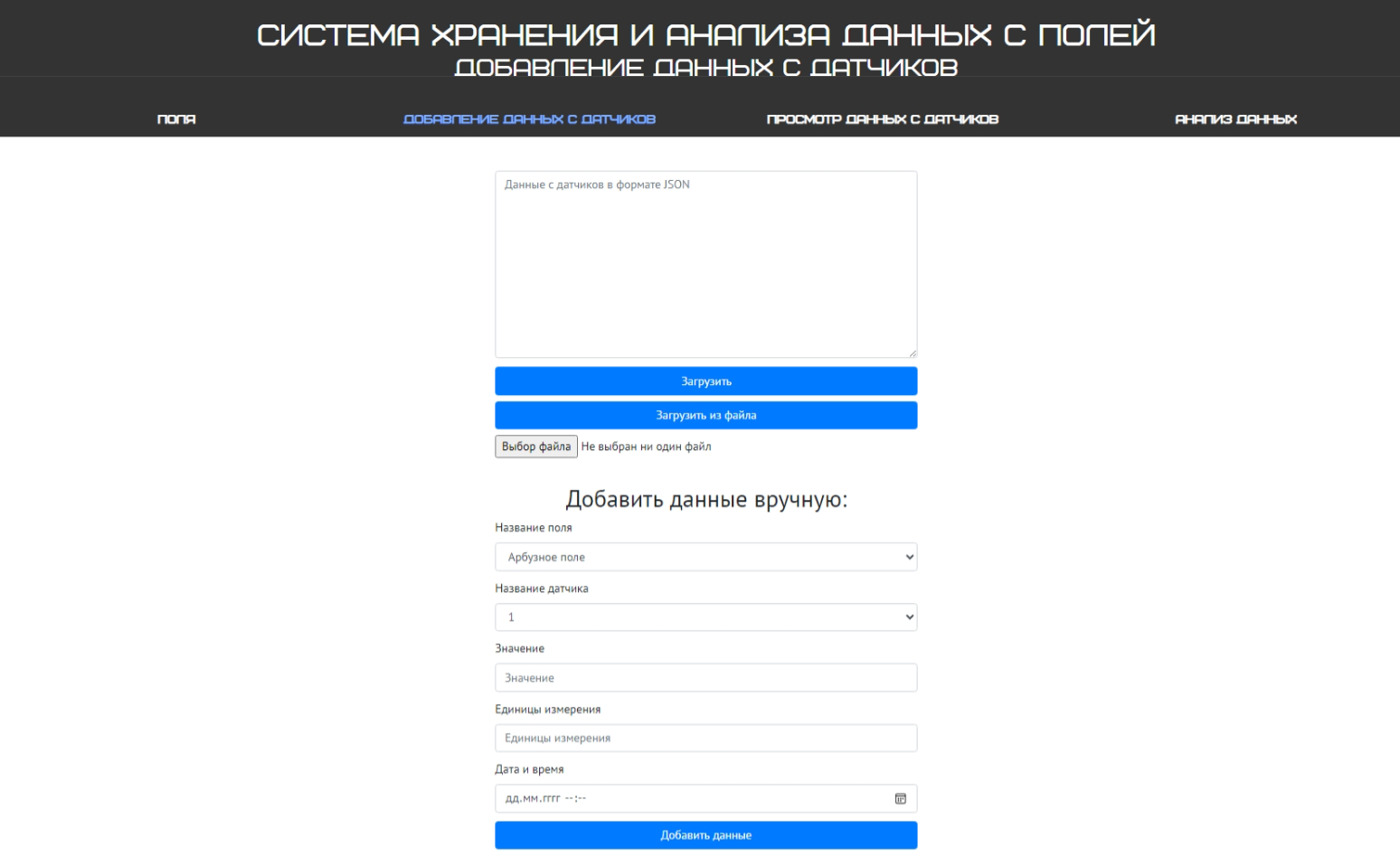


Рисунок 12 - Страница с добавлением данных с датчиков

## Просмотр и редактирование данных с датчиков

Пользователь должен иметь возможность просматривать данные, собранные с датчиков, в удобном формате. На веб-странице отображается таблица с данными, где каждый ряд представляет собой запись с датчика. Пользователь может использовать фильтры для поиска данных по полю, датчику и временной метке. Это позволяет быстро находить и анализировать необходимые данные. Также возможно сортировать данные по различным критериям. Для редактирования существующей записи пользователь выбирает нужное поле в таблице и нажимает кнопку "Редактировать", после чего открывается форма с текущими данными, которые можно изменить. Для удаления записи, пользователь выбирает ее в таблице и нажимает кнопку "Удалить", после чего система запрашивает подтверждение действия и, при согласии, удаляет запись из базы данных.

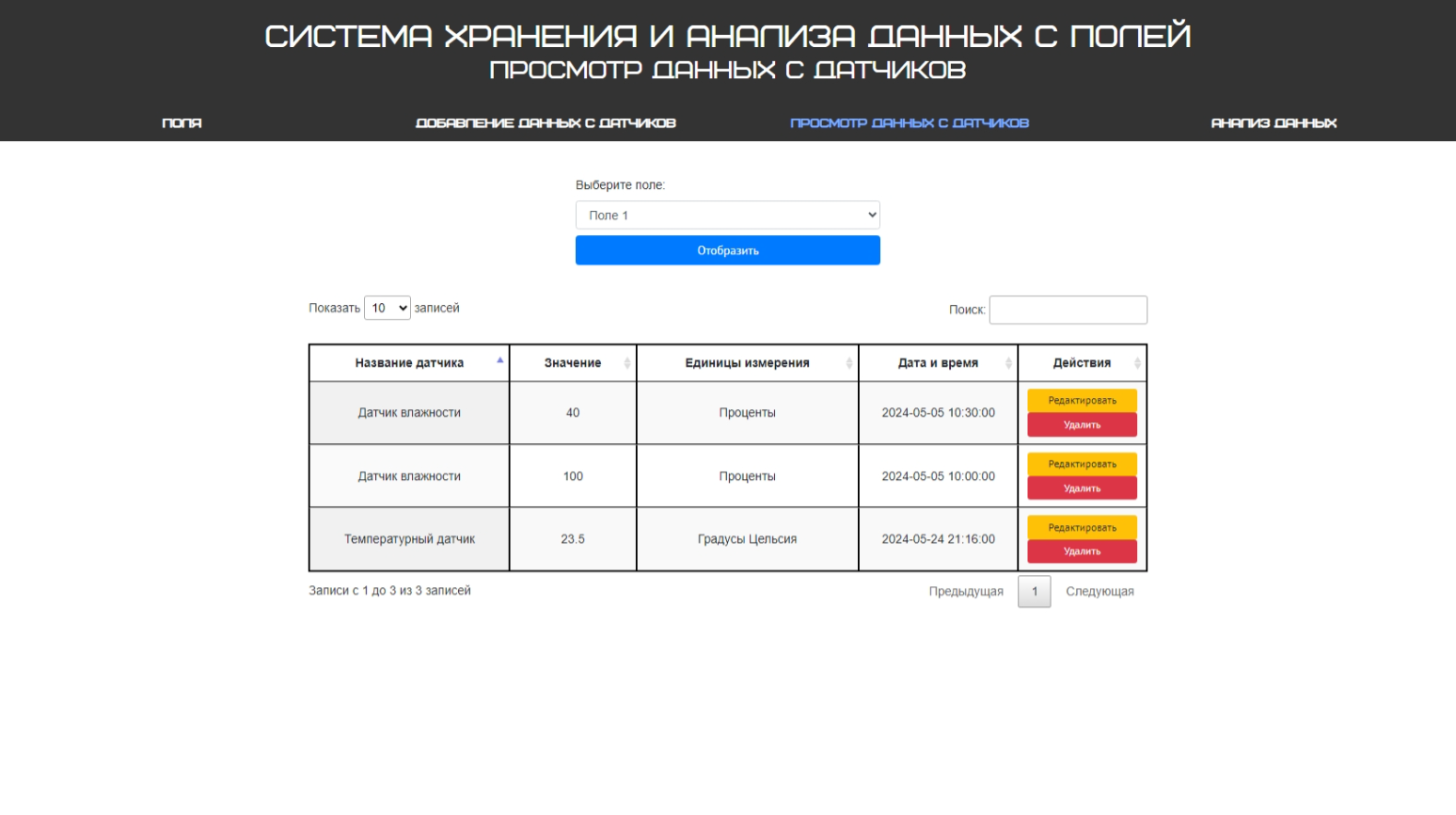


Рисунок 13 - Страница с просмотром данных с датчиков

## Анализ данных

Пользователь должен иметь возможность создавать графики на основе данных с датчиков. На веб-странице отображаются выпадающие списки выбора поля и датчика. Пользователь выбирает нужное поле, датчик, дату начала и дату окончания. После нажимает кнопку «Создать график» и на странице появляется график. Также есть возможность отображения графика за все время на основе всех данных от выбранного датчика. Для вывода такого графика нужно выбрать поле и датчик, а затем нажать на кнопку «График за все время».

