

КвадраИОТ

Описание функциональных характеристик

Дата: апрель, 2026 г.

Версия: 1.5.1

ООО «Квадра системс»

https//www.quadra-system.ru

Аннотация

Настоящий документ содержит описание функциональных характеристик программы КвадраИОТ (КИОТ), в том числе ее описание, информацию о ее назначении, основных функциональных характеристиках и возможностях, а также реализуемых задачах.

Глоссарий

RPM-пакет	–	пакет установки программного обеспечения для Linux-систем семейства RPM, содержащий файлы приложения и сценарии установки.
Java	–	программная среда, необходимая для запуска программы КвадраИОТ.
PostgreSQL	–	система управления базами данных, используемая КИОТ для хранения конфигурации, пользователей, событий и других данных.
Программа, ПО	–	КвадраИОТ, КИОТ.

Содержание

1 ОПИСАНИЕ И НАЗНАЧЕНИЕ ПРОГРАММЫ 4
2 ВЫПОЛНЕНИЕ ПРОГРАММЫ 5
2.1 Создание нового устройства 5
2.2 Проверка подключения устройства 6
2.3 Создание дашборда 7
2.4 Добавление виджетов в дашборд 9
2.5 Создание пользователя клиента 13
2.6 Назначение устройств и виджетов клиентам 16
2.7 Создание правил сигналов 20
2.8 Настройка цепочек правил 23
3 КОНТАКТЫ 45

1 Описание и назначение программы

Программа КвадраИОТ представляет собой сервис, предназначенный для обмена данными с подключёнными IoT-устройствами, их мониторинга, настройки и отображения рабочих параметров.

Программа предназначена для сбора, обработки, хранения и визуализации данных, поступающих от удалённых устройств, а также для контроля их состояния через единый веб-интерфейс.

Коммуникация программы с устройствами осуществляется по поддерживаемым сетевым протоколам обмена данными. Серверная часть обеспечивает приём сообщений от устройств, их обработку, сохранение в базе данных и последующее отображение в пользовательском интерфейсе.

Функции программы:

- приём и обработка данных от подключённых IoT-устройств;
- регистрация и учёт подключённых устройств;
- визуализация телеметрических данных;
- мониторинг состояния устройств;
- настройка параметров устройств;
- отображение событий и сообщений системы;
- управление доступом пользователей.

2 Выполнение программы

2.1 Создание нового устройства

Для добавления устройства в навигационном меню программы (Рисунок 1) необходимо выбрать раздел **«Сущности»** (1) и перейти на страницу **«Устройства»** (2). Далее в правом верхнем углу следует нажать кнопку **«+»** (3) и выбрать пункт **«Добавить новое устройство»**.

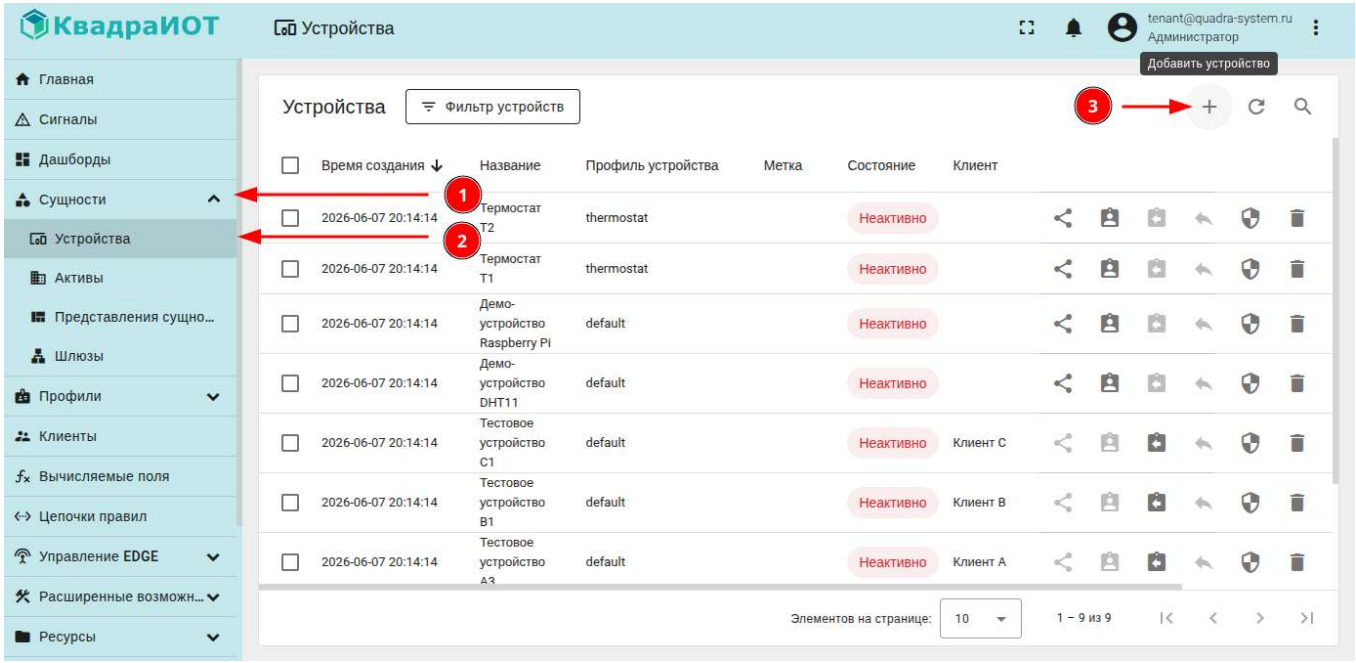


Рисунок 1 – Создание устройства

В открывшемся окне необходимо указать имя устройства и нажать кнопку **«Добавить»**. После создания устройства откроется окно проверки подключения. На этом этапе его можно закрыть и вернуться к настройке подключения позже.

После создания устройство появится в общем списке на странице устройств. Новые устройства обычно отображаются в верхней части таблицы. Если данные с устройства ещё не поступают, его состояние будет отображаться как неактивное.

После добавления система также создаёт уведомление. Его можно посмотреть через значок **колокольчика** в правом верхнем углу интерфейса.

Другие способы добавления устройств:

- Импорт из CSV;
- Автоматическая регистрация;
- Добавление через REST API.

2.2 Проверка подключения устройства

После создания устройства необходимо проверить его подключение к программе (Рисунок 2). Для этого на странице **«Устройства»** (1) необходимо выбрать **созданное устройство** (2). Справа откроется окно с его параметрами, в котором следует нажать кнопку **«Проверить подключение»** (3).

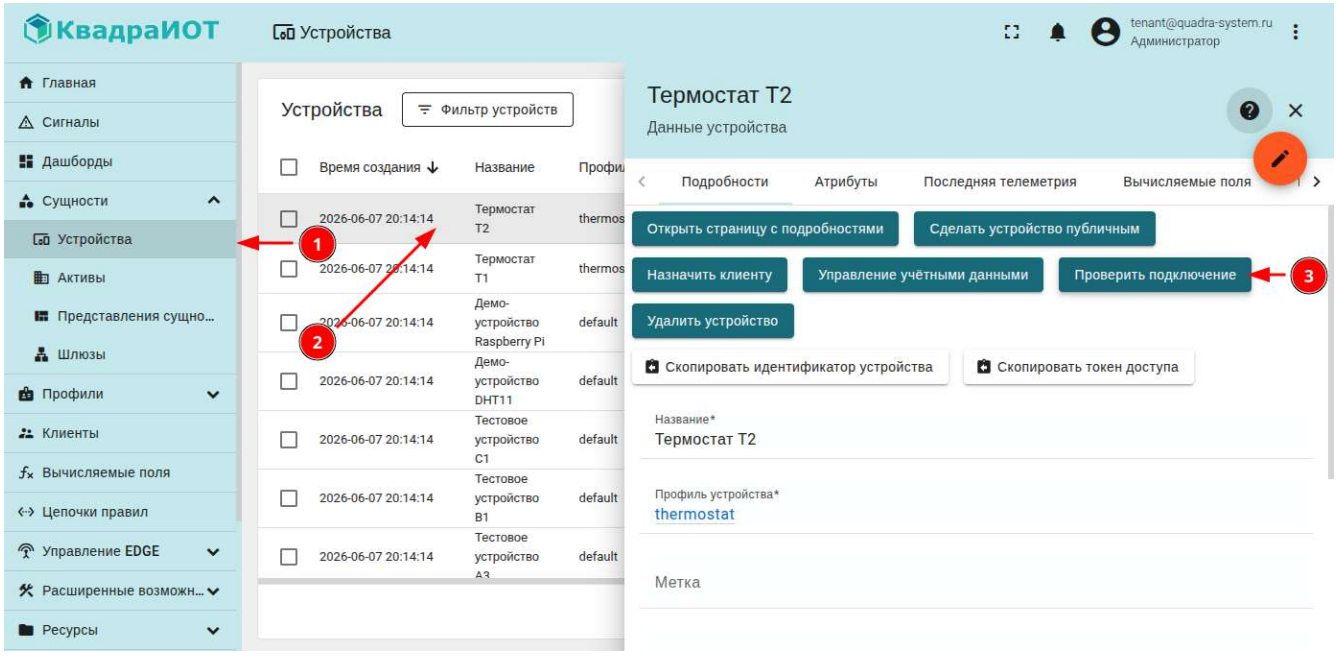


Рисунок 2 – Проверка подключения

После этого откроется окно проверки подключения (Рисунок 3).

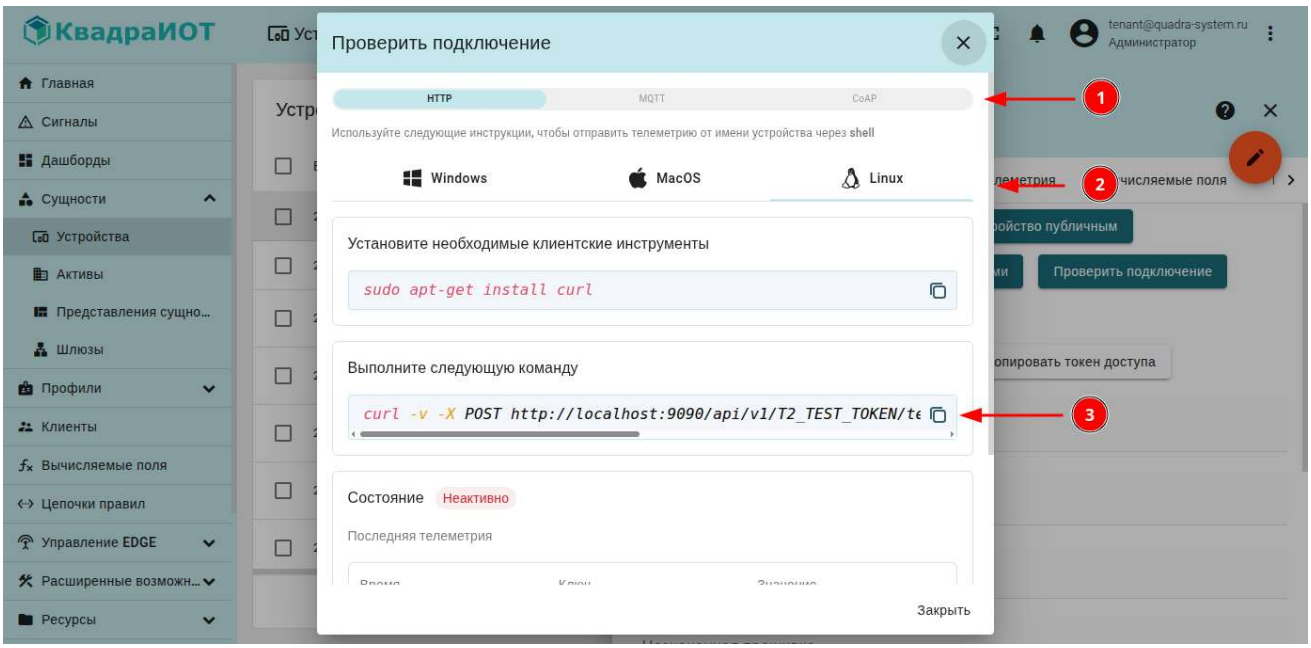


Рисунок 3 – Окно проверки подключения

В окне проверки подключения (Рисунок 3) необходимо выбрать используемый **протокол обмена данными** (1), а также **операционную систему** (2), на которой будет выполняться команда. Если это требуется, необходимо предварительно установить **клиентские инструменты, предложенные программой** (3).

Далее необходимо открыть терминал ОС (Рисунок 4) и выполнить в нем готовую **команду подключения** (1). После успешной отправки тестовых данных состояние устройства изменится с **«Неактивно»** на **«Активно»** (2). Ниже в этом же окне отобразятся полученные телеметрические данные. После проверки подключения окно можно **закрыть** (3).

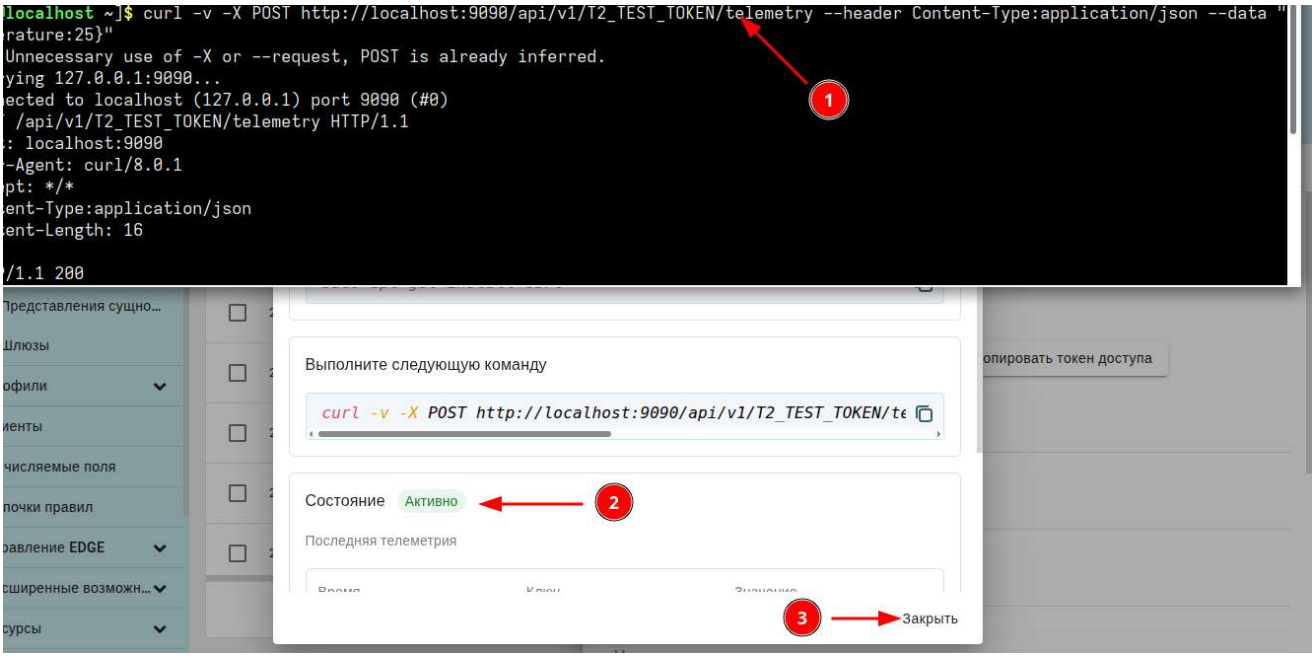


Рисунок 4 – Терминал проверки подключения

2.3 Создание дашборда

Дашборды в программе используется для отображения и контроля данных, поступающих от устройств.

Чтобы создать новый дашборд (Рисунок 5), необходимо в левом меню открыть раздел **«Дашборды»** (1). Затем в правом верхнем углу нажать кнопку **«+»** (2) и выбрать пункт **«Создать новый дашборд»**.