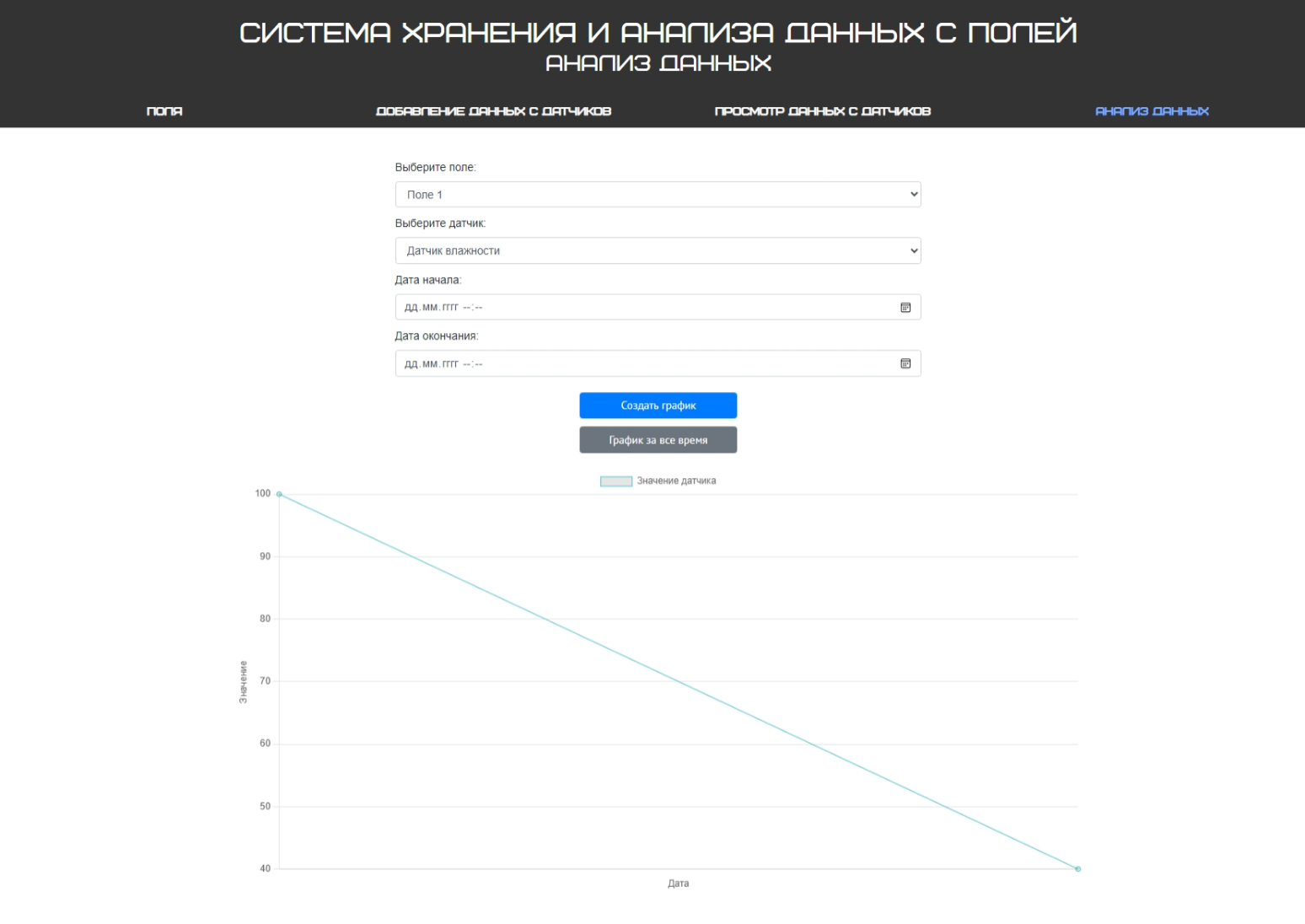


Рисунок 14 - Страница с анализом данных

## РАЗРАБОТКА АВТОМАТИЗИРОВАННОЙ СИСТЕМЫ

## Базы данных

Информационная система для хранения и анализа данных с полей включает в себя базу данных MongoDB. База данных содержит следующие основные коллекции: fields, sensors и sensorData. Каждая коллекция имеет четко определенные поля и связи, что обеспечивает целостность и структурированность данных. Коллекция fields хранит информацию о полях, включая их уникальный идентификатор и название. Коллекция sensors содержит данные о датчиках, установленных на полях, такие как название датчика и идентификатор соответствующего поля. Коллекция sensorData хранит значения, собранные с датчиков, включая временную метку, единицу измерения и значение, а также идентификаторы соответствующих полей и датчиков.

## Приложение сервер

Серверная часть системы реализована на языке программирования Java с использованием фреймворка Spring Boot. Сервер обрабатывает запросы от клиентов, взаимодействует с базой данных и отправляет обновленные данные клиентам. Основные компоненты серверной части включают контроллеры для обработки HTTP-запросов, сервисы для выполнения бизнес-логики и репозитории для взаимодействия с базой данных MongoDB.

**Описание класса MongoConfig**

Класс MongoConfig содержит настройки для подключения к базе данных MongoDB. Он включает конфигурацию базы данных, настройку конвертеров для преобразования данных и регистрацию необходимых бинов.

**Описание класса WebConfig**

Класс WebConfig конфигурирует параметры CORS для обеспечения взаимодействия клиентской части с серверной. Он включает методы для настройки разрешенных источников и методов HTTP-запросов.

**Описание класса FieldController**

Класс FieldController отвечает за обработку запросов, связанных с полями. Он включает методы для получения списка всех полей, получения конкретного поля по ID, создания нового поля, обновления существующего поля и удаления поля.

**Описание класса PageController**

Класс PageController отвечает за маршрутизацию запросов к HTML-страницам. Он включает методы для отображения страниц управления полями, данных с датчиков и других страниц пользовательского интерфейса.

**Описание класса SensorDataController**

Класс SensorDataController управляет данными, полученными с датчиков. Он включает методы для добавления новых данных, получения данных по полю и датчику, обновления и удаления данных.

**Описание класса GlobalExceptionHandler**

Класс GlobalExceptionHandler обрабатывает исключения, возникающие в приложении. Он включает методы для перехвата и обработки глобальных исключений, что позволяет улучшить стабильность и отладку системы.

**Описание класса Field**

Класс Field представляет модель данных поля. Он включает поля для хранения идентификатора, названия поля и списка датчиков, а также методы для получения и установки этих значений.

**Описание класса Sensor**

Класс Sensor представляет модель данных датчика. Он включает поле для хранения названия датчика и методы для получения и установки этого значения.

**Описание класса SensorData**

Класс SensorData представляет модель данных, собранных с датчиков. Он включает поля для хранения идентификатора, идентификатора поля, названия датчика, единицы измерения, временной метки и значения, а также методы для получения и установки этих значений.

**Описание интерфейса FieldRepository**

Интерфейс FieldRepository расширяет интерфейс MongoRepository и предоставляет методы для взаимодействия с коллекцией fields в базе данных MongoDB.

**Описание интерфейса SensorDataRepository**

Интерфейс SensorDataRepository расширяет интерфейс MongoRepository и предоставляет методы для взаимодействия с коллекцией sensorData в базе данных MongoDB.

**Описание класса SensorDataService**

Класс SensorDataService содержит бизнес-логику для работы с данными с датчиков. Он включает методы для сохранения данных, поиска данных по полю и датчику, обновления и удаления данных, а также экспорта данных.

**Описание класса DemoApplication**

Класс DemoApplication является точкой входа в приложение Spring Boot. Он содержит метод main, который запускает приложение.

## Приложение клиент

Клиентская часть приложения представляет собой веб-интерфейс, реализованный с использованием HTML, CSS и JavaScript. Веб-интерфейс предоставляет средства для управления полями, датчиками и данными, а также для анализа собранных данных.

На странице управления полями отображается таблица со списком всех полей и их датчиков. Пользователь может добавить новое поле, заполнив форму и нажав кнопку "Создать поле". Для редактирования существующего поля и его датчиков необходимо выбрать поле из списка, изменить данные в форме и нажать кнопку "Сохранить". Кнопка "Удалить" позволяет удалить выбранное поле из базы данных.

Страница управления данными содержит таблицу с данными, собранными с датчиков, включая временные метки, значения и единицы измерения. Пользователь может вручную добавить новые данные, заполнив форму и нажав кнопку "Создать". Для редактирования данных необходимо выбрать запись из списка, внести изменения в форме и нажать кнопку "Сохранить". Кнопка "Удалить" позволяет удалить выбранные данные из базы данных.

Страница анализа данных предоставляет инструменты для визуализации и анализа собранных данных. На странице доступен выбор параметров для построения графиков и диаграмм. Пользователь может выбрать поле, датчик и период времени для анализа. После выбора параметров и нажатия кнопки "Построить график" система отображает визуальный отчет, который помогает пользователю оценить тенденции и аномалии в данных.

## Разработка интерфейса

* + 1. **Просмотр, редактирование и удаление полей и их датчиков**

Пользователь должен иметь возможность просматривать информацию о существующих полях и их датчиков, добавлять новые поля с датчиками, редактировать данные существующих полей и датчиков и удалять ненужные записи. На веб-странице должна быть представлена таблица с перечнем всех полей с их датчиками. Для добавления нового поля пользователь вводит название поля и название датчиков и нажимает кнопку "Добавить поле". Для редактирования существующего поля пользователь выбирает нужное поле в таблице и нажимает кнопку "Редактировать", после чего открывается форма с текущими данными, которые можно изменить. Для удаления поля пользователь выбирает его в таблице и нажимает кнопку "Удалить", после чего система запрашивает подтверждение действия и, при согласии, удаляет запись из базы данных.

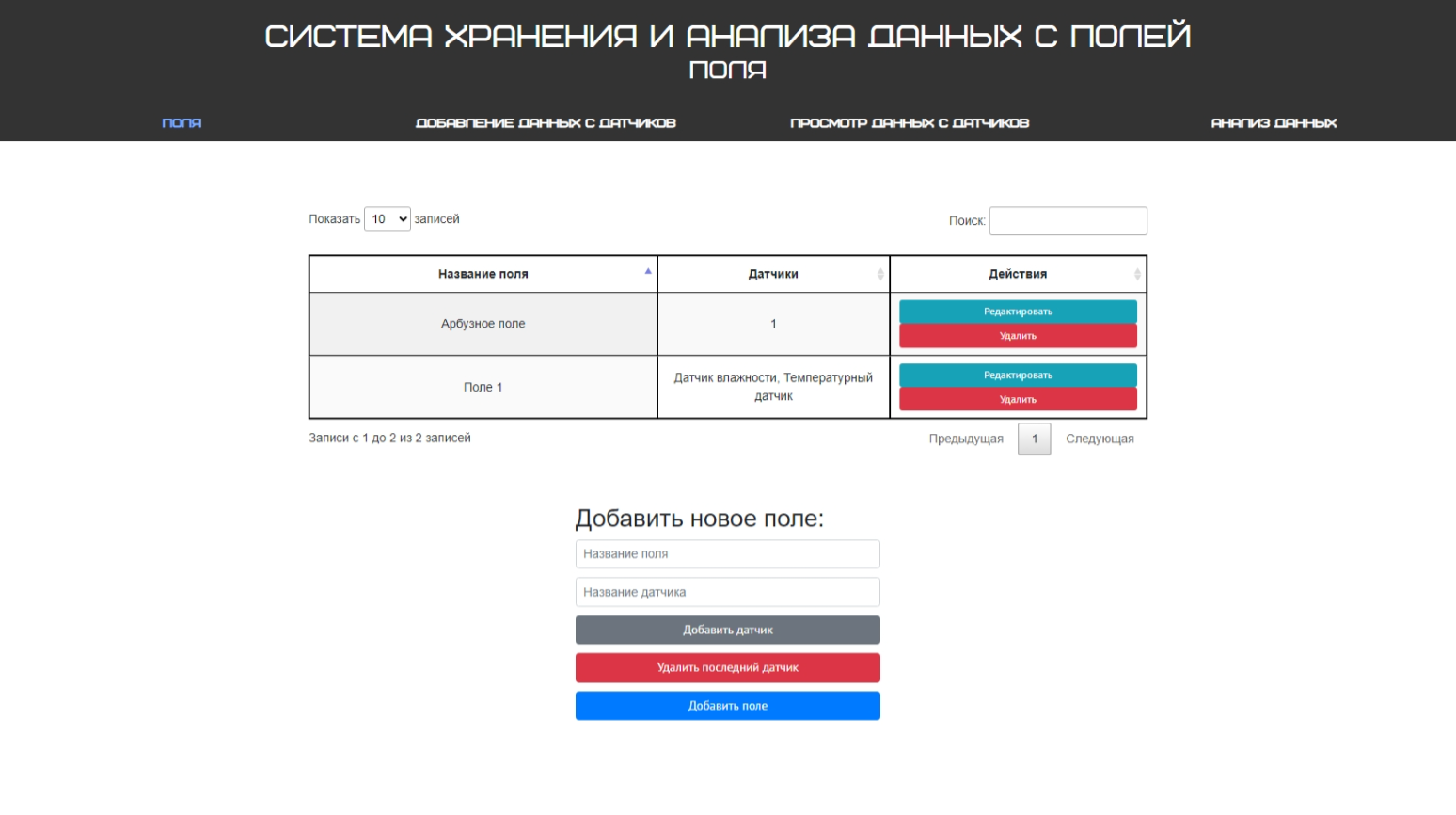


Рисунок 15 - Страница с полями и датчиками

## 4.4.2 Добавление данных с датчиков

Пользователь должен иметь возможность вручную добавлять данные, полученные с датчиков, в систему. На соответствующей веб-странице предоставляется форма для ввода данных, где пользователь выбирает поле, выбирает название датчика, вводит единицу измерения, вводит временную метку и вводит значение. После заполнения всех необходимых полей и нажатия кнопки "Добавить данные", данные сохраняются в базе данных. Также присутствует возможность добавления данных с базу данных с помощью загрузки JSON файла. Для этого нужно нажать на кнопку «Выбор файла», выбрать файл, а после нажать кнопку «Загрузить из файла». Также присутствует возможность добавления данных с помощью специального поля на странице. В нем можно написать данные в формате JSON и после нажать кнопку «Загрузить».