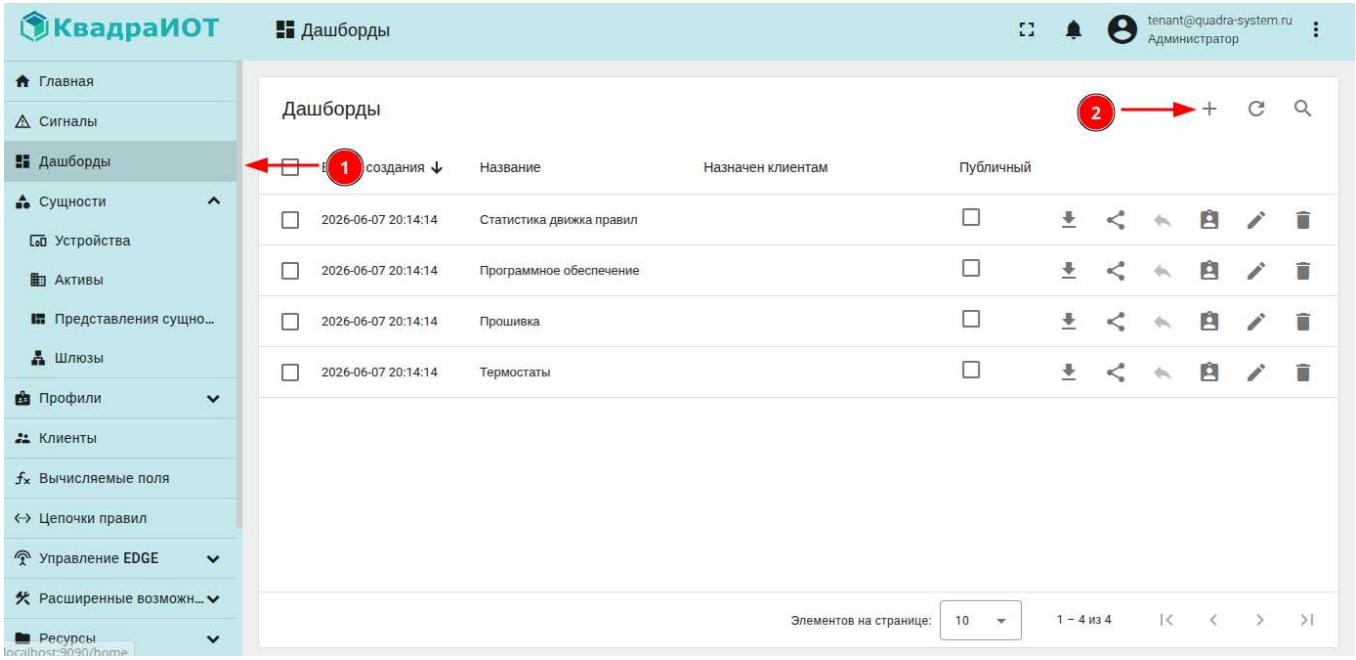


Рисунок 5 – Создание дашборда

В открывшемся окне (Рисунок 6) необходимо указать **название дашборда** (1). После этого необходимо нажать кнопку **«Добавить»** (2).

Поле описания не обязательно для заполнения.

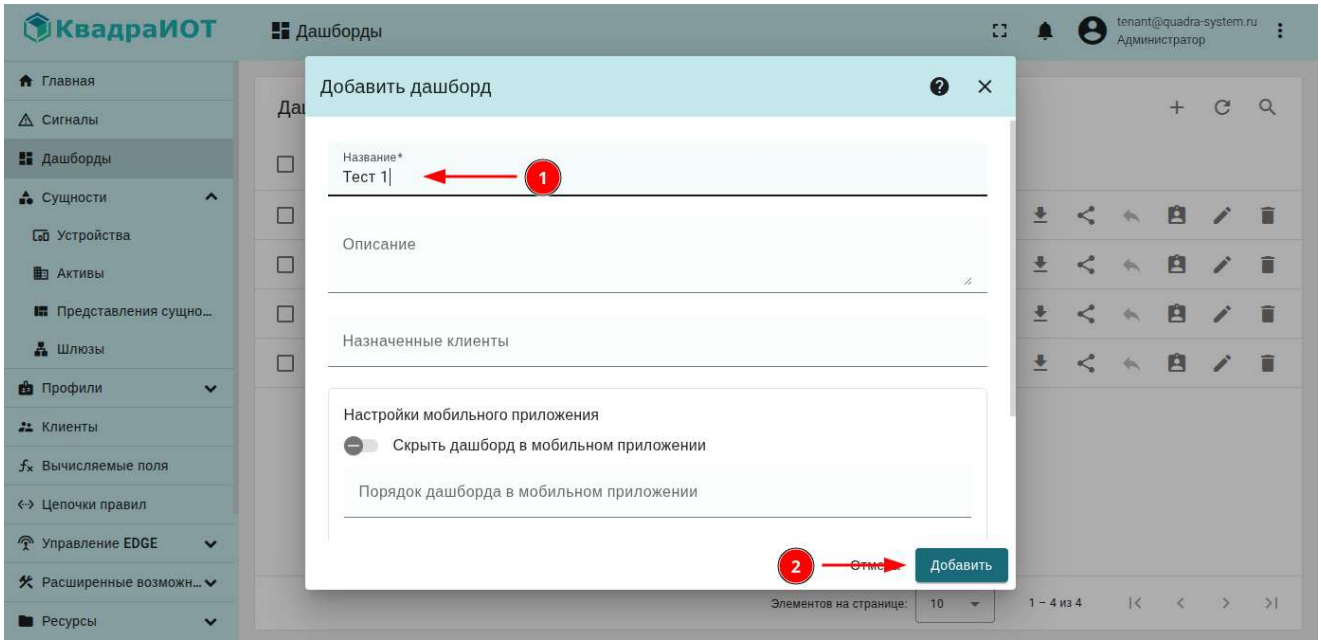


Рисунок 6 – Окно создания дашборда

Созданный дашборд откроется автоматически (Рисунок 7). Чтобы сохранить дашборд, необходимо нажать кнопку **«Сохранить»** (1). После сохранения дашборд появится в общем списке.

Новые дашборды обычно отображаются в верхней части таблицы.

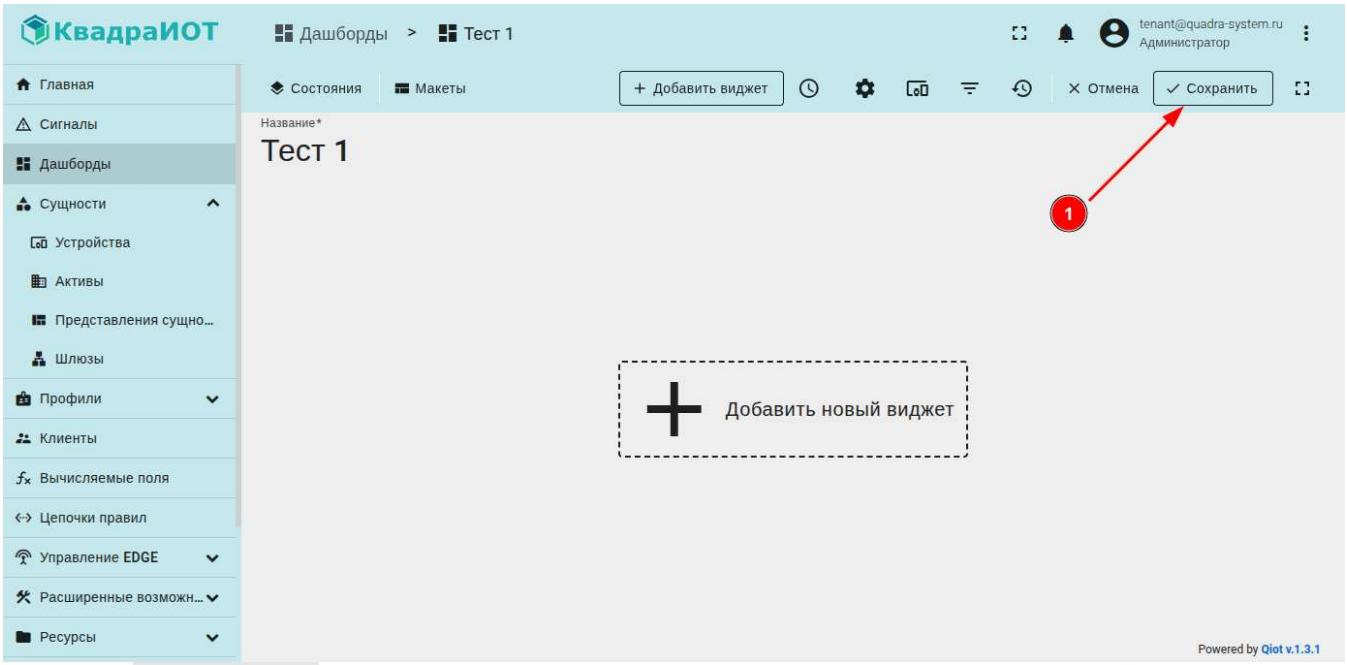


Рисунок 7 – Созданный дашборд

На странице созданного дашборда (Рисунок 7) можно сразу переходить к добавлению виджетов, нажав кнопку «Добавить новый виджет» в центре окна дашборда.

2.4 Добавление виджетов в дашборд

2.4.1 Таблица

Виджет **«Таблица»** используется для вывода списка устройств и их последних значений.

Для его добавления (Рисунок 8) необходимо открыть созданный дашборд и нажать кнопку **«Режим редактирования»** (1). После перехода в режим редактирования необходимо нажать **«Добавить виджет»** в верхней части окна. Если виджет добавляется впервые, можно нажать большую область **«Добавить новый виджет»** в центре экрана.

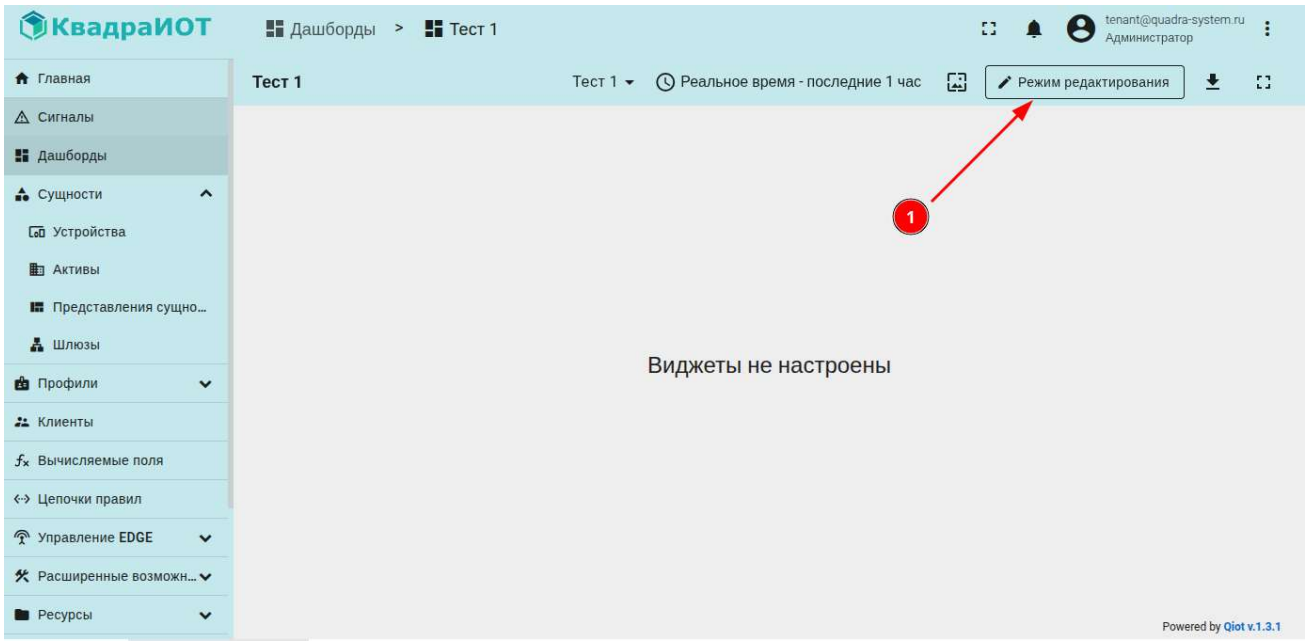


Рисунок 8 – Добавление виджета «Таблица»

Далее в библиотеке виджетов (Рисунок 9) необходимо выбрать группу **«Таблицы»**, а затем виджет **«Таблица сущностей»** (1).

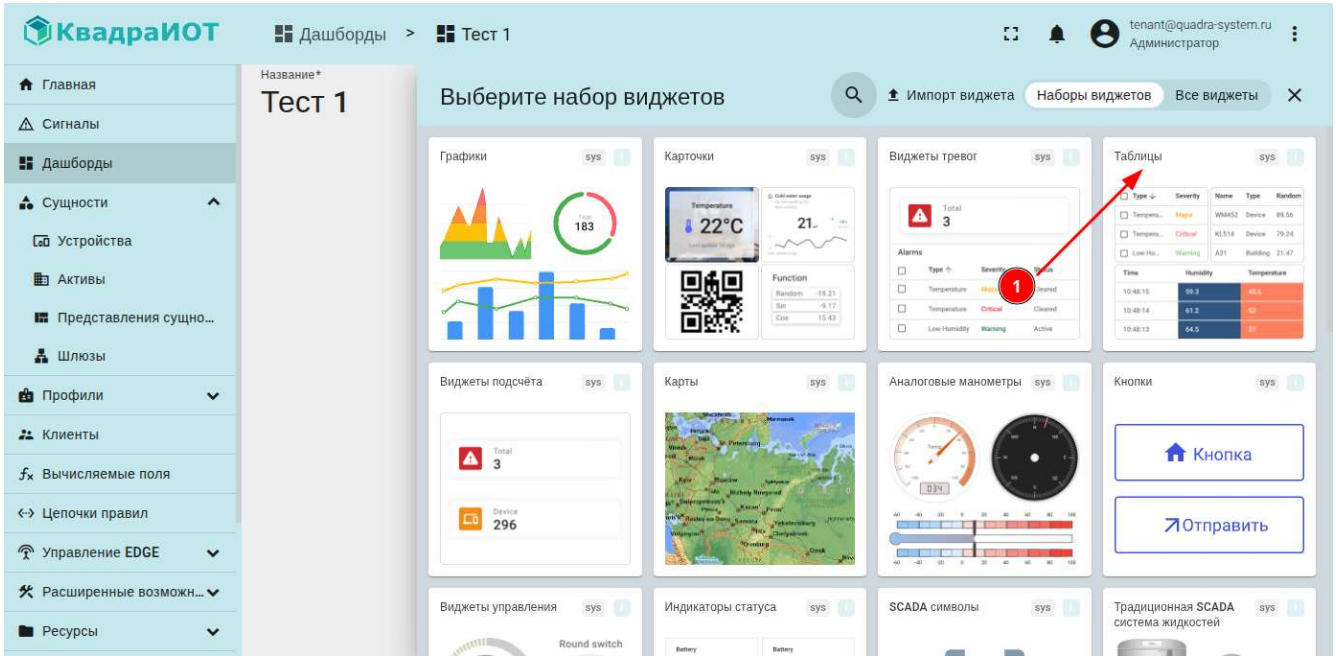


Рисунок 9 – Выбор виджета «Таблица» из библиотеки виджетов

После этого откроется окно настройки виджета (Рисунок 10). В качестве источника данных необходимо указать ранее созданное устройство **«Новое устройство»** (1). В списке колонок уже будет присутствовать поле имя, которое отвечает за вывод имени устройства. Чтобы добавить отображение

температуры, необходимо нажать кнопку **«Добавить колонку»** и выбрать ключ данных **temperature** (2).

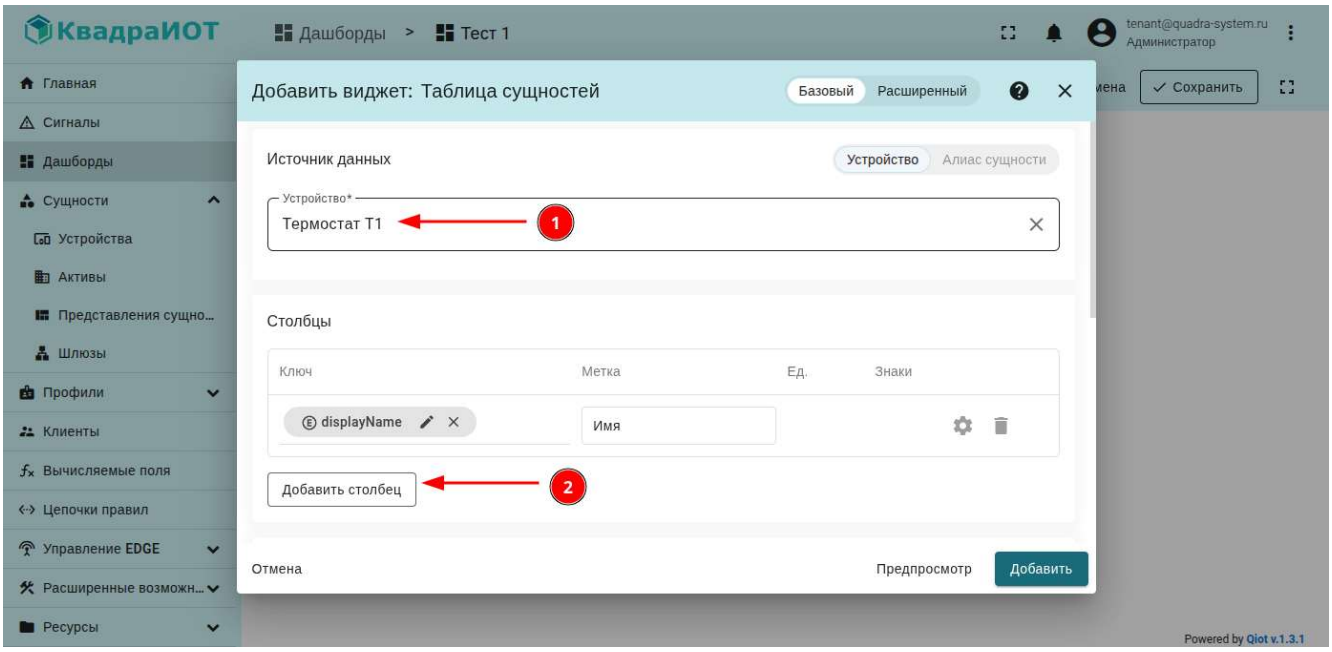


Рисунок 10 – Окно настройки виджета «Таблица»

При необходимости можно изменить подпись колонки, например указать температура, а также задать единицу измерения, например °C. После завершения настройки необходимо нажать кнопку **«Добавить».**

Добавленный виджет появится на дашборде. При необходимости его размер можно изменить, потянув за правый нижний угол. После этого необходимо нажать **«Сохранить»,** чтобы применить изменения.