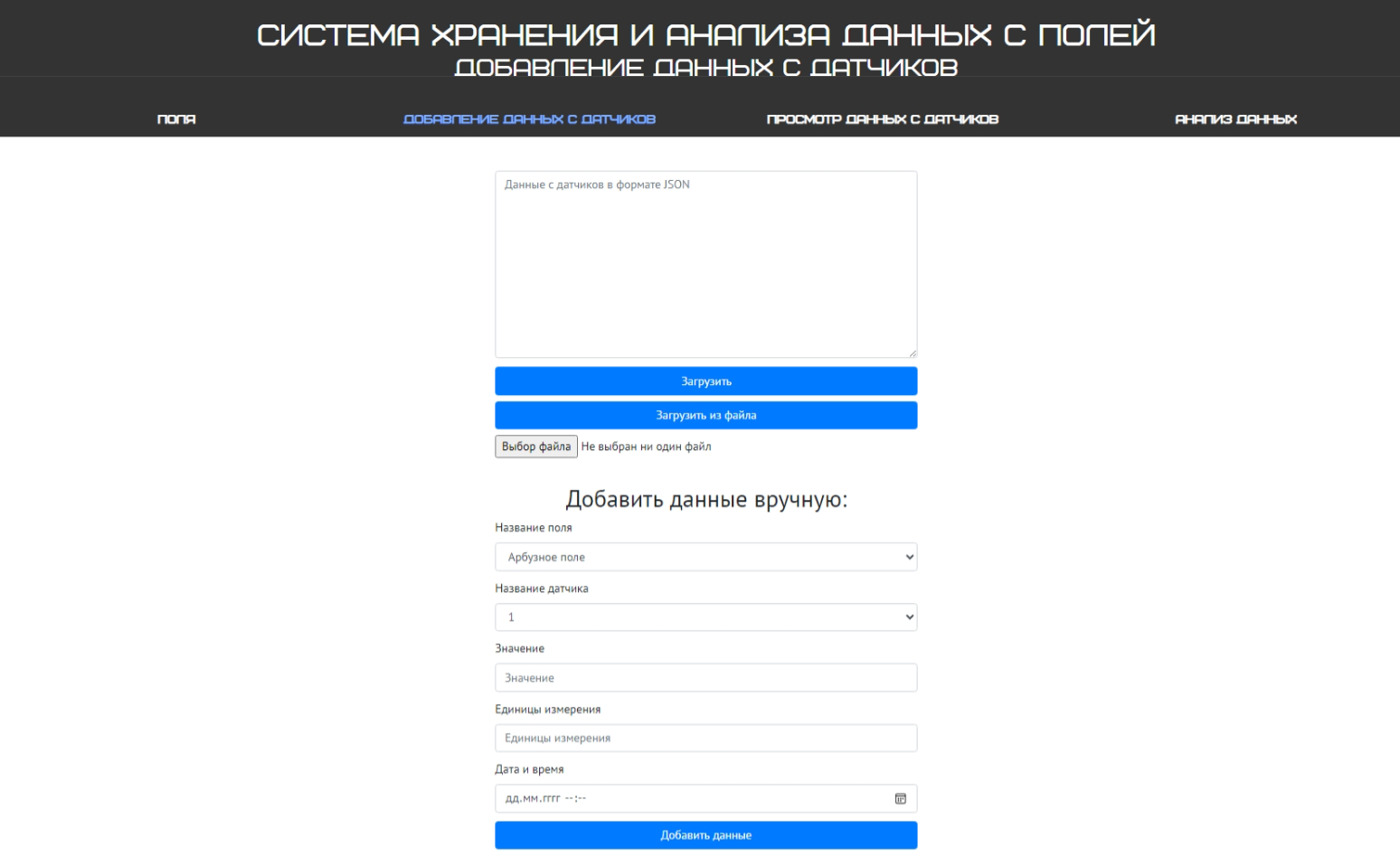


Рисунок 16 - Страница с добавлением данных с датчиков

## 4.4.3 Просмотр и редактирование данных с датчиков

Пользователь должен иметь возможность просматривать данные, собранные с датчиков, в удобном формате. На веб-странице отображается таблица с данными, где каждый ряд представляет собой запись с датчика. Пользователь может использовать фильтры для поиска данных по полю, датчику и временной метке. Это позволяет быстро находить и анализировать необходимые данные. Также возможно сортировать данные по различным критериям. Для редактирования существующей записи пользователь выбирает нужное поле в таблице и нажимает кнопку "Редактировать", после чего открывается форма с текущими данными, которые можно изменить. Для удаления записи, пользователь выбирает ее в таблице и нажимает кнопку "Удалить", после чего система запрашивает подтверждение действия и, при согласии, удаляет запись из базы данных.

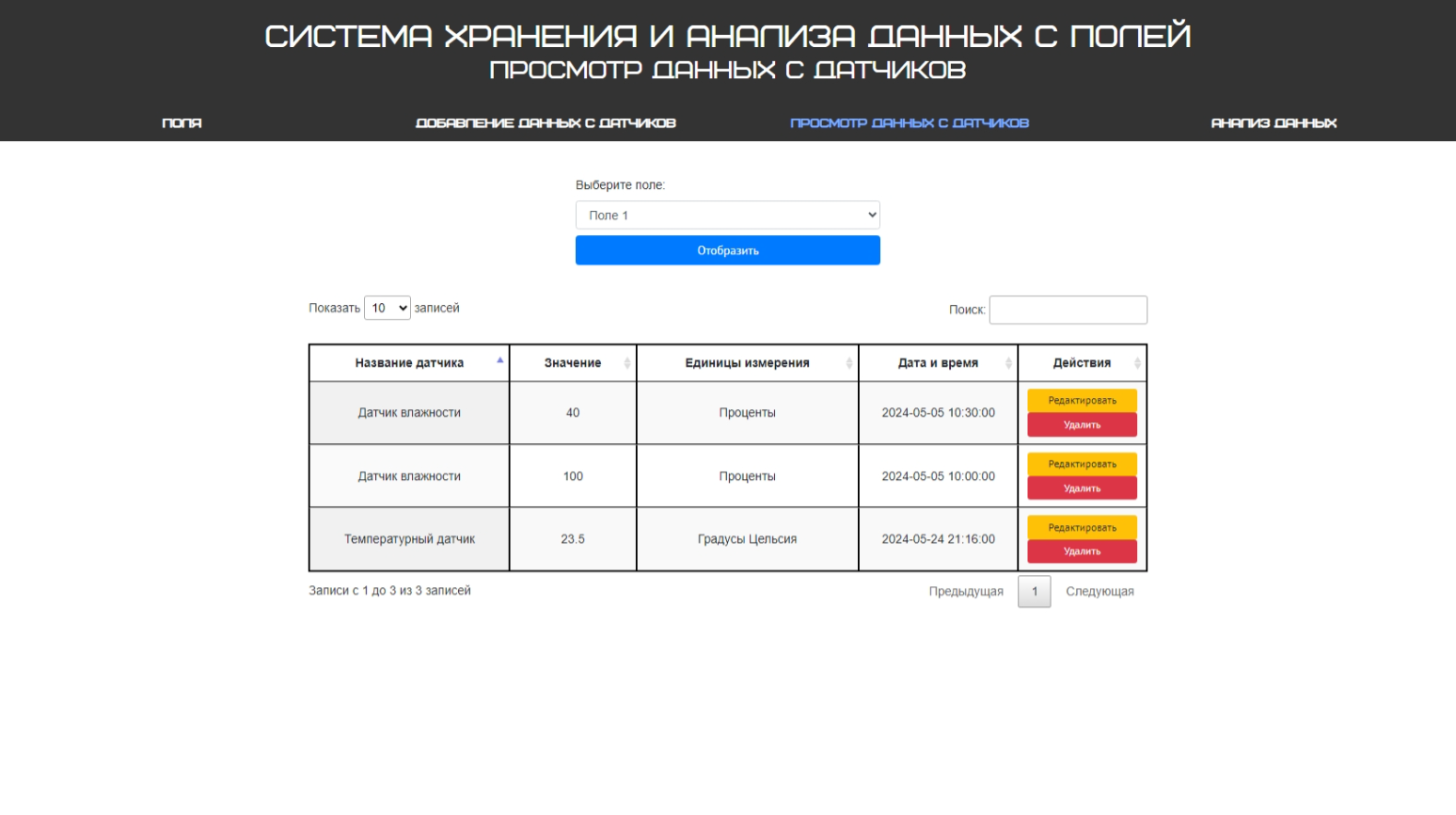


Рисунок 17 - Страница с просмотром данных с датчиков

## 4.4.4 Анализ данных

Пользователь должен иметь возможность создавать графики на основе данных с датчиков. На веб-странице отображаются выпадающие списки выбора поля и датчика. Пользователь выбирает нужное поле, датчик, дату начала и дату окончания. После нажимает кнопку «Создать график» и на странице появляется график. Также есть возможность отображения графика за все время на основе всех данных от выбранного датчика. Для вывода такого графика нужно выбрать поле и датчик, а затем нажать на кнопку «График за все время».