2.4.2 График временных рядов

Виджет **«График временных рядов»** используется для отображения изменения показаний устройства за период времени.

Для его добавления необходимо открыть дашборд (Рисунок 7) в режиме редактирования и нажать **«Добавить виджет»** в центре дашборда. В библиотеке виджетов следует выбрать группу **«Графики»** (1), затем виджет **«График временных рядов»** (2).

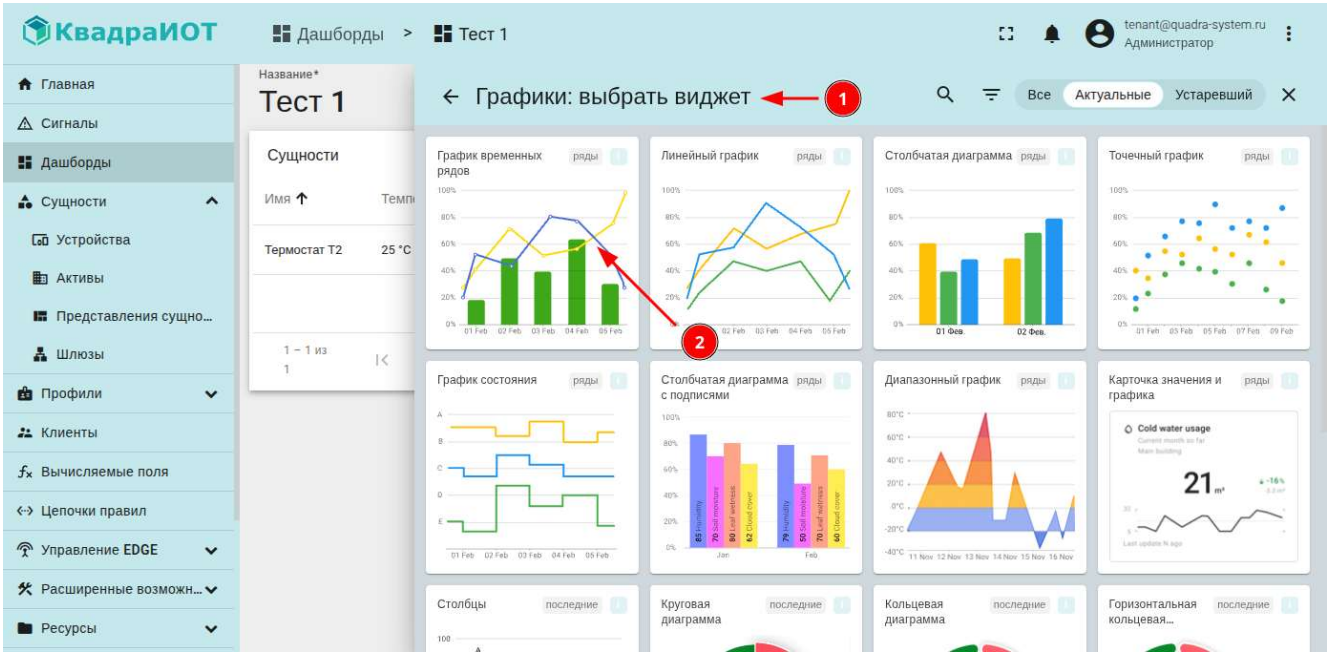


Рисунок 11 – Выбор виджета «График временных рядов» из библиотеки виджетов

В окне настройки виджета (Рисунок 12) в качестве источника данных необходимо указать устройство **«Новое устройство»** (1). В разделе рядов данных необходимо выбрать ключ **temperature** (2), после чего нажать **«Добавить»** (3).

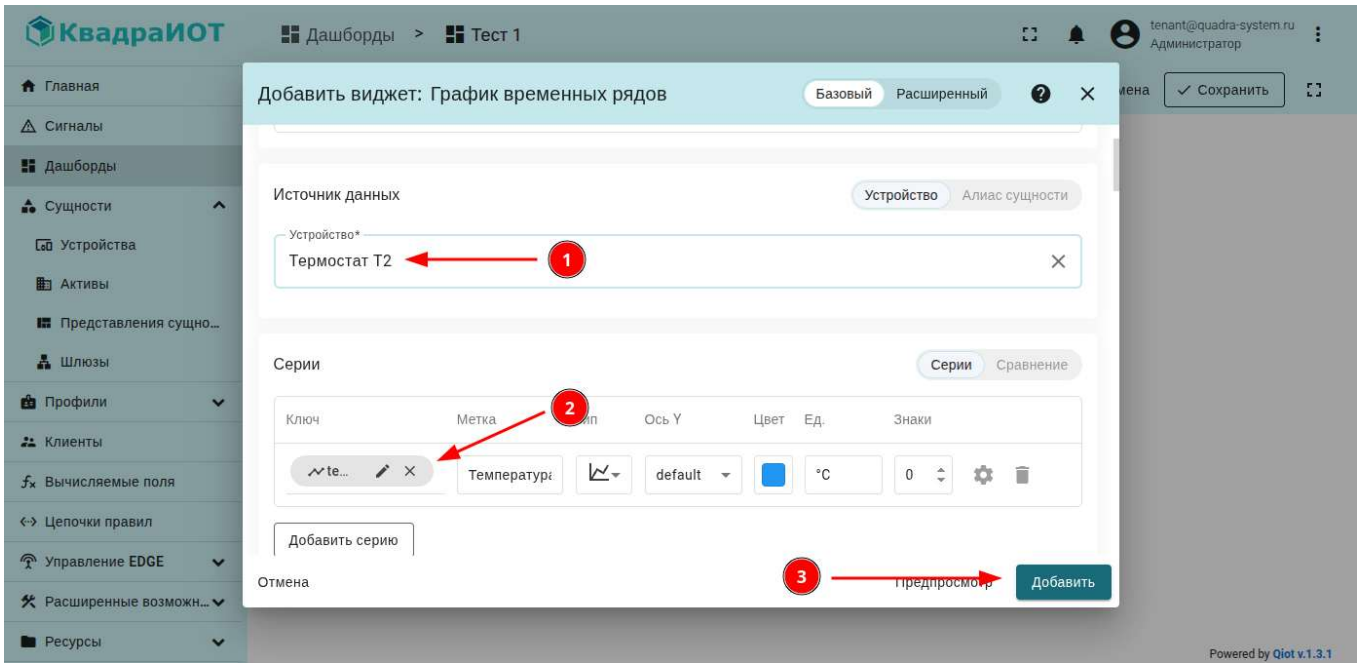


Рисунок 12 – Окно настройка виджета «График временных рядов»

После этого виджет появится на дашборде. При необходимости его можно увеличить, потянув за правый нижний угол. Затем необходимо нажать **«Сохранить»**.

По умолчанию график отображает данные за короткий промежуток времени. Чтобы изменить параметры отображения, необходимо открыть окно настройки временного интервала (Рисунок 13). В нем можно выбрать режим **«Реальное время»** или **«История»** (2), задать **диапазон времени** (3), **функцию агрегации** (4) и **интервал группировки** (5). После изменения параметров необходимо нажать **«Обновить»**.

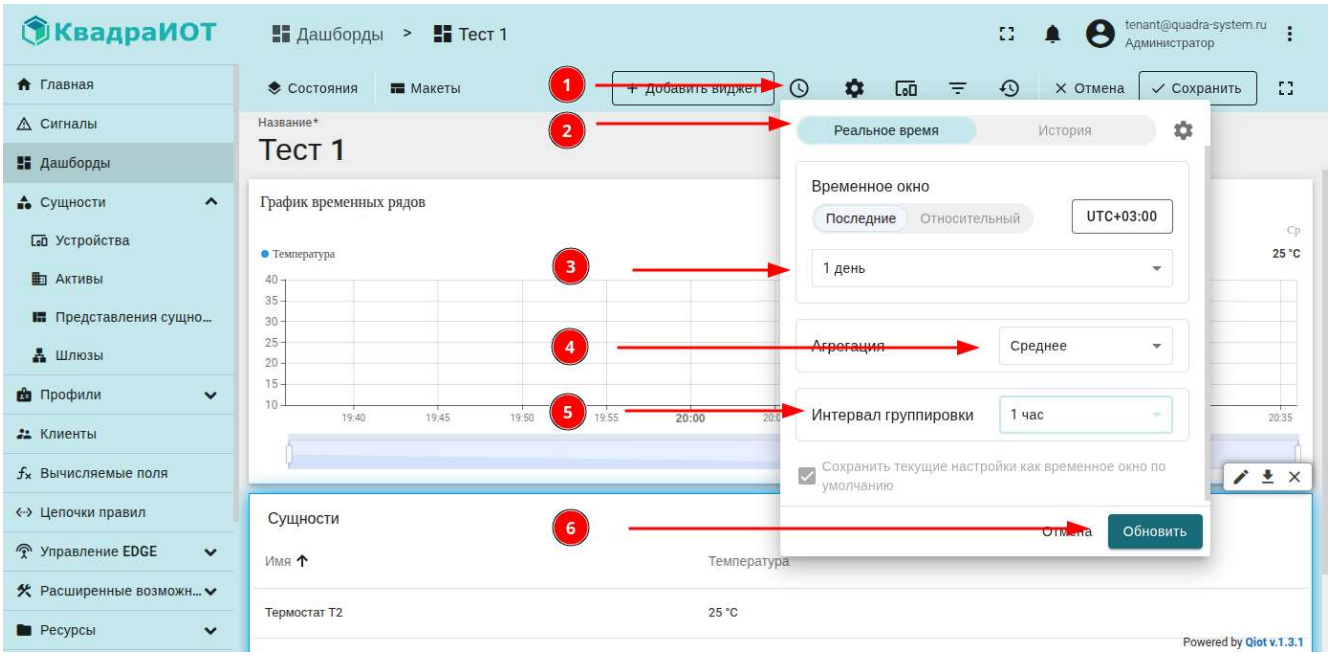


Рисунок 13 – Окно настройки временного интервала виджета «График временных рядов»

2.5 Создание пользователя клиента

Чтобы клиент мог войти в систему, для него необходимо создать отдельного пользователя (Рисунок 14). Для этого необходимо открыть раздел **«Клиенты»** (1), найти нужного клиента и нажать значок управления пользователями клиента. Затем в правом верхнем углу необходимо нажать **«Добавить пользователя»** (2).

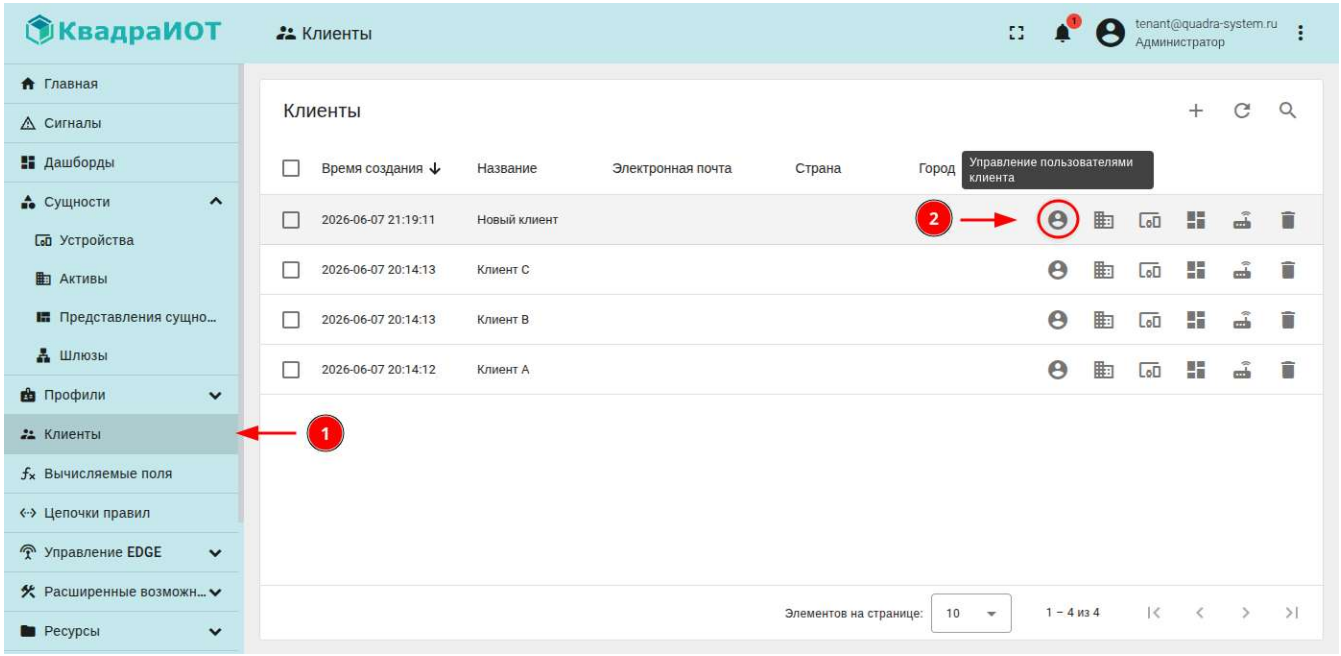


Рисунок 14 – Создание пользователя клиента

В открывшемся окне (Рисунок 15) необходимо указать **данные пользователя (1-5)**, после чего нажать «Добавить».

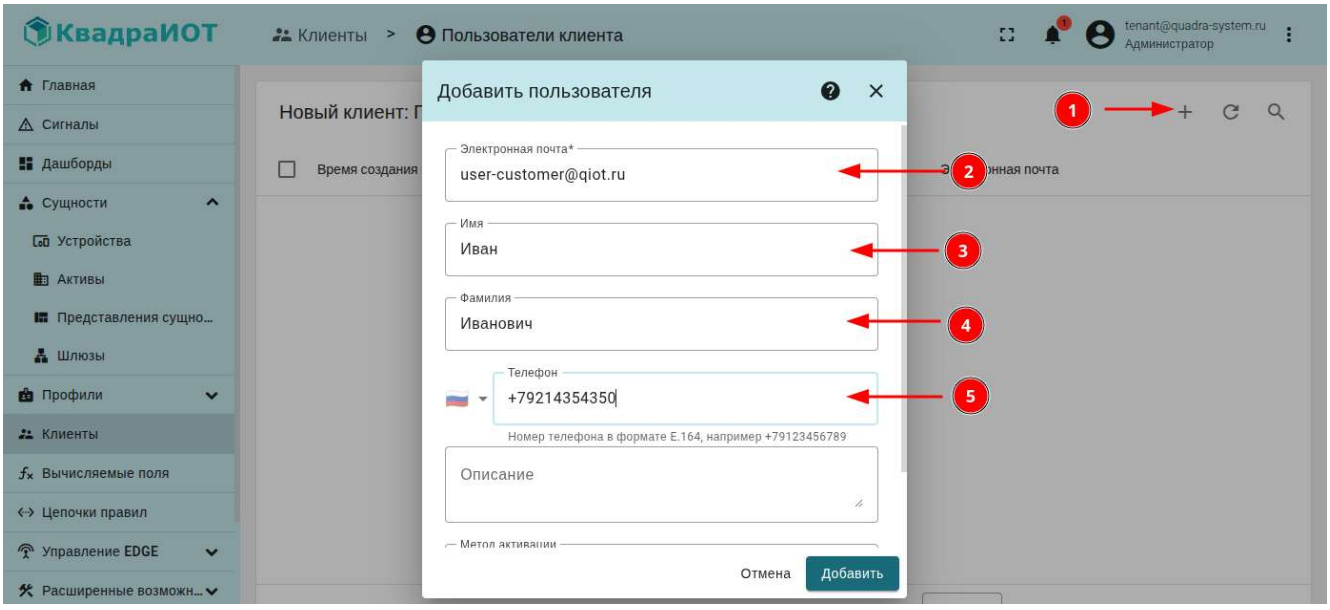


Рисунок 15 – Окно добавления пользователя клиента

После создания система покажет ссылку активации. Ее необходимо скопировать и сохранить, так как она потребуется для установки пароля. После этого следует нажать «ОК».