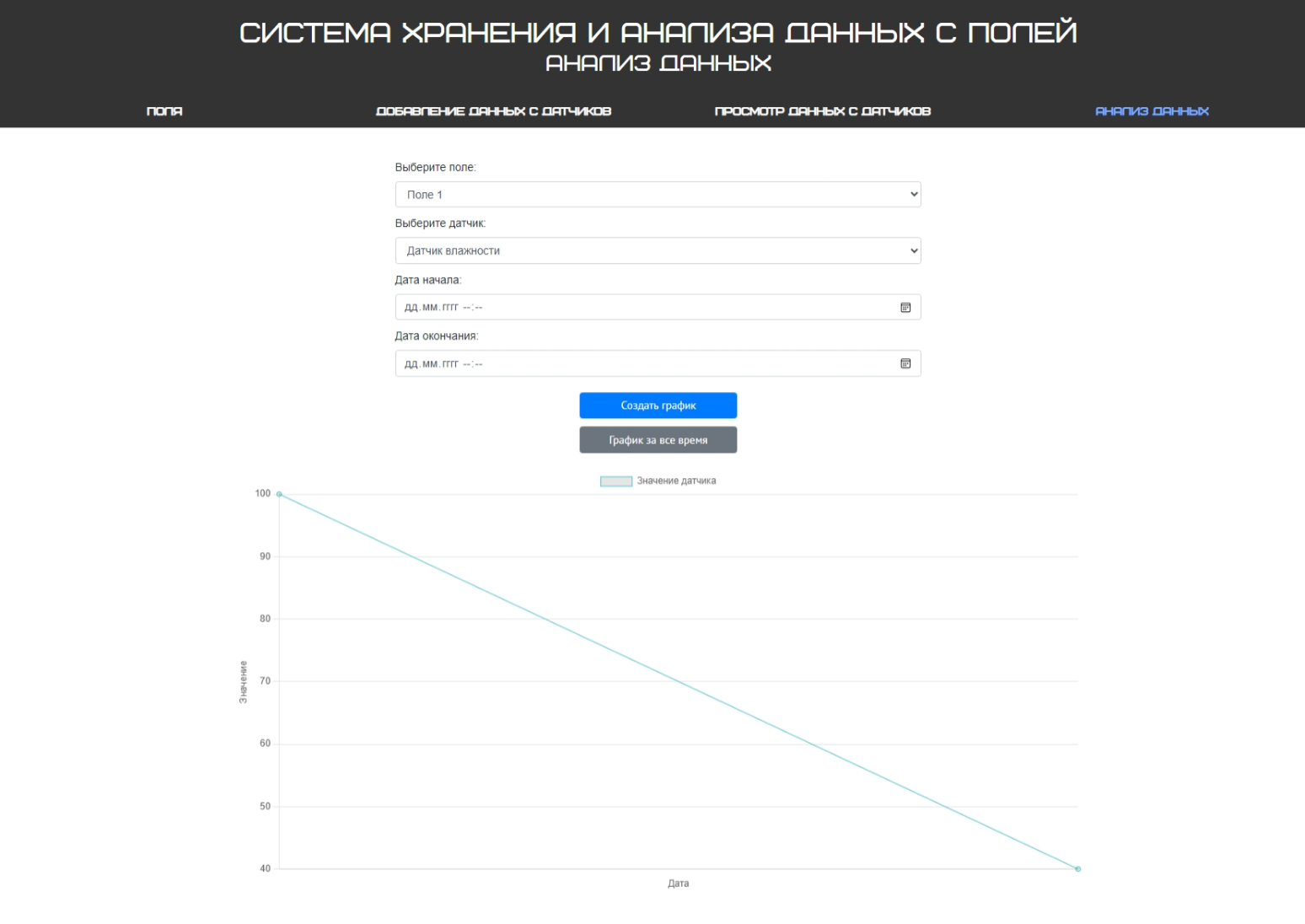


Рисунок 18 - Страница с анализом данных

## ЗАКЛЮЧЕНИЕ

Таким образом, в ходе разработки информационной системы для хранения и анализа данных с полей было проведено исследование предметной области, в результате которого были поставлены цели и задачи информационной системы, выявлены основные пользователи системы. Было проведено проектирование и комплексное моделирование системы. Функциональное моделирование осуществлялось с использованием методологий IDEF0 и IDEF3, а также DFD. Помимо этого, использовалось объектное моделирование в методике UML. Моделирование позволило декомпозировать систему и выявить основные компоненты, а также сформировать структуру проекта. В ходе работы были сформированы технические требования, которые содержат информацию о функциональности, производительности, надежности, безопасности, интерфейсе и других аспектах разрабатываемого программного обеспечения.

Опираясь на эти требования, в итоге была реализована информационная система для хранения и анализа данных с полей. Система включает в себя серверную и клиентскую части, обеспечивающие обмен данными между пользователями системы и базой данных MongoDB.

Ключевыми достоинствами разработанной информационной системы являются:

1. Автоматизация процессов: система позволяет сократить время выполнения операций и минимизировать ручной ввод данных благодаря автоматизированным процессам.

2. Улучшенная оперативность: благодаря возможности быстрого доступа к актуальным данным, пользователи могут оперативно реагировать на изменения и принимать решения на основе актуальной информации.

Общий результат проекта демонстрирует успешную реализацию информационной системы, способствующей повышению эффективности и качества управления сельскохозяйственными процессами. Дальнейшее развитие системы может включать в себя добавление новых функциональных возможностей, улучшение пользовательского интерфейса и оптимизацию производительности для обеспечения более эффективной работы персонала.

## СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Башам, Д., Сьерра, К., и Бейтс, Б. Head First Java. – Москва: Эксмо, 2020. – 720 с.
2. Гамма, Э., Хелм, Р., Джонсон, Р., и Влиссидес, Дж. Паттерны проектирования. – Санкт-Петербург: Питер, 2016. – 368 с.
3. Конноли, Т., Бегг, К. Базы данных. Проектирование, реализация и сопровождение. – Москва: Вильяме, 2003. – 350 с.
4. Фаулер, М. UML. Основы. – Москва: ДМК Пресс, 2006. – 208 с.
5. Фримен, Э., и Фримен, Э. Изучаем Java. – Москва: Питер, 2021. – 864 с.
6. Харрис, А. Spring в действии. – Москва: ДМК Пресс, 2019. – 720 с.
7. ГОСТ 34.601-90. Информационная технология. Комплекс стандартов на автоматизированные системы. Автоматизированные системы стадии создания (утвержден постановлением Госстандарта СССР от 29.12.1990 № 3469).
8. Документация MongoDB. Официальный сайт MongoDB. URL: https://docs.mongodb.com/
9. Документация Spring Boot. Официальный сайт Spring. URL: https://spring.io/projects/spring-boot
10. Документация Thymeleaf. Официальный сайт Thymeleaf. URL: https://www.thymeleaf.org/documentation.html

## ПРИЛОЖЕНИЕ А

|  |
| --- |
| **ИНФОРМАЦИОННАЯ СИСТЕМА ДЛЯ ХРАНЕНИЯ И АНАЛИЗА ДАННЫХ С ПОЛЕЙ** |
| **Руководство программиста** |

**Санкт-Петербург 2024**

**Содержание**

[**1. Назначение и условия применения программы 2**](#_Toc167752557)

[**1.1 Назначение программы 2**](#_Toc167752558)

[**1.2 Функции, выполняемые программой 2**](#_Toc167752559)

[**1.3 Условия, необходимые для выполнения программы 2**](#_Toc167752560)

[**1.3.1. Объем оперативной памяти 2**](#_Toc167752561)

[**1.3.2. Требования к составу периферийных устройств 3**](#_Toc167752562)

[**1.3.3. Требования к параметрам периферийных устройств 3**](#_Toc167752563)

[**1.3.4. Требования к программному обеспечению 3**](#_Toc167752564)

[**1.3.5. Требования к персоналу (программисту) 3**](#_Toc167752565)

[**2. Характеристика программы 4**](#_Toc167752566)

[**2.1 Описание основных характеристик программы 4**](#_Toc167752567)

[**2.1.1. Режим работы программы 4**](#_Toc167752568)

[**3. Обращение к программе 4**](#_Toc167752569)

[**3.1 Логика использования контроллеров 4**](#_Toc167752570)

[**3.2 Описание JavaScript функций контроллеров 8**](#_Toc167752571)

[**4. Входные и выходные данные 13**](#_Toc167752572)

[**4.1 Организация используемой входной информации 13**](#_Toc167752573)

[**4.2 Организация используемой выходной информации 15**](#_Toc167752574)

**1. Назначение и условия применения программы**

**1.1 Назначение программы**

Программа "Система управления сельскохозяйственными полями" предназначена для автоматизации сбора, хранения и анализа данных с различных датчиков, установленных на полях. Она позволяет пользователям эффективно управлять сельскохозяйственными процессами, повышать урожайность и принимать обоснованные решения на основе собранных данных.

**1.2 Функции, выполняемые программой**

* Загрузка данных с датчиков: Загрузка информации о влажности почвы, температуре и других параметрах.
* Анализ данных: Проведение анализа собранных данных для выявления тенденций и аномалий.
* Визуализация данных: Представление данных в виде графиков и таблиц для удобного восприятия.
* Управление датчиками: Возможность добавления, обновления и удаления информации о датчиках.

**1.3 Условия, необходимые для выполнения программы**

Для обеспечения надлежащей работы системы управления сельскохозяйственными полями 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 датчиков 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.

**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 Windows 10 или новее.
* 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 Windows, 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 сельскохозяйственными полями представляет собой комплексное программное решение, разработанное для мониторинга и управления данными, поступающими с датчиков, установленных на сельскохозяйственных полях. Программа позволяет пользователям эффективно управлять данными, проводить их анализ и визуализацию, а также управлять датчиками и полями.

**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 Логика использования контроллеров**

**FieldController**

1. Функция: getAllFields

* Назначение функции: получение списка всех полей.
* Входные параметры: нет
* Выходные параметры: список всех полей (List<Field>)
* Логика работы: получает все записи из коллекции полей в базе данных и возвращает их в виде списка.