2.5.1 Настройка стартового дашборда для пользователя клиента

После создания пользователя можно сразу настроить дашборд, который будет открываться у него после входа в систему (Рисунок 16). Для этого необходимо открыть созданного пользователя (1-2), перейти в режим редактирования и в поле «Дашборд по умолчанию» выбрать **«Новый дашборд»** (3). При необходимости можно включить параметр **«Всегда в полноэкранном режиме»** (4). После этого необходимо сохранить изменения.

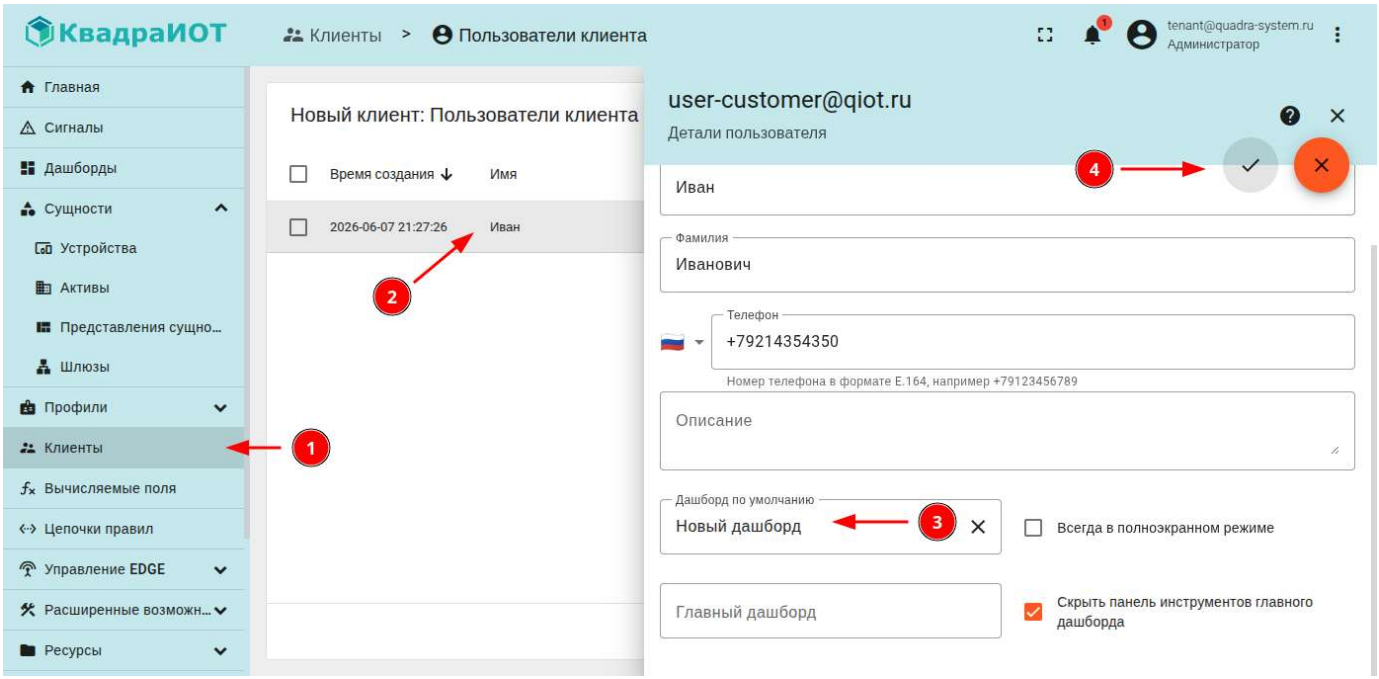


Рисунок 16 – Привязка дашборда к пользователю

2.5.2 Активация пользователя клиента

После создания пользователя клиента необходимо завершить его активацию. Для этого необходимо открыть в браузере ранее сохранённую ссылку активации. После перехода по ссылке откроется форма создания пароля. В ней необходимо дважды ввести новый пароль и нажать кнопку **«Создать пароль»**.

После этого пользователь клиента сможет войти в систему под своей учетной записью. Ему будут доступны назначенные устройства, дашборды и работа с аварийными событиями в пределах выданных прав.

2.6 Назначение устройств и виджетов клиентам

Каждому клиенту программы (группе пользователей) можно назначить для просмотра и доступа свои устройствам, данные и дашборды.

Перед назначением устройства или виджета, необходимо предварительно создать нового клиента (если он не создан ранее).

2.6.1 Добавление нового клиента

Для добавления нового клиента (Рисунок 17) необходимо открыть раздел **«Клиенты (1)»**, затем в правом верхнем углу нажать кнопку **«+» (2)** и указать название клиента (3) в открывшемся окне. После чего, для его сохранения, нажать **«Добавить» (4).**

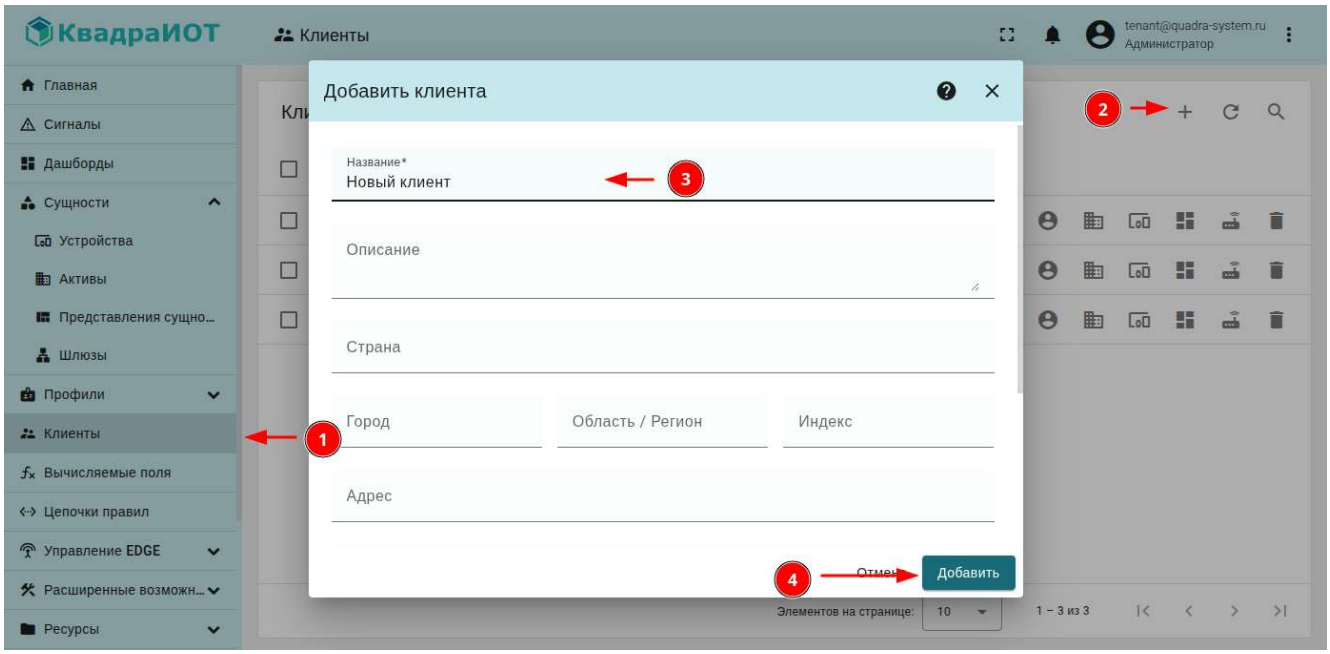


Рисунок 17 – Добавление клиента

После сохранения новый клиент появится в общем списке.

2.6.2 Назначение устройства клиенту

После создания клиента устройство можно привязать к нему. В этом случае клиент получит доступ к данным как устройства, так и связанным с ним элементов.

Для привязки устройства к клиенту (Рисунок 18) необходимо открыть раздел **«Устройства» (1)**, выбрать **требуемое устройство (2)** и в открывшейся карточки устройства нажать кнопку **«Назначить клиенту (3)».**

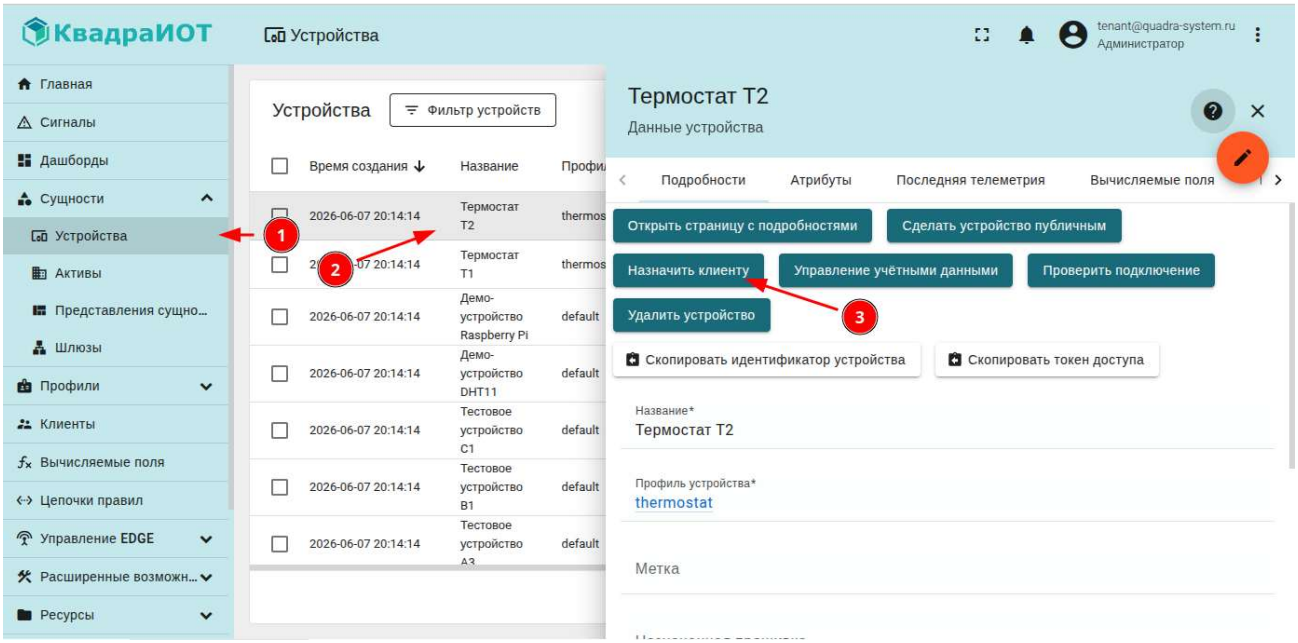


Рисунок 18 – Привязка устройства к клиенту

После этого в таблице устройств клиента в столбце «Клиент» отобразится имя владельца устройства. Проверить назначение устройства клиенту (Рисунок 19) можно также через раздел **«Клиенты»** (1), значок управления **устройствами** (2).

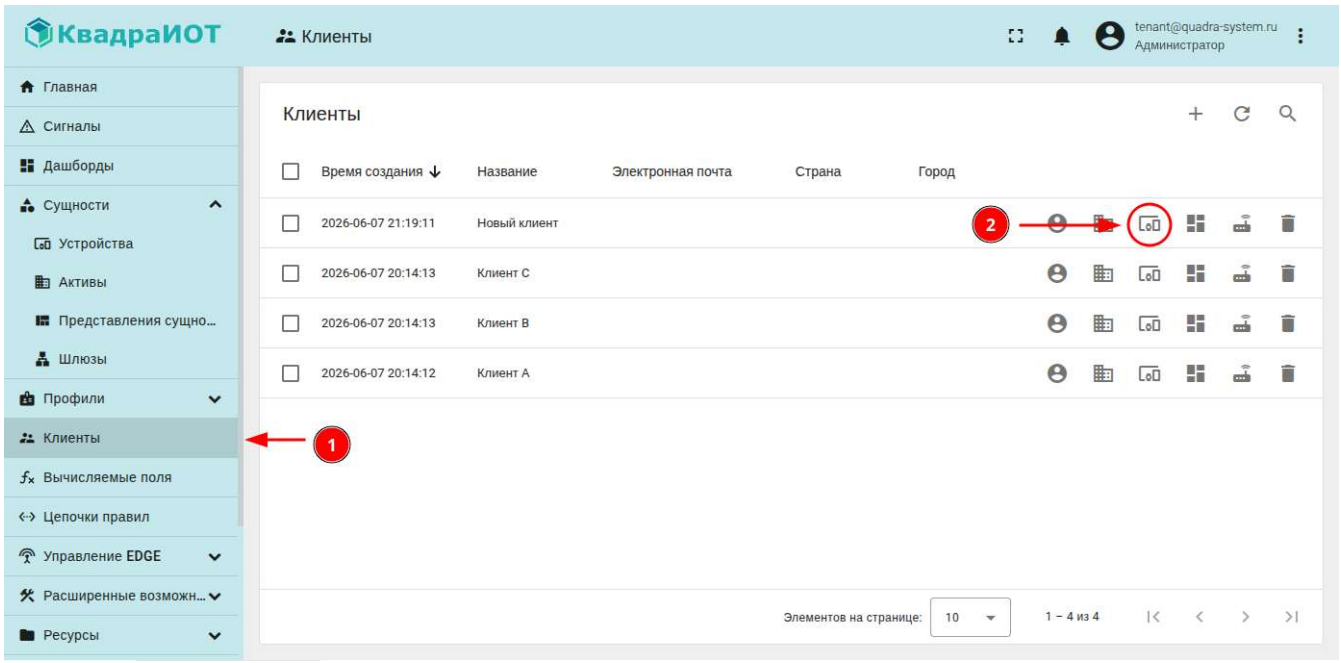


Рисунок 19 – Таблица устройств клиента

2.6.3 Создание устройства сразу с привязкой к клиенту

Устройство можно сразу создать уже привязанным к клиенту.

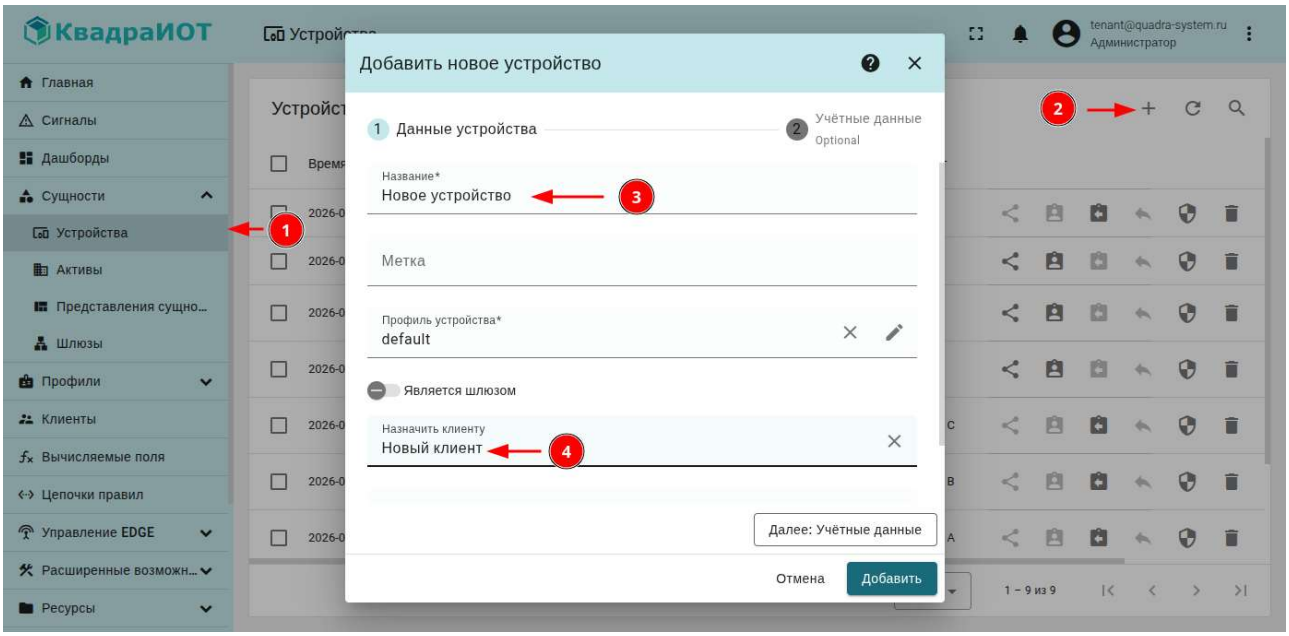


Рисунок 20 – Привязка устройства к клиенту

Для этого необходимо открыть раздел «Устройства», нажать «+», выбрать «Добавить новое устройство», указать имя устройства и в поле назначения выбрать нужного клиента. После этого необходимо нажать «Добавить» и закрыть окно проверки подключения.