2. Функция: getFieldById

* Назначение функции: получение информации о поле по его идентификатору.
* Входные параметры: string id
* Выходные параметры: поле (Field) или null, если поле не найдено
* Логика работы: находит поле в базе данных по его идентификатору и возвращает его. Если поле не найдено, возвращает null.

3. Функция: createField

* Назначение функции: создание нового поля.
* Входные параметры: поле (Field)
* Выходные параметры: созданное поле (Field)
* Логика работы: принимает данные нового поля, сохраняет их в базе данных и возвращает сохраненное поле.

4. Функция: updateField

* Назначение функции: обновление информации о поле.
* Входные параметры: string id, Поле (Field)
* Выходные параметры: обновленное поле (Field)
* Логика работы: находит поле по идентификатору, обновляет его данными из входного параметра и сохраняет изменения в базе данных, возвращая обновленное поле.

5. Функция: deleteField

* Назначение функции: удаление поля по его идентификатору.
* Входные параметры: string id
* Выходные параметры: нет
* Логика работы: удаляет запись о поле из базы данных по его идентификатору.

**SensorDataController**

1. Функция: addSensorData

* Назначение функции: добавление данных с датчиков.
* Входные параметры: список данных с датчиков (List<SensorData>)
* Выходные параметры: сохраненные данные с датчиков (List<SensorData>)
* Логика работы: принимает список данных с датчиков, сохраняет их в базе данных и возвращает сохраненные данные.

2. Функция: getFieldSensorData

* Назначение функции: получение данных с датчиков для конкретного поля.
* Входные параметры: string fieldId, int page, int size
* Выходные параметры: список данных с датчиков (List<SensorData>)
* Логика работы: получает данные с датчиков, связанные с указанным полем, из базы данных с учетом пагинации и возвращает их в виде списка.

3. Функция: getSensorDataById

* Назначение функции: получение данных с датчика по его идентификатору.
* Входные параметры: string id
* Выходные параметры: данные с датчика (SensorData) или null, если данные не найдены
* Логика работы: находит данные с датчика по идентификатору и возвращает их. Если данные не найдены, возвращает null.

4. Функция: updateSensorData

* Назначение функции: обновление данных с датчика.
* Входные параметры: string id, Данные с датчика (SensorData)
* Выходные параметры: обновленные данные с датчика (SensorData)
* Логика работы: находит данные с датчика по идентификатору, обновляет их данными из входного параметра и сохраняет изменения в базе данных, возвращая обновленные данные.

5. Функция: deleteSensorData

* Назначение функции: удаление данных с датчика по его идентификатору.
* Входные параметры: string id
* Выходные параметры: нет
* Логика работы: удаляет запись о данных с датчика из базы данных по его идентификатору.

6. Функция: getSensorData

* Назначение функции: получение данных с датчика за определенный период.
* Входные параметры: string fieldId, String sensorName, String start, String end
* Выходные параметры: список данных с датчиков (List<SensorData>)
* Логика работы: получает данные, собранные указанным датчиком в указанный период, и возвращает их в виде списка.

7. Функция: getAllTimeSensorData

* Назначение функции: получение всех данных с датчика.
* Входные параметры: string fieldId, String sensorName
* Выходные параметры: список всех данных с датчика (List<SensorData>)
* Логика работы: получает все данные, собранные указанным датчиком, и возвращает их в виде списка.

**PageController**

1. Функция: getFieldsPage

* Назначение функции: отображение страницы с полями.
* Входные параметры: нет
* Выходные параметры: имя HTML файла (String)
* Логика работы: возвращает имя HTML файла для отображения страницы с полями.

2. Функция: getFieldDataPage

* Назначение функции: отображение страницы с данными полей.
* Входные параметры: нет
* Выходные параметры: имя HTML файла (String)
* Логика работы: возвращает имя HTML файла для отображения страницы с данными полей.

3. Функция: getAddSensorDataPage

* Назначение функции: отображение страницы для добавления данных с датчиков.
* Входные параметры: нет
* Выходные параметры: имя HTML файла (String)
* Логика работы: возвращает имя HTML файла для отображения страницы для добавления данных с датчиков.

4. Функция: getAnalysisPage

* Назначение функции: отображение страницы анализа данных.
* Входные параметры: нет
* Выходные параметры: имя HTML файла (String)
* Логика работы: возвращает имя HTML файла для отображения страницы анализа данных.

**3.2 Описание JavaScript функций контроллеров**

**Основные функции для работы с полями**

1. Функция: fetchFields

* Назначение функции: получение списка всех полей и обновление данных таблицы на странице.
* Входные параметры: нет
* Выходные параметры: нет
* Логика работы: отправляет запрос на сервер для получения списка всех полей, после чего обновляет данные таблицы на странице.

2. Функция: addField

* Назначение функции: добавление нового поля.
* Входные параметры: нет (Использует значения из полей ввода на странице)
* Выходные параметры: нет
* Логика работы: собирает данные нового поля и его датчиков из полей ввода на странице, отправляет запрос на сервер для добавления нового поля, после чего обновляет таблицу полей на странице.

3. Функция: editField

* Назначение функции: редактирование данных существующего поля.
* Входные параметры: string fieldId
* Выходные параметры: нет
* Логика работы: отправляет запрос на сервер для получения данных выбранного поля по его идентификатору, заполняет поля ввода в модальном окне для редактирования данными поля.

4. Функция: saveFieldChanges

* Назначение функции: сохранение изменений в данных поля.
* Входные параметры: нет (Использует значения из модального окна редактирования)
* Выходные параметры: нет
* Логика работы: собирает обновленные данные поля из модального окна, отправляет запрос на сервер для сохранения изменений, после чего обновляет таблицу полей на странице.

5. Функция: deleteField

* Назначение функции: удаление поля.
* Входные параметры: string fieldId
* Выходные параметры: Нет
* Логика работы: отправляет запрос на сервер для удаления поля по его идентификатору, после чего обновляет таблицу полей на странице.

**Функции для работы с датчиками**

1. Функция: addSensorInput

* Назначение функции: добавление нового поля ввода для датчика.
* Входные параметры: нет
* Выходные параметры: нет
* Логика работы: добавляет новое поле ввода для датчика на странице формы добавления или редактирования поля.

2. Функция: removeLastSensorInput

* Назначение функции: удаление последнего поля ввода для датчика.
* Входные параметры: нет
* Выходные параметры: нет
* Логика работы: удаляет последнее поле ввода для датчика на странице формы добавления или редактирования поля.

3. Функция: fetchSensorData

* Назначение функции: получение данных с датчиков для конкретного поля.
* Входные параметры: string fieldId, int page, int size
* Выходные параметры: нет
* Логика работы: отправляет запрос на сервер для получения данных с датчиков для конкретного поля, используя параметры пагинации, и обновляет данные таблицы на странице.

**Визуализация и анализ данных**

1. Функция: highlightActiveNavItem

* Назначение функции: подсветка активного элемента навигационного меню.
* Входные параметры: нет
* Выходные параметры: нет
* Логика работы: определяет текущий URL страницы и подсвечивает соответствующий элемент навигационного меню как активный.

2. Функция: createGraph

* Назначение функции: создание и отображение графика изменений данных с датчика.
* Входные параметры: string fieldId, String sensorName, String period
* Выходные параметры: нет
* Логика работы: отправляет запрос на сервер для получения данных с датчика за указанный период, создает и отображает график изменений данных с датчика на странице.

3. Функция: displaySensorData

* Назначение функции: Отображение данных с датчиков.
* Входные параметры: Object data
* Выходные параметры: нет
* Логика работы: обновляет данные таблицы или графика на странице для отображения полученных данных с датчиков.

**Обработка событий и взаимодействие с пользователем**

1. Функция: handleFileSelect

* Назначение функции: обработка выбора файла пользователем для загрузки данных с датчиков.
* Входные параметры: event event
* Выходные параметры: Нет
* Логика работы: Обрабатывает выбор файла пользователем, читает содержимое файла и отправляет данные на сервер для загрузки данных с датчиков.

2. Функция: resetForm

* Назначение функции: Сброс формы добавления поля.
* Входные параметры: Нет
* Выходные параметры: Нет
* Логика работы: Сбрасывает форму добавления поля к первоначальному состоянию после успешного добавления нового поля.

3. Функция: confirmDeleteField

* Назначение функции: Подтверждение удаления поля.
* Входные параметры: String fieldId
* Выходные параметры: Нет
* Логика работы: Открывает модальное окно для подтверждения удаления поля, передает идентификатор поля для удаления.