После сохранения новое устройство сразу появится в списке и будет закреплено за выбранным клиентом.

2.6.4 Назначение дашборда клиенту

После создания дашборда его можно назначить клиенту. В этом случае клиентские пользователи смогут просматривать этот дашборд в режиме чтения.

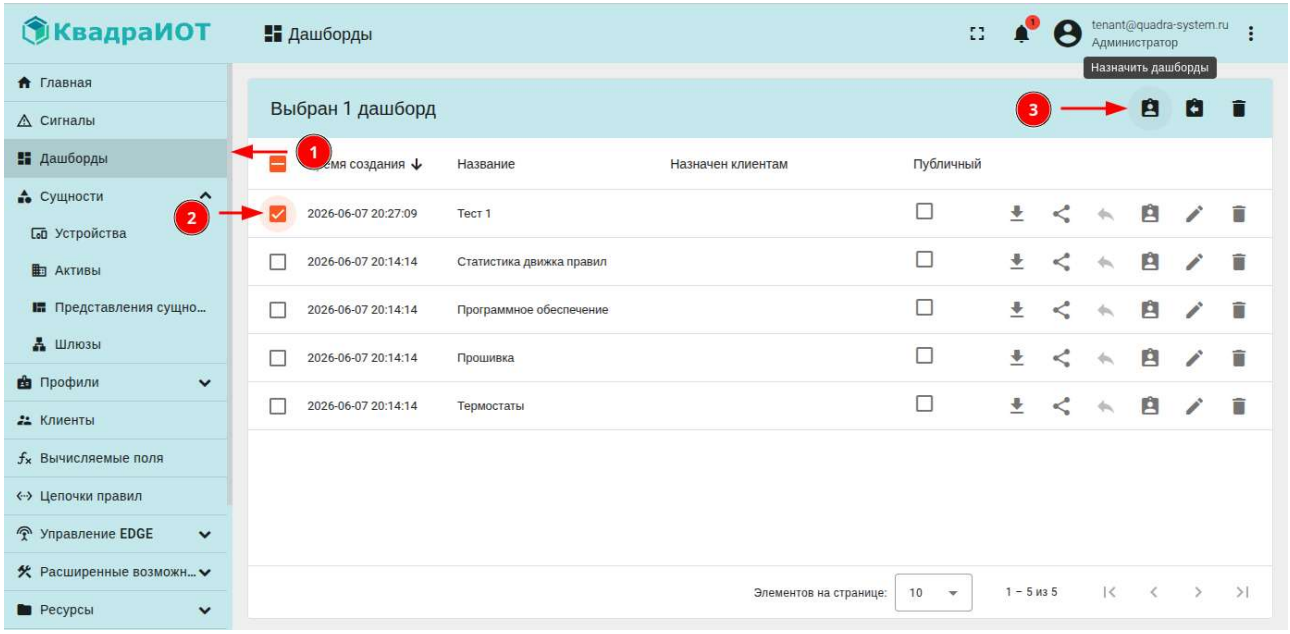


Рисунок 21 – Привязка дашборда к клиенту

Для этого необходимо открыть раздел «Дашборды», отметить нужный дашборд и нажать кнопку «Назначить дашборды». В открывшемся окне следует выбрать клиента и нажать «Назначить» (Рисунок 26 - № 1, 2, 3).

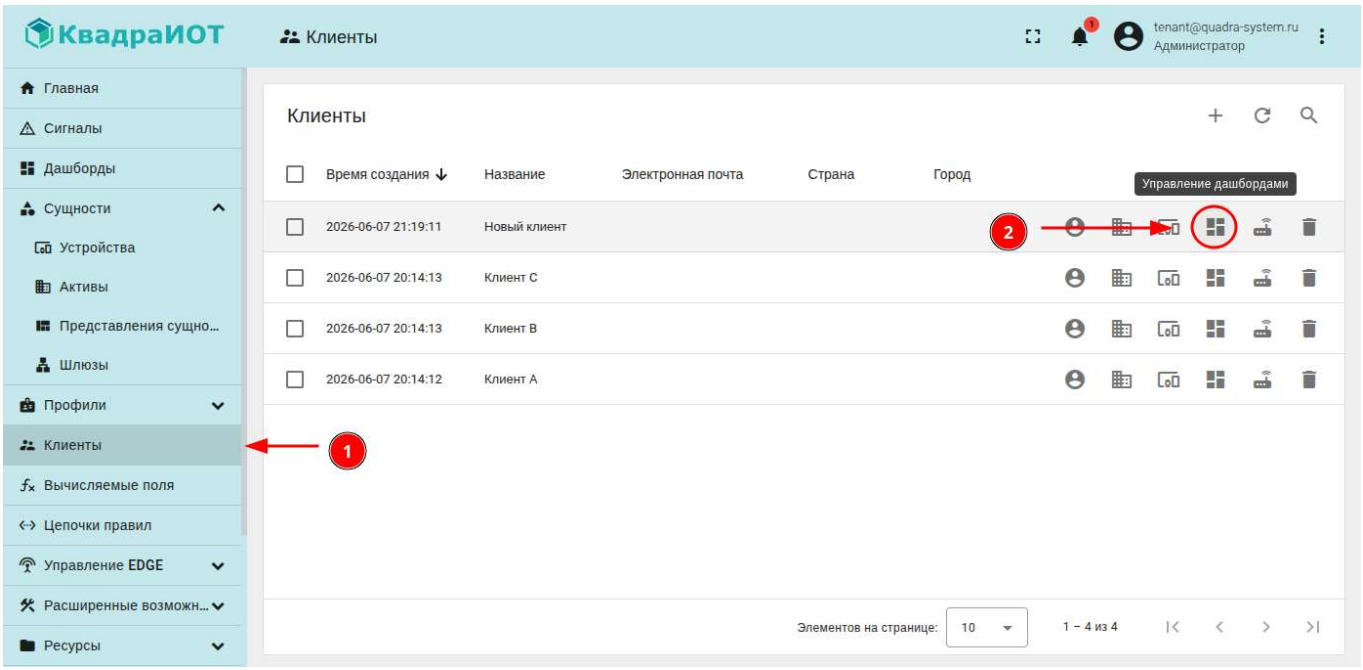


Рисунок 22 – Просмотр дашбордов клиента

Проверить результат можно через раздел «Клиенты». Для этого необходимо найти нужного клиента и нажать значок управления дашбордами. После этого откроется список дашбордов, назначенных данному клиенту (Рисунок 27 - № 1, 2).

2.7 Создание правил сигналов

Правила сигналов позволяют автоматически фиксировать превышение заданных значений.

Правило можно создавать как для отдельного устройства, так и на уровне профиля устройства. Если одно и то же условие должно применяться сразу к нескольким устройствам, необходимо настраивать его через профиль.

Для настройки правила сигнала для одного устройства (Рисунок 23) необходимо открыть раздел **«Сущности»→«Устройства»** (1), выбрать устройство, например, **Новое устройство (2)**, затем перейти на вкладку **«Правило сигналов» (3)**. После этого следует нажать кнопку **«+»** и выбрать пункт **«Создать новое правило» (4)**.

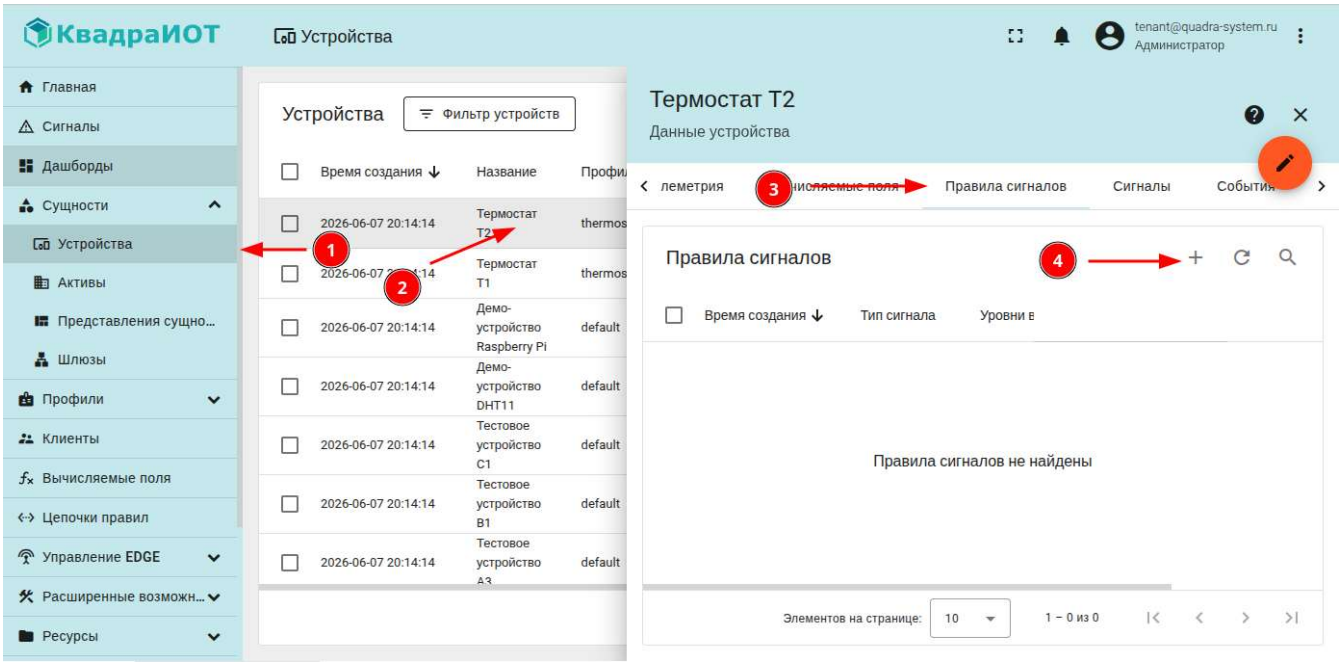


Рисунок 23 – Настройка правила события устройства

В открывшемся окне (Рисунок 24) в поле **«Тип сигнала» (1)** необходимо указать его название, например Высокая температура. Далее необходимо добавить аргумент, который будет использоваться в условии его фиксации, для чего в разделе аргументов необходимо нажать **«Добавить аргумент» (2)** и указать:

- **тип сущности** (например, текущая сущность);
- **тип аргумента** (например, последняя телеметрия);
- **ключ временного ряда** (например, temperature);
- **имя аргумента** (например, temperature).

После этого необходимо подтвердить создания правила сигнала, нажав кнопку **«Добавить»**.

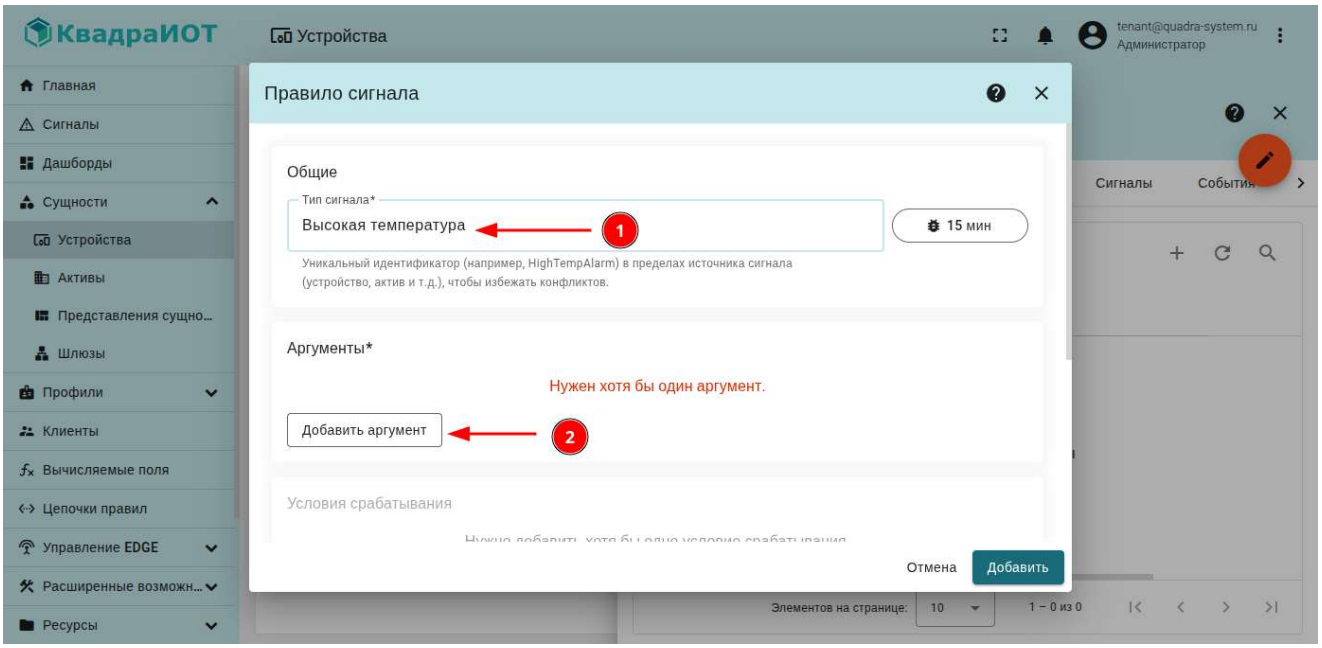


Рисунок 24 – Добавление правила сигнала

Для добавления условия срабатывания сигнала (Рисунок 25) необходимо нажать **«Добавить условие срабатывания» (1).**

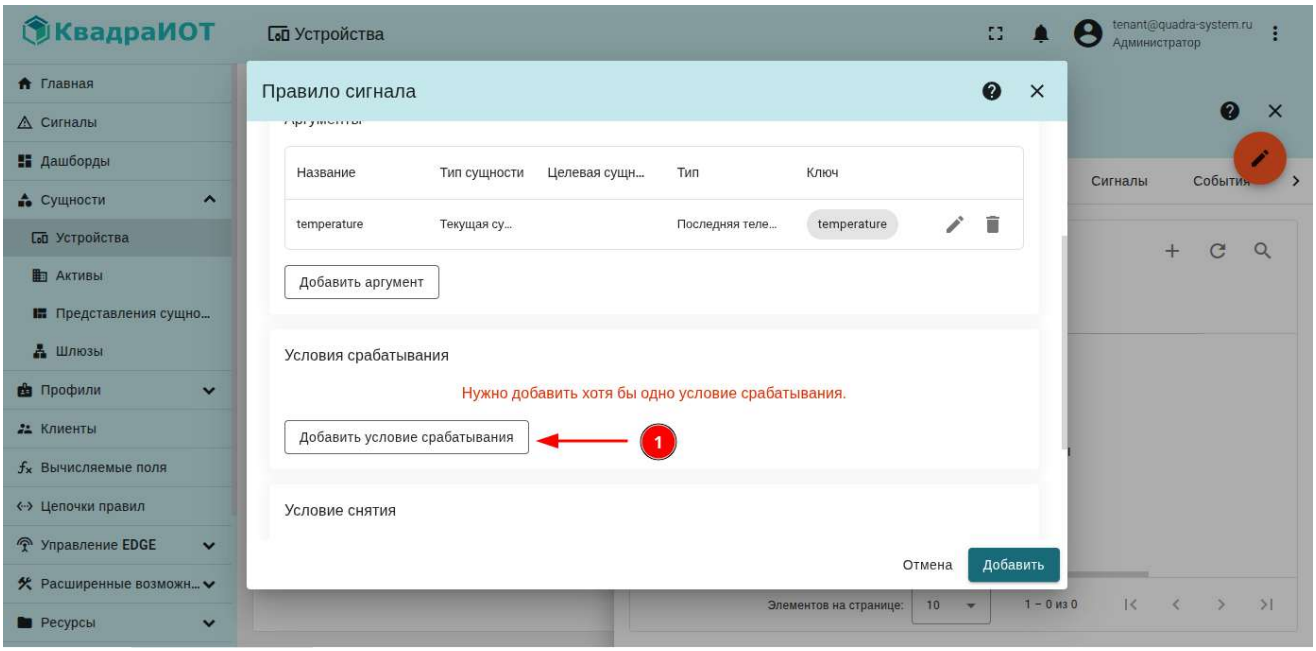


Рисунок 25 – Добавление условия срабатывания сигнала

В открывшемся окне срабатывания сигнала (Рисунок 26) необходимо заполнить поля **«Уровень важности»** (1), **«Условие»** (2), **«Расписание»** (3) и «Дополнительная информация» (при необходимости).