Эти функции и контроллеры обеспечивают полное взаимодействие пользователя с системой управления сельскохозяйственными полями, от добавления и управления полями и датчиками до анализа и визуализации собранных данных.

**4. Входные и выходные данные**

**4.1 Организация используемой входной информации**

Входная информация для системы управления сельскохозяйственными полями включает:  
**1. Данные с датчиков:**

* Описание: Показания различных датчиков, установленных на полях.
* Формат: JSON-объекты, содержащие идентификатор поля, имя датчика, единицу измерения, метку времени и значение.
* Пример:

{

"fieldId": "123",

"sensorName": "soil\_moisture",

"unit": "%",

"timestamp": "2024-05-28T14:00:00Z",

"value": 25.5

}

**2. Пользовательский ввод:**

* Описание: Данные, вводимые пользователями через веб-интерфейс для создания и редактирования полей и датчиков.
* Формат: Поля ввода HTML-форм, которые отправляются на сервер в формате JSON.
* Пример:

{

"fieldName": "Field 1",

"sensors": [

{"sensorName": "temperature"},

{"sensorName": "humidity"}

]

}

**3. Файлы с заданиями:**

* Описание: Файлы, загружаемые пользователями, содержащие задания для добавления данных с датчиков.
* Формат: JSON-файлы.
* Пример:

[

{

"fieldId": "123",

"sensorName": "temperature",

"unit": "°C",

"timestamp": "2024-05-28T14:00:00Z",

"value": 20.0

}

]

**4. Запросы к API:**

* Описание: HTTP-запросы к API для получения, добавления, обновления и удаления данных.
* Формат: JSON-объекты, передаваемые в теле запроса.
* Пример GET-запроса:

GET /fields

**4. Пример POST-запроса:**

POST /fields

{

"fieldName": "Field 1",

"sensors": [

{"sensorName": "temperature"},

{"sensorName": "humidity"}

]

}

**4.2 Организация используемой выходной информации**

Выходная информация, генерируемая системой, включает:

**1. Данные полей и датчиков:**

* Описание: Информация о полях и датчиках, а также показания датчиков.
* Формат: JSON-объекты, возвращаемые в ответ на запросы API.
* Пример:

[

{

"id": "123",

"fieldName": "Field 1",

"sensors": [

{"sensorName": "temperature"},

{"sensorName": "humidity"}

]

}

]

**2. Графики и визуализации:**

* Описание: Графическое представление данных с датчиков для анализа.
* Формат: HTML и JavaScript для отображения графиков на веб-страницах.
* Пример: График изменения температуры с датчика во времени.

Эти входные и выходные данные обеспечивают полный цикл взаимодействия пользователя с системой управления сельскохозяйственными полями, от ввода и обработки данных до их анализа и визуализации.

## ПРИЛОЖЕНИЕ Б

|  |
| --- |
| **ИНФОРМАЦИОННАЯ СИСТЕМА ДЛЯ ХРАНЕНИЯ И АНАЛИЗА ДАННЫХ С ПОЛЕЙ** |
| **Руководство пользователя** |

**Санкт-Петербург 2024**

**Оглавление**

[**1. Введение 2**](#_Toc167743827)

[**1.1. Область применения 2**](#_Toc167743828)

[**1.2. Краткое описание возможностей 2**](#_Toc167743829)

[**1.3 Уровень подготовки пользователя 2**](#_Toc167743830)

[**2. Назначение и условия применения системы 3**](#_Toc167743831)

[**2.1. Назначение системы 3**](#_Toc167743832)

[**2.2. Условия применения системы 4**](#_Toc167743833)

[**2.3. Операционные условия 5**](#_Toc167743834)

[**3. Подготовка к работе 6**](#_Toc167743835)

[**3.1. Порядок загрузки данных и программ 6**](#_Toc167743836)

[**3.2. Порядок проверки работоспособности 7**](#_Toc167743837)

[**4. Описание функций 8**](#_Toc167743838)

[**5. Аварийные ситуации 14**](#_Toc167743839)

[**6. Рекомендации по освоению 15**](#_Toc167743840)

1. **Введение**

**1.1. Область применения**

Требования настоящего документа применяются для разработки, тестирования и эксплуатации системы управления сельскохозяйственными полями, включая сбор и анализ данных с датчиков. Эта система используется для мониторинга различных параметров полей, таких как влажность почвы, температура и другие важные показатели, что позволяет эффективно управлять сельскохозяйственными процессами и повышать урожайность.

**1.2. Краткое описание возможностей**

Система управления сельскохозяйственными полями предназначена для автоматизации сбора данных с различных датчиков, установленных на полях, их анализа и представления в удобной для пользователей форме. Основные функции системы включают:

* Сбор данных с датчиков: автоматический сбор информации о влажности почвы, температуре и других параметрах.
* Анализ данных: проведение анализа собранных данных для выявления тенденций и аномалий.
* Визуализация данных: представление данных в виде графиков и таблиц для удобного восприятия.
* Управление датчиками: возможность добавления, обновления и удаления информации о датчиках.

**1.3 Уровень подготовки пользователя**

Для эффективного использования системы управления сельскохозяйственными полями пользователь должен обладать следующими навыками и знаниями:

Опыт работы с операционной системой: знание основ работы с операционной системой, на которой развернута система (например, Windows или macOS).

* Навыки работы с браузером: умение пользоваться интернет-браузером для доступа к веб-интерфейсу системы.
* Знание основ работы с датчиками: понимание принципов работы датчиков, используемых в сельском хозяйстве, и их основных характеристик.
* Навыки анализа данных: базовые знания в области анализа данных, умение интерпретировать графики и таблицы.
* Работа с документацией: умение читать и понимать техническую документацию, следовать инструкциям по установке и настройке системы.

Эти навыки необходимы для того, чтобы пользователь мог полноценно взаимодействовать с системой, обеспечивать ее стабильную работу и эффективно решать поставленные задачи.

1. **Назначение и условия применения системы**

**2.1. Назначение системы**

Система управления сельскохозяйственными полями создана для того, чтобы помочь пользователям собирать, хранить и анализировать данные, поступающие с различных сенсоров, установленных на сельскохозяйственных полях. Основное назначение системы — предоставить пользователям возможность не только вручную добавлять и управлять информацией о полях, датчиках и собранных данных, но и с помощью JSON файла, а также визуализировать эти данные для принятия обоснованных решений.

Основные цели системы включают:

* Мониторинг состояния полей: предоставление пользователям инструмента не только ручного ввода данных о полях и датчиках, но и с помощью JSON файла, а также возможности отслеживать изменения параметров окружающей среды.
* Анализ данных: возможность построения графиков и проведения анализа данных, собранных с сенсоров, для выявления тенденций и принятия решений.
* Управление датчиками: добавление, обновление и удаление информации о датчиках и полях.

**2.2. Условия применения системы**

Система предназначена для использования в сельскохозяйственных предприятиях, фермерских хозяйствах и исследовательских учреждениях, занимающихся агротехникой. Для успешного внедрения и эксплуатации системы необходимо учитывать следующие условия:

* **Оборудование:**
  + Компьютер или сервер, на котором будет развернута система, с доступом к интернету для передачи данных.
  + Сенсоры, установленные на полях, для сбора данных о различных параметрах окружающей среды.
* **Программное обеспечение:**
  + Операционная система, совместимая с системой управления (например, Windows, macOS, Linux).
  + Веб-браузер для доступа к интерфейсу системы (рекомендуется использовать современные браузеры, такие как Google Chrome, Mozilla Firefox, Safari).
* **Инфраструктура:**
  + Надежное интернет-соединение для передачи данных с сенсоров на сервер и для удаленного доступа к системе.
  + Энергоснабжение для обеспечения бесперебойной работы сенсоров и сервера.
* **Подготовка персонала:**
  + Обучение пользователей работе с системой, включая добавление полей, датчиков, ввод данных и построение графиков.
  + Регулярное техническое обслуживание сенсоров и программного обеспечения для поддержания их работоспособности.

**2.3. Операционные условия**

Система должна функционировать в различных операционных условиях, в зависимости от специфики сельскохозяйственного производства и климатических условий региона. Это включает:

* Температурные условия: сенсоры и серверы должны быть устойчивыми к температурным колебаниям, характерным для региона эксплуатации.
* Влажность: оборудование должно быть защищено от влаги и пыли, особенно в полевых условиях.
* Поддержка различных культур: система должна быть гибкой и настраиваемой под различные виды сельскохозяйственных культур и их специфические требования.