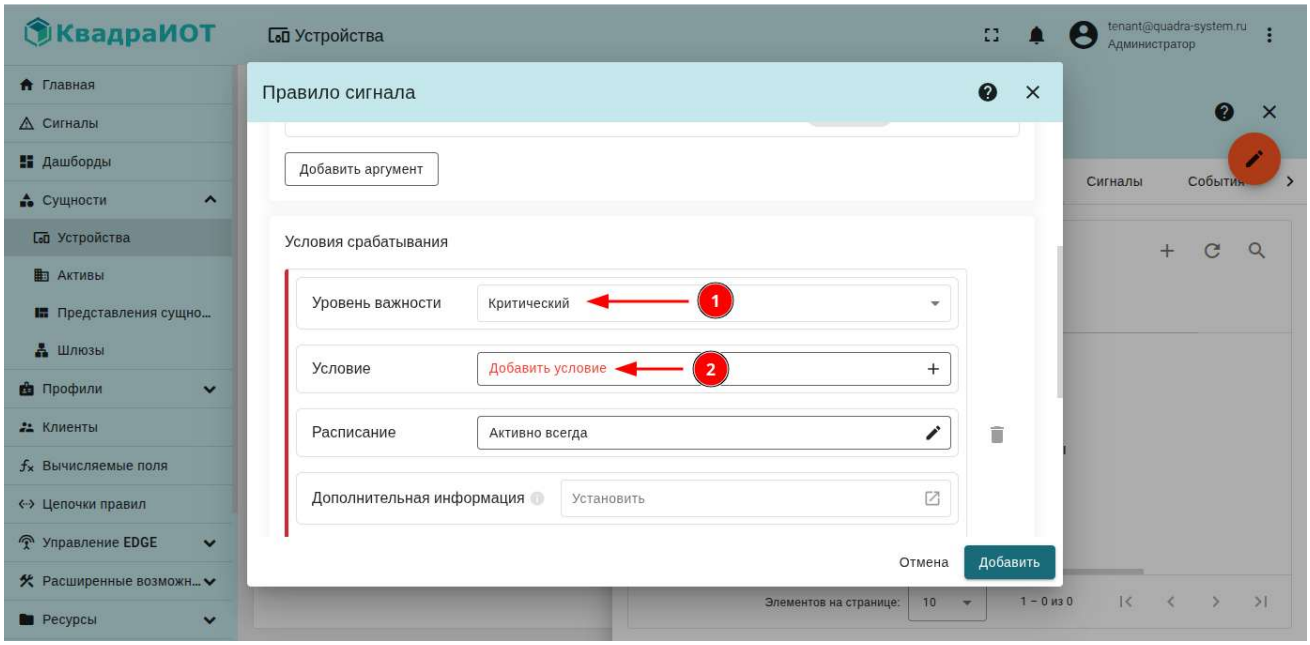


Рисунок 26 – Настройка условия срабатывания сигнала

При заполнении поля «Условие» необходимо добавить фильтр аргумента (Рисунок 27) **«Добавить фильтр аргумента»**.

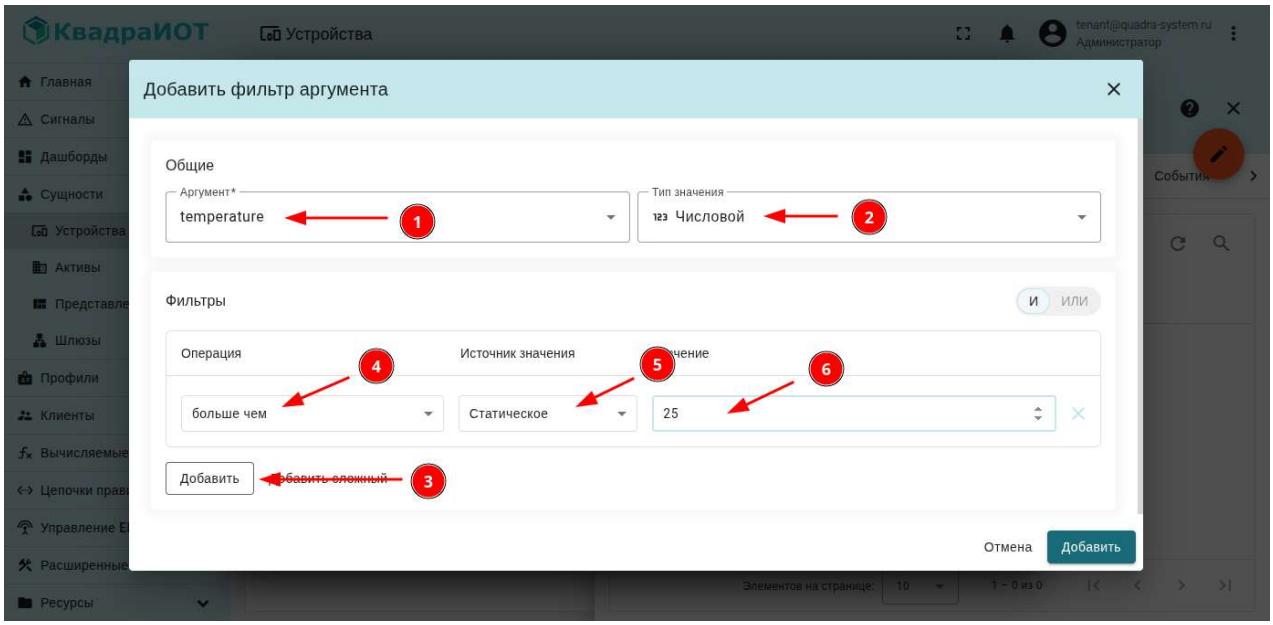


Рисунок 27 – Добавление фильтра аргумента

В настройках фильтра необходимо указать:

- **аргумент** (1), например temperature;
- **тип значения** (2), например, числовой;
- **операция** (4), например, больше чем;
- **источник значения** (5), например, статическое;
- **значение** (6), например 25.

После этого необходимо нажать **«Добавить»**, выбрать тип условия **«Простой»**, нажать **«Сохранить»**, а затем еще раз **«Добавить»** для сохранения правила. После сохранения правило сразу активируется.

При необходимости аварийное событие можно подтвердить (значок **галочки**) или снять (значок **крестика**) прямо в таблице:

После создания аварии система также отправляет уведомление в **центр уведомлений**. Для просмотра уведомления необходимо нажать на значок **колокольчика** в правом верхнем углу интерфейса.

2.8 Настройка цепочек правил

Цепочка правил (Рисунок 28) позволяет создавать сложные сценарии, выходящие за рамки одиночных правил сигналов, и представляет собой последовательность логических блоков/узлов (Таблица 1), соединенных отношениями/связями (Таблица 2).

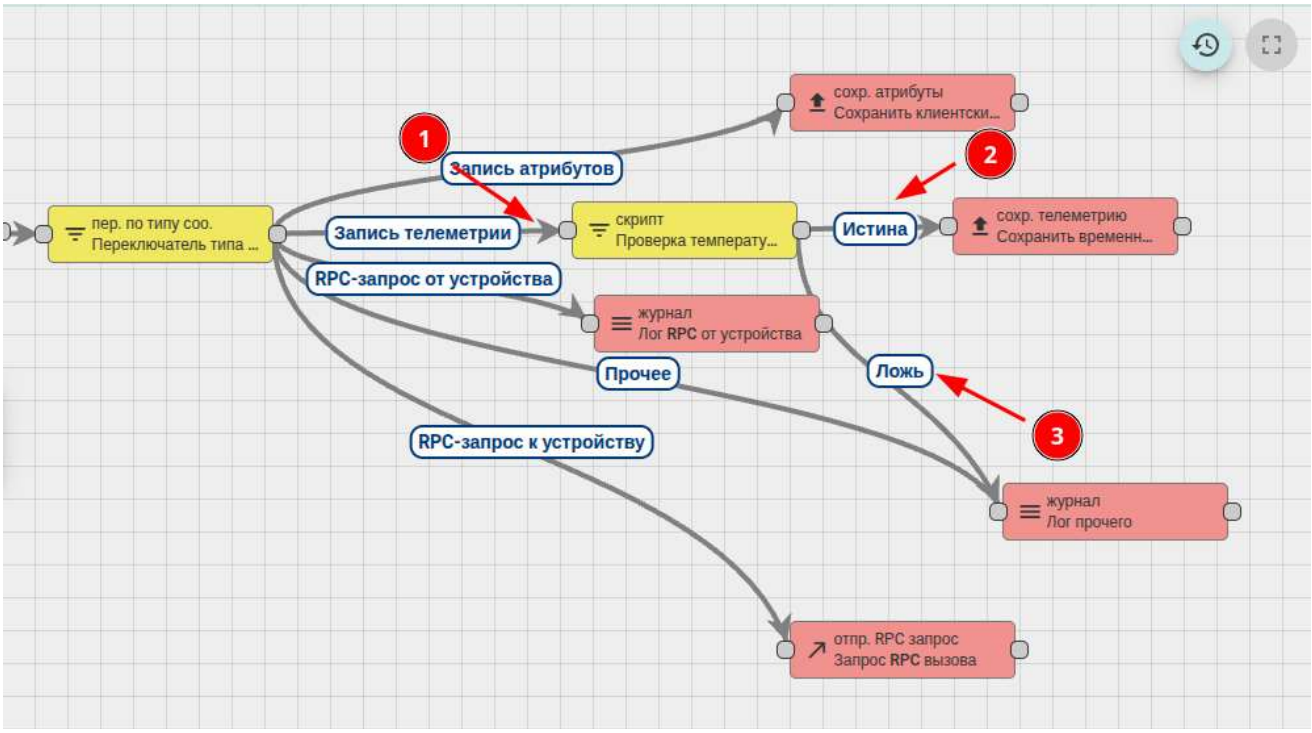


Рисунок 28 – Цепочка правил

Таблица 1 Перечень типов узлов, участвующих в создании цепочек правил

Тип узла	Цветовое обозначение	Назначение
Фильтрация		Проверка входящих сообщений и выбор дальнейшего направления их обработки
Обогащение (дополнение)		Добавление к сообщению данных
Преобразование		Изменение сообщения, его содержимое, структуру, метаданные или источник
Действие		Выполнение операций с данными или объектами
Внешние		Рассылка данных во внешние сервисы
Потоки		Управление движением сообщений между цепочками правил и очередями обработки

Таблица 2 Перечень типов связей, участвующих в создании цепочек правил

Тип связи	Назначение
Общие и служебные связи	Определяют общий результат обработки сообщения и выбирают дальнейшую ветку цепочки: успешно, ошибка, истина, ложь, прочее или пользовательский выход.
Получение данных и команд от устройств	Разделяют входящие сообщения устройства на атрибуты, телеметрию и RPC-запросы.
Состояние и подключение устройств	Передают события активности, неактивности, подключения и отключения устройств.
Жизненный цикл и назначение сущностей	Обрабатывают создание, изменение, удаление, назначение, смену владельца и участие сущностей в группах.
Атрибуты и временные ряды	Сообщают об обновлении или удалении атрибутов и телеметрических временных рядов.
Тревоги	Отслеживают создание, изменение, подтверждение, назначение и снятие тревог.
Комментарии к тревогам	Передают события создания и изменения комментариев к тревогам.

Тип связи	Назначение
Состояния RPC-команд	Отражают этапы выполнения RPC-команды: постановку в очередь, отправку, доставку, успешное выполнение, тайм-аут, ошибку, просрочку или удаление.
Связи между сущностями	Обрабатывают создание, изменение и удаление связей между сущностями платформы.
Внешние и системные события	Передают запросы REST API и другие системные события, например запуск формирования отчёта.

2.8.1 Описание блоков

2.8.1.1 ФИЛЬТРАЦИЯ

Узлы фильтрации (Таблица 3) используются для проверки входящих сообщений и выбора дальнейшего направления их обработки. Они не сохраняют данные и не изменяют их напрямую, а определяют, должно ли сообщение пройти дальше, быть отклонено или перейти в определённую ветку цепочки.

Таблица 3 Перечень и назначение узлов фильтрации

Наименование	Назначение
пер. источника	Направляет сообщение по разным веткам в зависимости от источника сообщения. Используется, когда необходимо разделить обработку сообщений от разных типов объектов.
пер. по скрипту	Направляет сообщение в одну или несколько веток по условию, заданному в скрипте. Используется для гибкой маршрутизации.
пер. по типу соо.	Разделяет сообщения по их типу и отправляет их в разные ветки цепочки. Например, телеметрия идёт в сохранение временных рядов, атрибуты - в сохранение атрибутов, RPC-запросы - в отдельную обработку.
пер. проф. акт.	Направляет сообщение по разным веткам в зависимости от профиля актива. Например, для разных типов объектов можно настроить отдельную обработку.
пер. проф. устр.	Разделяет сообщения по профилю устройства. Например, сообщения от термостата можно отправить в одну ветку, а сообщения от датчика влажности - в другую.

Наименование	Назначение
проверка полей	Проверяет, есть ли в сообщении или его метаданных нужные поля. Например, можно проверить, пришло ли значение температуры.
проверка связи	Проверяет, существует ли связь между текущим объектом и другой сущностью. Например, связано ли устройство с активом, клиентом или другим объектом.
скрипт	Выполняет пользовательское условие через скрипт. Если условие истинное, сообщение идёт по одной ветке, если нет - по другой.
статус тревоги	Проверяет статус тревоги. Например, активна ли тревога, подтверждена ли она или уже очищена. Используется, когда необходимо обрабатывать только тревоги с определённым состоянием.
фильтр источника	Проверяет источник сообщения и пропускает только те сообщения, которые соответствуют заданному условию.
фильтр типа сообщ.	Пропускает только сообщения определённого типа. Например, только телеметрию, только атрибуты или только RPC-запросы.
GPS геофильтр	Проверяет координаты объекта и определяет, находится ли он внутри заданной геозоны. Используется для объектов с GPS-данными.

2.8.1.2 ОБОГАЩЕНИЕ (ДОПОЛНЕНИЕ)

Узлы обогащения (Таблица 4) добавляют к сообщению дополнительные данные: атрибуты, телеметрию, сведения об устройстве, клиенте, владельце системы или связанных объектах. Они нужны, чтобы перед дальнейшей обработкой сообщение содержало больше информации.

Таблица 4 Перечень и назначение узлов обогащения (дополнения)

Наименование	Назначение
арт. источника	Добавляет атрибуты объекта, который отправил сообщение. Например, если сообщение пришло от устройства, блок может добавить его серверные, клиентские или общие атрибуты.

Наименование	Назначение
арт. клиента	Добавляет к сообщению атрибуты клиента, к которому относится устройство или объект. Используется, когда для дальнейшей обработки нужны данные клиента.
арт. тенанта	Добавляет атрибуты владельца системы. Например, общие настройки, пороговые значения или параметры, которые применяются ко многим устройствам.
арт. устр-ва	Добавляет атрибуты или последнюю телеметрию устройства, которое связано с текущим объектом. Например, можно получить данные контроллера, связанного с датчиком.
вычислить дельту	Сравнивает текущее значение с предыдущим и рассчитывает разницу. Например, можно определить, насколько изменилась температура по сравнению с прошлым показанием.
детали клиента	Добавляет сведения о клиенте: название, email, адрес и другие данные профиля. Например, эти данные можно использовать при отправке уведомлений.
детали тенанта	Добавляет сведения о владельце системы: имя, email, адрес и другие данные профиля.
поля источника	Добавляет основные сведения об источнике сообщения: имя, метку, профиль и другие поля объекта.
связ. данные	Добавляет данные связанного объекта: атрибуты, телеметрию или поля. Используется, когда обработка зависит не только от текущего устройства, но и от связанных с ним объектов.
телем. источника	Добавляет к сообщению исторические значения телеметрии источника. Например, можно получить прошлое значение температуры для сравнения с текущим.
учетка устр.	Добавляет к сообщению данные для идентификации устройства. Используется, когда в дальнейшей обработке нужны сведения об учётных данных устройства.

2.8.1.3 ПРЕОБРАЗОВАНИЕ

Узлы преобразования (Таблица 5) изменяют само сообщение: его содержимое, структуру, метаданные или источник. Они нужны, когда перед

сохранением или отправкой данные необходимо подготовить: переименовать, разделить, удалить лишнее или преобразовать под нужный формат.

Таблица 5 Перечень и назначение узлов преобразования

Наименование	Назначение
дедупликация	Объединяет или отбирает сообщения от одного источника за заданный промежуток времени. Может пропустить первое сообщение, последнее или объединённый результат.
копир. ключи	Копирует выбранные поля между данными сообщения и метаданными. Например, значение из сообщения можно перенести в метаданные, чтобы другие блоки могли его использовать.
переим. ключи	Переименовывает поля в данных сообщения или метаданных. Например, temp можно заменить на temperature, чтобы привести данные к единому виду.
разделить массив	Разделяет JSON-массив на отдельные сообщения. Например, если пришёл массив из нескольких измерений, каждое измерение можно отправить дальше как отдельное сообщение.
скрипт	Выполняет пользовательский скрипт и изменяет данные сообщения, метаданные или тип сообщения. Может создать одно новое сообщение или несколько сообщений.
смена источника	Меняет источник сообщения на другую сущность. Например, если данные пришли от шлюза, но относятся к конкретному устройству, блок может сделать источником именно это устройство.
сооб. в письмо	Подготавливает сообщение для отправки по электронной почте: формирует отправителя, получателя, тему и текст письма.
удал. ключи	Удаляет ненужные поля из данных сообщения или метаданных. Используется, чтобы очистить сообщение перед дальнейшей обработкой.
json path	Извлекает нужную часть данных из JSON-сообщения по заданному пути и заменяет ею исходное содержимое сообщения.

2.8.1.4 ДЕЙСТВИЯ

Узлы действий (Таблица 6) выполняют конкретные операции с данными или объектами системы: сохраняют телеметрию и атрибуты, создают или очищают тревоги, отправляют RPC-команды, пишут данные в журнал, назначают объекты клиентам и изменяют состояние системы.

Таблица 6 Перечень и назначение узлов действий

Наименование	Назначение
вычисл. поля	Запускает обработку вычисляемых полей для входящих данных без сохранения исходного сообщения в базу.
генератор сообщ.	Создаёт сообщения через заданные интервалы времени с помощью скрипта. Используется для тестовых или служебных сообщений.
журнал	Формирует запись и сохраняет её в журнал системы. Обычно используется для отладки цепочек правил.
задержка (устарело)	Временно задерживает сообщение перед дальнейшей отправкой. Может использоваться для паузы между этапами обработки.
конец синх.	Завершает этап синхронизации сообщения и передаёт результат дальше по цепочке.
копир. в просмотр	Копирует изменения атрибутов устройства или актива в связанное представление сущности.
мат. функция	Выполняет математические операции над значениями из сообщения, метаданных, атрибутов или телеметрии.
назн. клиенту	Назначает источник сообщения определённому клиенту.
начало синх.	Запускает этап синхронизации сообщения для дальнейшей передачи или обработки.
отпр. на edge	Передаёт сообщение из облачного экземпляра системы на один или несколько экземпляров Edge.
отпр. RPC запрос	Отправляет RPC-команду устройству и возвращает ответ устройства как данные исходящего сообщения.
отпр. RPC ответ	Отправляет данные сообщения как ответ на RPC-запрос устройства.

Наименование	Назначение
профиль устройства...	Проверяет правила тревог, заданные в профиле устройства, и может создать, обновить или очистить тревогу.
снять с клиента	Убирает источник сообщения из текущего назначения клиенту.
снять тревогу	Очищает активную тревогу для источника сообщения. Например, если температура вернулась в норму, блок закрывает тревогу о перегреве.
создать связь	Создаёт связь между источником сообщения и другой выбранной сущностью.
создать тревогу	Создаёт новую тревогу или обновляет уже существующую активную тревогу для источника сообщения.
сост. устройства	Изменяет состояние подключения устройства: активно, подключается или отключено.
сохр. атрибуты	Сохраняет данные сообщения как атрибуты источника. Например, режим работы, порог температуры или состояние устройства.
сохр. в quadradb	Сохраняет данные сообщения в пользовательскую таблицу базы данных по заданному соответствию полей.
сохр. телеметрию	Сохраняет данные сообщения как телеметрию источника. Например, температуру, влажность, давление или напряжение.
счетчик сообщ.	Считает количество сообщений за заданный промежуток времени и передаёт результат дальше как телеметрию.
удал. атрибуты	Удаляет указанные атрибуты у источника сообщения.
удалить связь	Удаляет связь между источником сообщения и выбранной сущностью.
GPS геособытия	Проверяет координаты объекта относительно заданной геозоны и определяет события входа, выхода или нахождения внутри зоны.
rest call reply	Отправляет данные сообщения как HTTP-ответ на REST-запрос, который ранее был передан в цепочку правил.

2.8.1.5 ВНЕШНИЕ УЗЛЫ

Внешние узлы (Таблица 7) нужны для отправки данных из цепочки правил во внешние сервисы: почту, SMS, облачные сервисы и т.д.

Таблица 7 Перечень и назначение внешних узлов

Наименование	Назначение
отпр. письмо	Отправляет письмо через SMTP. Обычно используется после блока преобразования в email, где заранее формируются получатель, тема и текст письма.
отпр. уведомл.	Отправляет уведомление через встроенный центр уведомлений системы. Может использоваться для оповещения пользователей о событиях или тревогах.
отпр. SMS	Отправляет SMS-сообщение через настроенного провайдера. Используется для срочных уведомлений, например при аварии или превышении порога.
kafka	Отправляет данные сообщения в топики Apache Kafka. Используется для передачи больших потоков данных во внешние системы обработки.
mqtt	Публикует данные сообщения во внешний MQTT-брокер. Например, можно передать телеметрию в другую IoT-систему.
rabbitmq	Отправляет данные сообщения в обменник RabbitMQ. Используется для передачи сообщений между сервисами через очередь.
rest api call	Отправляет HTTP-запрос во внешний сервис. Например, можно передать данные на сторонний сервер, вызвать вебхук или получить ответ от внешнего API.

2.8.1.6 ПОТОКИ

Узлы потоков (Таблица 8) управляют движением сообщений между цепочками правил и очередями обработки. Они позволяют подтверждать обработку сообщения, передавать его в другую очередь, направлять в отдельную цепочку правил или возвращать результат обратно в основную цепочку.

Таблица 8 Перечень и назначение узлов потоков

Наименование	Назначение
вход цепочки	Передаёт сообщение в другую выбранную цепочку правил. Используется для разделения большой логики на несколько отдельных цепочек, чтобы основная схема была проще и понятнее.
выход цепочки	Возвращает результат обработки из вложенной цепочки обратно в основную цепочку правил. Используется вместе с входом цепочки, когда после обработки необходимо продолжить работу в исходной схеме.
контр. точка	Передаёт копию сообщения в выбранную очередь для отдельной или последовательной обработки. Используется, когда разные типы сообщений необходимо обрабатывать с разным приоритетом или в отдельном порядке.
подтверждение	Подтверждает, что сообщение успешно обработано в очереди, и передаёт его дальше по цепочке. Используется, когда необходимо явно отметить сообщение как обработанное.

2.8.2 Описание связей

2.8.2.1 ОБЩИЕ И СЛУЖЕБНЫЕ СВЯЗИ

Общие и служебные связи (Таблица 9) используются для управления направлением движения сообщения между узлами цепочки правил. Они позволяют отделить штатную обработку от ошибок, проверить выполнение заданного условия, направить неизвестные сообщения в отдельную ветку или вернуть результат из дочерней цепочки. Благодаря этим связям для каждого возможного результата работы узла можно настроить собственный последующий сценарий обработки.

Таблица 9 Перечень и описание общих и служебных связей

Наименование	Описание
Успешно	Узел успешно обработал сообщение. Сообщение передаётся в следующую ветку цепочки.

Наименование	Описание
Ошибка	При обработке сообщения возникла ошибка. Связь используется для перехода к журналированию, резервной логике или обработчику ошибок.
Истина	Условие фильтра выполнено. Сообщение направляется по положительной ветке.
Ложь	Условие фильтра не выполнено. Сообщение направляется по отрицательной ветке.
Прочее	Тип сообщения не относится к предопределённым типам или является пользовательским.
Тайм-аут	Обработка не завершилась за установленное время. Может использоваться как отдельный результат в узлах, поддерживающих контроль времени выполнения.
Пользовательская связь	Связь с произвольным названием, созданным в переключателе, скрипте или другой настраиваемой логике, например «Норма», «Предупреждение» или «Опасность».
Выход дочерней цепочки	Возвращает сообщение из вложенной цепочки в родительскую. Название связи совпадает с названием выходного узла дочерней цепочки.

2.8.2.2 ПОЛУЧЕНИЕ ДАННЫХ И КОМАНД ОТ УСТРОЙСТВ

Связи получения данных и команд от устройств (Таблица 10) предназначены для разделения сообщений, которыми устройство обменивается с платформой КИоТ. С их помощью цепочка правил определяет, поступили ли от устройства атрибуты, телеметрические показания или RPC-запрос. Такое разделение необходимо, поскольку каждый вид сообщения требует собственной логики: телеметрия может сохраняться и проверяться по пороговым значениям, атрибуты

- изменять параметры устройства, а RPC-запросы - вызывать выполнение удалённых команд.

Таблица 10 Перечень и описания связей получения данных и команд от устройств

Наименование	Описание
Запись атрибутов	Устройство передало значения атрибутов, например версию программного обеспечения, режим работы или параметры конфигурации.
Запись телеметрии	Устройство передало телеметрические данные: показания датчиков, измерения, счётчики или другие временные значения.
RPC-запрос от устройства	Устройство отправило серверу удалённый вызов процедуры и ожидает обработку или ответ.
RPC-запрос к устройству	Сервер или цепочка правил отправляет устройству удалённую команду.

2.8.2.3 СОСТОЯНИЕ И ПОДКЛЮЧЕНИЕ УСТРОЙСТВ

Связи состояния и подключения устройств (Таблица 11) применяются для контроля доступности и текущего состояния устройств. Они позволяют цепочке правил реагировать на подключение и отключение оборудования, а также определять переход устройства в активное или неактивное состояние. На основании таких событий могут формироваться тревоги, уведомления о потере связи, записи в журнале и команды для проверки работоспособности оборудования.

Таблица 11 Перечень и описание связей состояния и подключения устройств

Наименование	Описание
Событие активности	Устройство перешло в активное состояние после получения данных или другого признака активности.
Событие неактивности	Устройство признано неактивным, поскольку данные от него не поступали в течение установленного периода.
Событие подключения	Устройство установило соединение с платформой.

Наименование	Описание
Событие отключения	Устройство разорвало соединение с платформой или транспортным сервисом.

2.8.2.4 ЖИЗНЕННЫЙ ЦИКЛ И НАЗНАЧЕНИЕ СУЩНОСТЕЙ

Связи жизненных циклов и назначений сущностей (Таблица 12) используются для обработки изменений, происходящих с сущностями платформы: устройствами, активами, пользователями, дашбордами и другими объектами. Они позволяют автоматически реагировать на создание, изменение, удаление, назначение или переназначение сущности, смену владельца и изменение состава групп. Это необходимо для синхронизации данных, управления правами доступа, обновления связанных объектов и выполнения действий при изменении структуры системы.

Таблица 12 Перечень и описания связей жизненного цикла и назначения сущностей

Наименование	Назначение
Сущность создана	В платформе создано устройство, актив, пользователь, дашборд или другая сущность.
Сущность обновлена	Изменены основные сведения, настройки или свойства существующей сущности.
Сущность удалена	Сущность удалена из платформы.
Сущность назначена	Сущность назначена заказчику, пользователю или другой области ответственности.
Сущность снята с назначения	Ранее выполненное назначение сущности отменено.
Сущность назначена от арендатора	Событие передачи или переназначения сущности от арендатора. Точное поведение зависит от модели владения и версии платформы.
Сущность назначена арендатору	Сущность передана или назначена арендатору. В локализованной версии КИоТ термин может соответствовать пользователю или владельцу.
Владелец изменён	У сущности изменён владелец или область владения.

Наименование	Назначение
Добавлено в группу	Сущность добавлена в группу сущностей.
Удалено из группы	Сущность исключена из группы сущностей.

2.8.2.5 АТРИБУТЫ И ВРЕМЕННЫЕ РЯДЫ

Связи атрибутов и временных рядов (Таблица 13) предназначены для контроля изменений атрибутов и телеметрических данных сущностей. Атрибуты обычно содержат настройки, характеристики и служебные параметры объекта, а временные ряды - значения, изменяющиеся во времени, например температуру, давление или расход. Связи позволяют запускать дополнительную обработку после добавления, изменения или удаления таких данных, например пересчёт показателей, проверку допустимых значений или передачу информации во внешнюю систему.

Таблица 13 Перечень и описание связей атрибутов и временных рядов

Наименование	Описание
Атрибуты обновлены	Значения атрибутов сущности добавлены или изменены.
Атрибуты удалены	Один или несколько атрибутов сущности удалены.
Временные ряды обновлены	Телеметрические значения временного ряда добавлены или обновлены.
Временные ряды удалены	Телеметрические значения или ключи временного ряда удалены.

2.8.2.6 ТРЕВОГИ

Связи тревог (**Ошибка! Неверная ссылка закладки.**) обеспечивают обработку всех этапов жизненного цикла тревожного события. С их помощью можно отдельно реагировать на создание тревоги, изменение её параметров или серьёзности, подтверждение пользователем, назначение ответственного и последующее снятие тревоги. Это позволяет настроить уведомление сотрудников, эскалацию критических событий, регистрацию действий оператора и автоматическое выполнение команд при возникновении или устранении аварийной ситуации.

Таблица 14 Перечень и назначение узлов потоков

Наименование	Описание
Тревога	Общее событие тревоги. Может использоваться для маршрутизации всех операций с тревогой через одну ветку.
Тревога создана	Создана новая тревога для устройства, актива или другой сущности.
Тревога обновлена	Изменены параметры или сведения существующей тревоги.
Серьёзность тревоги обновлена	Изменён уровень серьёзности тревоги, например с предупреждения на критический.
Тревога подтверждена	Пользователь подтвердил, что тревога принята в работу или просмотрена.
Тревога снята	Тревожное состояние устранено, и активная тревога закрыта.
Тревога назначена	Тревога назначена ответственному пользователю.
Тревога снята с назначения	Ответственный пользователь снят с тревоги или назначение отменено.

2.8.2.7 КОММЕНТАРИИ К ТРЕВОГАМ

Связи комментариев к тревогам (Таблица 15) используются для отслеживания действий пользователей, связанных с добавлением и изменением комментариев к тревогам. Комментарии могут содержать сведения о причине события, выполненных действиях, результатах проверки или дальнейших рекомендациях. Обработка таких событий позволяет вести историю работы с тревогой, уведомлять ответственных сотрудников и сохранять информацию для последующего анализа.

Таблица 15 Перечень и описание связей комментариев к тревогам

Наименование	Описание
Комментарий создан	К тревоге добавлен новый комментарий.

Наименование	Описание
Комментарий обновлён	Существующий комментарий к тревоге изменён.

2.8.2.8 СОСТОЯНИЯ RPC-КОМАНД

Связи состояния RPC-команд (Таблица 16) позволяют контролировать прохождение RPC-команды от момента её создания до получения результата выполнения. Цепочка правил может определить, была ли команда поставлена в очередь, отправлена, доставлена устройству и успешно выполнена. Отдельные связи для тайм-аута, просрочки и ошибки дают возможность настроить повторную отправку команды, создать тревогу или зарегистрировать причину невыполнения операции.

Таблица 16 Перечень и описание связей состояния RPC-команд

Наименование	Описание
RPC в очереди	RPC-команда сформирована и поставлена в очередь на отправку устройству.
RPC отправлен	RPC-команда отправлена транспортному сервису или устройству.
RPC доставлен	Подтверждена доставка RPC-команды устройству.
RPC успешен	RPC-команда успешно выполнена, получен положительный результат.
Тайм-аут RPC	Ответ на RPC-команду не получен в пределах установленного времени ожидания.
RPC просрочен	Истёк срок действия RPC-команды до её успешного выполнения.
RPC не удался	RPC-команда завершилась ошибкой или устройство сообщило о невозможности выполнения.
RPC удалён	RPC-команда или запись о ней удалена из платформы.

2.8.2.9 СВЯЗИ МЕЖДУ СУЩНОСТЯМИ

Связи между сущностями (Таблица 17) используются при создании, изменении и удалении логических отношений между сущностями платформы. Например, устройство может быть связано с активом, помещением, производственным участком или другим объектом. Обработка изменений таких отношений необходима для поддержания актуальной структуры системы, поиска связанных объектов, обновления прав доступа и запуска действий, зависящих от положения сущности в общей иерархии.

Таблица 17 Перечень и описание связей между сущностями

Наименование	Описание
Связь добавлена или обновлена	Между двумя сущностями создана новая связь либо изменены параметры существующей связи.
Связь удалена	Удалена отдельная связь между сущностями.
Все связи удалены	Удалены все связи, относящиеся к указанной сущности.

2.8.2.10 ВНЕШНИЕ И СИСТЕМНЫЕ СОБЫТИЯ

Связи внешних и системных событий (Таблица 18) предназначены для обработки событий, которые поступают не непосредственно от устройства, а от внешних сервисов или внутренних механизмов платформы. К ним относятся запросы через REST API и события, связанные с формированием отчётов. Такие связи позволяют запускать цепочки правил по команде сторонней информационной системы, обрабатывать интеграционные запросы и выполнять служебные операции платформы.

Таблица 18 Перечень и описание связей внешних и системных событий

Наименование	Описание
REST API-запрос	В цепочку правил поступил запрос через программный интерфейс REST API.
Генерация отчёта	Запущено формирование отчёта или обработка события, связанного с генерацией отчёта.