Система управления сельскохозяйственными полями разработана с учетом этих условий, что обеспечивает её универсальность и адаптивность к различным сценариям использования. Пользователи могут самостоятельно вводить и управлять данными, а также анализировать их с помощью инструментов визуализации, предоставляемых системой.

1. **Подготовка к работе**

**3.1. Порядок загрузки данных и программ**

Чтобы система управления сельскохозяйственными полями 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.

1. 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:8081/fields-page. Это адрес, по которому сотрудники смогут входить в систему и начинать работу.

Эти шаги помогут правильно запустить систему управления сельскохозяйственными полями и обеспечат её стабильную и безопасную работу.

**3.2. Порядок проверки работоспособности**

Чтобы убедиться в том, что система управления сельскохозяйственными полями работает и доступна для использования, можно выполнить следующие шаги:

1. Запуск браузера:

* Найдите иконку браузера на рабочем столе или в меню приложений.
* Щелкните по иконке браузера, чтобы открыть его.

1. Ввод адреса системы:
   * В адресной строке браузера введите URL-адрес, http://localhost:8081 и нажмите клавишу Enter.
2. Проверка основных функций:

* Добавление поля: перейдите в раздел добавления полей и создайте новое поле, указав его название.
* Добавление датчика: в разделе управления датчиками добавьте новый датчик, связанный с ранее созданным полем.
* Ввод данных с датчиков: перейдите в раздел ввода данных и добавьте несколько записей с показаниями датчиков.
* Просмотр данных: убедитесь, что вы можете просматривать добавленные данные в разделе визуализации, строить графики и анализировать информацию.

1. Проверка доступности системы:

* Если перед вами открылась страница, где отображаются поля и их датчики, это значит, что система работает правильно.

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

1. **Описание функций**

В системе управления сельскохозяйственными полями доступны следующие основные функции, которые помогают пользователям управлять полями, датчиками и собранными данными:

4.1. Управление полями и датчиками

* Добавление поля и датчика:
* Перейдите на страницу «Поля».
* Введите название «Поля» в поле для ввода. 
* Введите название датчика в поле для ввода.



* Если вы хотите добавить еще датчик, нажмите на соответствующую кнопку и введите в появившемся поле для ввода название датчика.



* Нажмите кнопку «Добавить поле».



* Новое поле и его датчики будет добавлено в систему и станет доступным для дальнейшего управления.
* Редактирование поля и датчика:
* Перейдите на страницу «Поля».
* Выберите в таблице поле, которое хотите отредактировать, и нажмите кнопку «Редактировать».



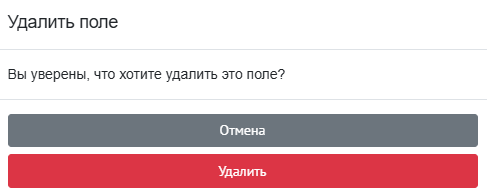
* В появившемся окне внесите необходимые изменения и нажмите «Сохранить изменения». (Изменять названия существующих датчиков можно при условии, что в базе данных нет данных с этих датчиков, но вы в любом случае сможете добавить новые датчики)



* Обновленная информация о поле будет сохранена в системе.
* Удаление поля и датчика:
* Перейдите на страницу «Поля».
* Выберите поле, которое хотите удалить, и нажмите кнопку «Удалить».



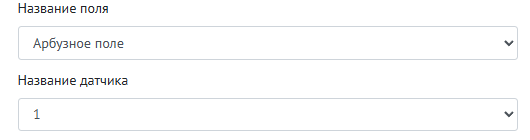
* Подтвердите удаление.



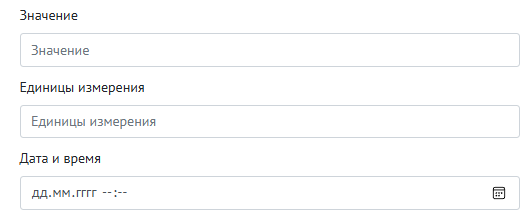
* Поле и его датчики будет удалено из системы.

4.2. Ввод данных с датчиков

* Добавление данных вручную:
  + Перейдите на страницу «Добавление данных с датчиков».
  + Выберите поле и датчик, к которым относятся данные.



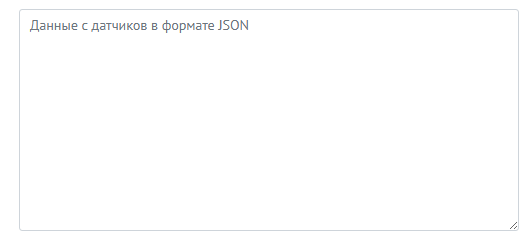
* + Введите значение параметра, единицу измерения и дату/время измерения.



* + Нажмите кнопку «Добавить данные».



* + Новые данные будут добавлены в систему и станут доступными для анализа.
* Добавление данных автоматически:
  + Перейдите на страницу «Добавление данных с датчиков».
  + Введите в окно данные в JSON формате.



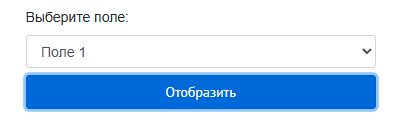
* + Нажмите кнопку «Загрузить».



* + Также вы можете загрузить данные из файла нажав на соответствующую кнопку.  
    
  + Нажмите кнопку «Загрузить из файла».



* Редактирование данных:
  + Перейдите на страницу «Просмотр данных с датчиков».
  + Выберите поле, в котором вы хотите редактировать данные с датчиков, и нажмите кнопку «Отобразить».



* + Нажмите кнопку «Редактировать».



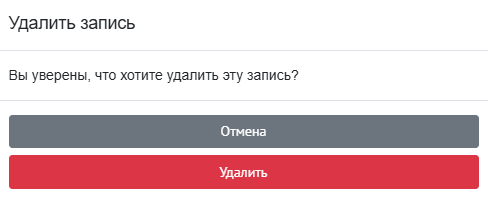
* + Внесите необходимые изменения и нажмите «Сохранить изменения».



* + Обновленная информация о данных будет сохранена в системе.
* Удаление данных:
  + Перейдите в раздел «Данные с датчиков».
* Выберите запись, которую хотите удалить, и нажмите кнопку «Удалить».



* Подтвердите удаление.



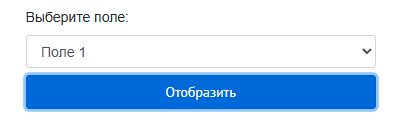
* + Запись данных будет удалена из системы.

4.3. Визуализация данных

* Просмотр полей и датчиков:
  + Перейдите на страницу «Поля».
  + Данные будут отображены в виде таблицы с возможностью сортировки и фильтрации.
* Просмотр полей и датчиков:

    Перейдите на страницу «Просмотр данных с датчиков».

        Выберите поле и нажмите кнопку «Отобразить».



* + Данные будут отображены в виде таблицы с возможностью сортировки и фильтрации.
* Построение графиков:
  + Перейдите на страницу «Анализ данных».
  + Выберите поле, датчик и период времени, за который вы хотите построить график.



* + Нажмите кнопку «Создать график».



* + Система построит график на основе выбранных параметров, отображая изменения значений датчика во времени.
  + Также присутствует возможность отображения графика за все время при нажатии на кнопку «График за все время». В этом случае график будет строиться на основе всех данных с этого датчика.



1. **Аварийные ситуации**

При использовании системы управления сельскохозяйственными полями 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. **Сервер не найден**:
   * **Описание ошибки**: невозможно соединиться с сервером.
   * **Возможные причины**: проблемы с сетью, сервер временно недоступен или неправильно введен адрес.
4. **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 сельскохозяйственными полями. 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 сельскохозяйственными полями 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 добавить данные с датчика и посмотреть график на основе загруженных данных.

1. **Запуск системы**:
   * Откройте браузер и введите адрес системы https://localhost:8081/fields-page в адресную строку.
2. **Добавление данных с датчика**:
   * Перейдите на страницу «Добавление данных с датчиков».
   * Загрузите JSON файл через кнопку «Выбор файла».
   * Нажмите кнопку «Загрузить из файла».
3. **Создание графика**:
   * Перейдите на страницу «Анализ данных».
   * Выберите поле, датчик и период времени, за который вы хотите построить график.
   * Нажмите кнопку «Создать график».

**Правила запуска и выполнения**

* **Подготовьте рабочее место**: убедитесь, что ваш компьютер подключен к сети и что все необходимые программы установлены и функционируют корректно.
* **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 сельскохозяйственными полями, а также поможет избежать распространенных ошибок и недоразумений в процессе её эксплуатации.