2.8.3 Пример настройки цепочек правил

В разделе ниже описан пример проверки входящей телеметрии через цепочку правил. В данном примере реализована фильтрация сохраняемых значений температура: значение сохраняется только в том случае, если ее оно находится в диапазоне от –40 до +80 °С. Все остальные значения не сохраняются, но, при этом, записываются в журнал.

Для создания подобной цепочки правил (Рисунок 29) необходимо открыть раздел **«Цепочки правил»** (1) и перейти в корневую цепочку правил. Затем из **списка узлов** (2) необходимо перетащить узел **«Скрипт»** (3) на рабочую область.

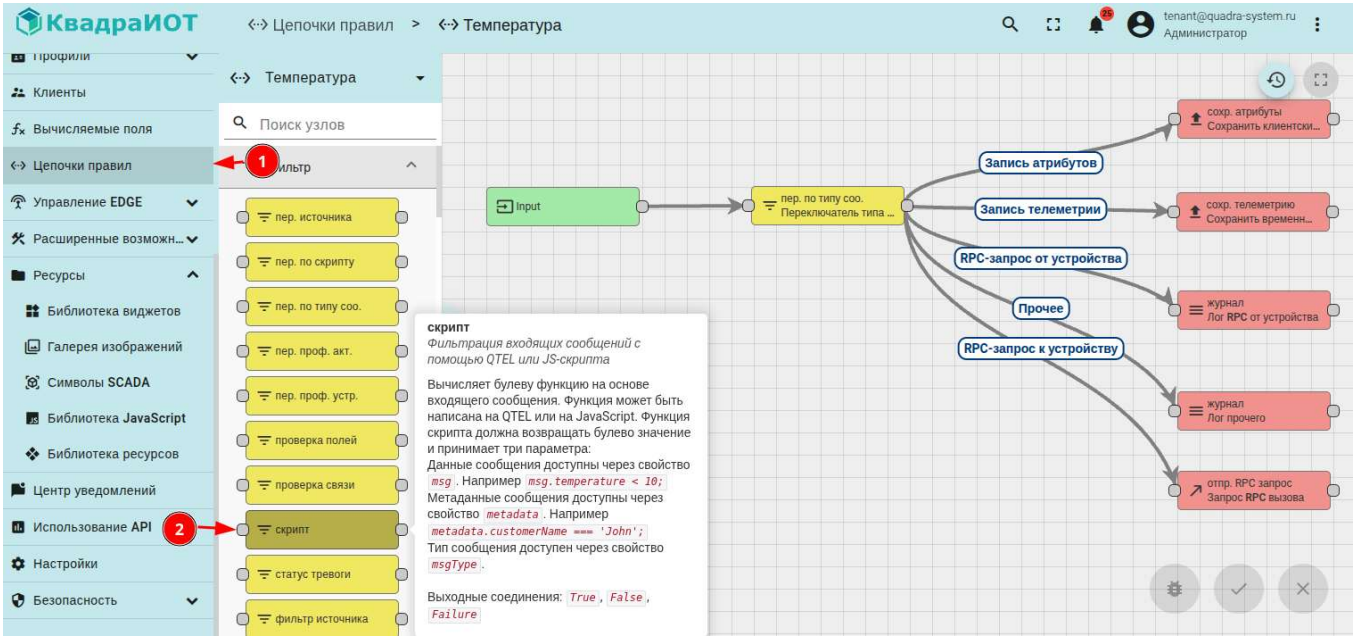


Рисунок 29 – Пример создания цепочки правил

В окне настройки узла (Рисунок 30) необходимо указать **имя** (1), **выбрать режим** (2).

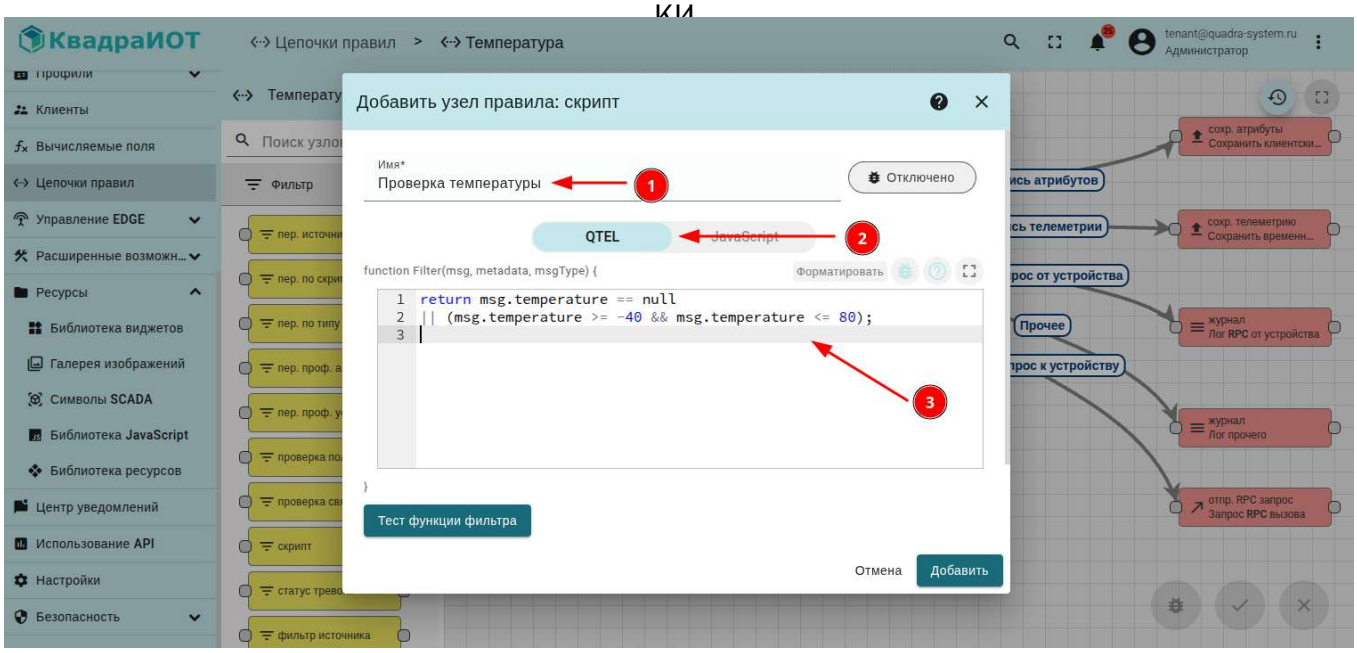


Рисунок 30 – Настройка узла типа «Скрипт»

В **окно скрипта (3)** необходимо написать непосредственно сам скрипт обработки сигнала:

```
return msg.temperature == null
|| (msg.temperature >= -40 && msg.temperature <= 80);
```

После написания скрипта необходимо нажать кнопку «Добавить» и перейти к настройки связей в цепочки правил (Рисунок 31). Для этого необходимо:

1. Удалить связь «**Post telemetry**» от узла «**пер. по типу соо.**» к узлу «**сохр. Телеметрию**» (1).

2. Создать новую связь «**Post telemetry**» от узла «**пер. по типу соо.**» к узлу «**проверка температуры**» (2).

3. Создать две связи от узла «**Проверка температуры**» (3):

- связь «**True**» к узлу «**сохр. Телеметрию**» – для сохранения корректных данных;
- связь «**False**» к узлу «**Журнал**» – для записи некорректных данных в журнал.

После завершения настройки необходимо нажать кнопку сохранения.

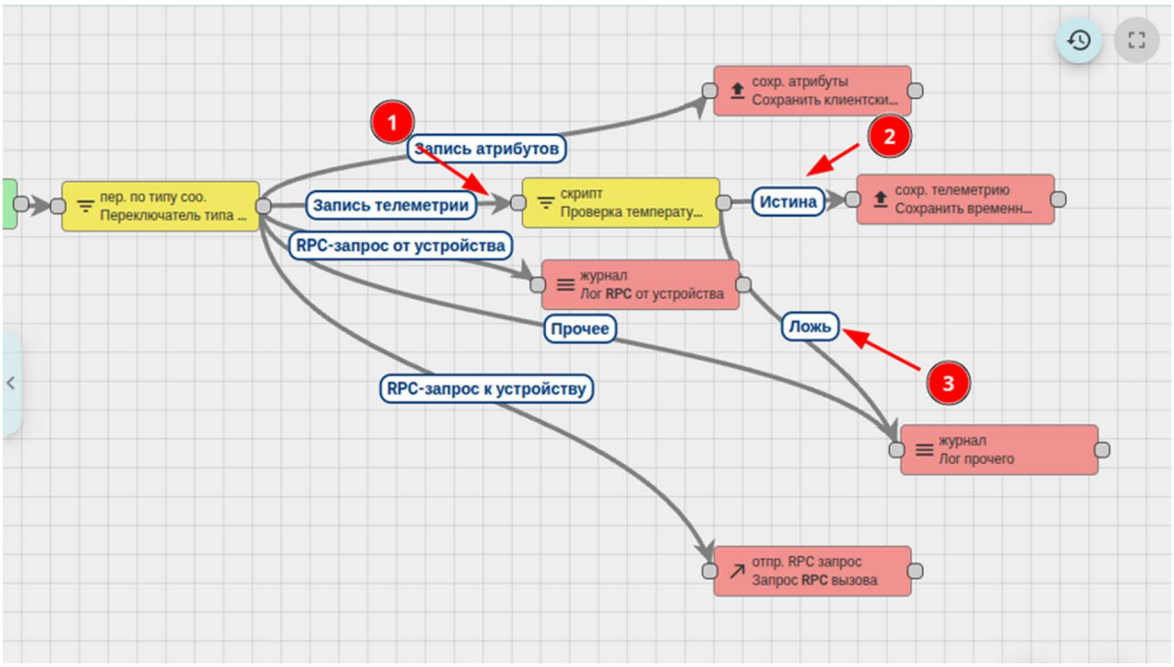


Рисунок 31 – Настройка связей в цепочке правил

В результате созданной цепочки правил требуемые значения температуры будут сохраняться в системе, а остальные отсекаться, но при этом записываться в журнал.

2.8.3.1 ПРОВЕРКА РАБОТЫ ПРАВИЛА

Для проверки настроенной цепочки правил необходимо **добавить новое устройство** (Рисунок 32), затем открыть его карточку и скопировать **токен доступа** (Рисунок 33).

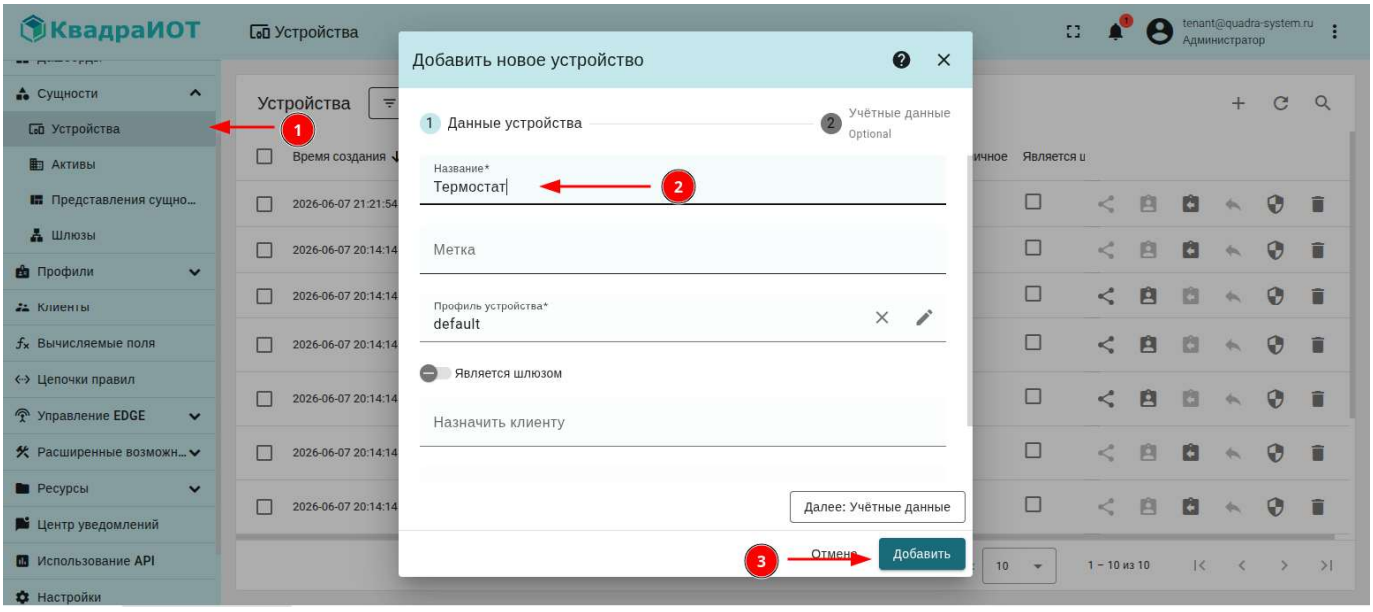


Рисунок 32 – Добавление нового устройства

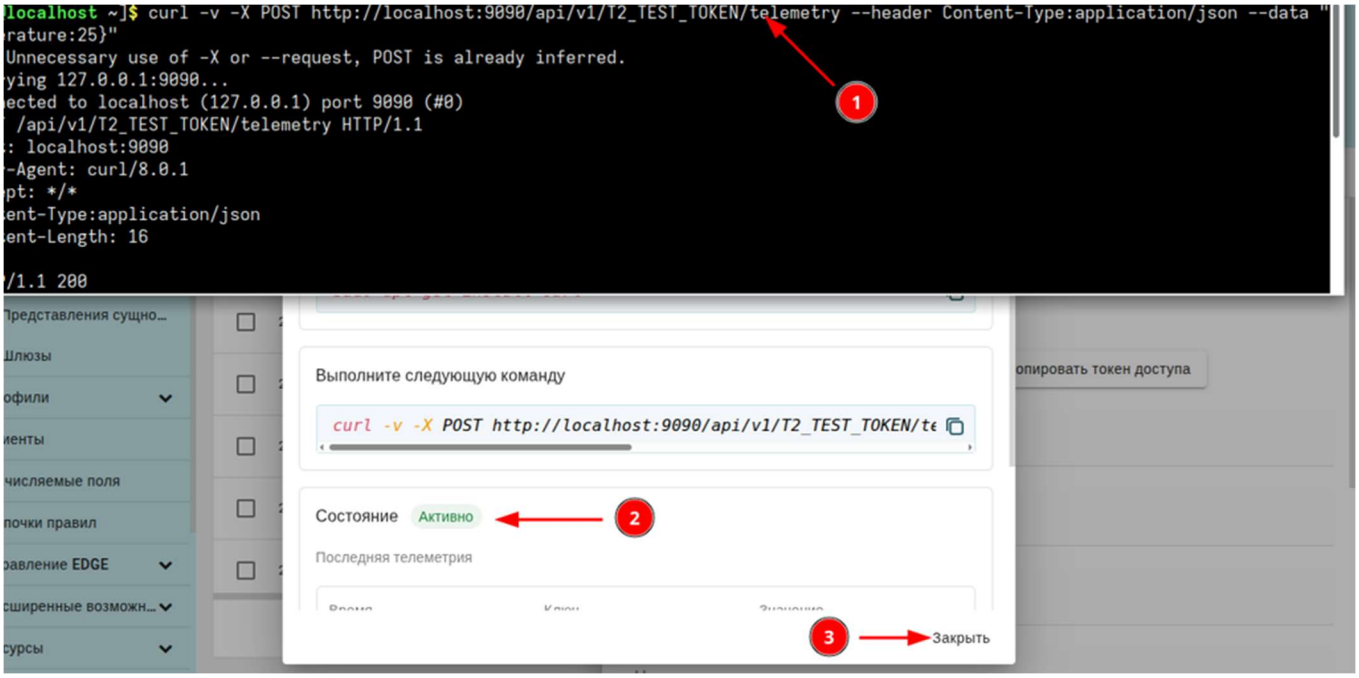


Рисунок 33 – Копирование токена доступа нового устройства

Далее через **REST API** необходимо отправить тестовые значения температуры. Если отправить значение, выходящее за допустимый диапазон, например **99**, телеметрия не будет сохранена в разделе последних данных устройства. Если отправить корректное значение, например **30**, оно будет успешно сохранено.

Для отправки данных со значением «99» используется команда:

```
curl -v -X POST -d '{"temperature":99}' http://localhost:9090/api/v1/$ACCESS_TOKEN/telemetry --header "Content-Type:application/json"
```

Для отправки данных со значением «30» используется команда:

```
curl -v -X POST -d '{"temperature":30}' http://localhost:9090/api/v1/$ACCESS_TOKEN/telemetry --header "Content-Type:application/json"
```

Далее переменную «**$ACCESS_TOKEN**» в настройках устройства необходимо заменить на фактический токен устройства (Рисунок 34).

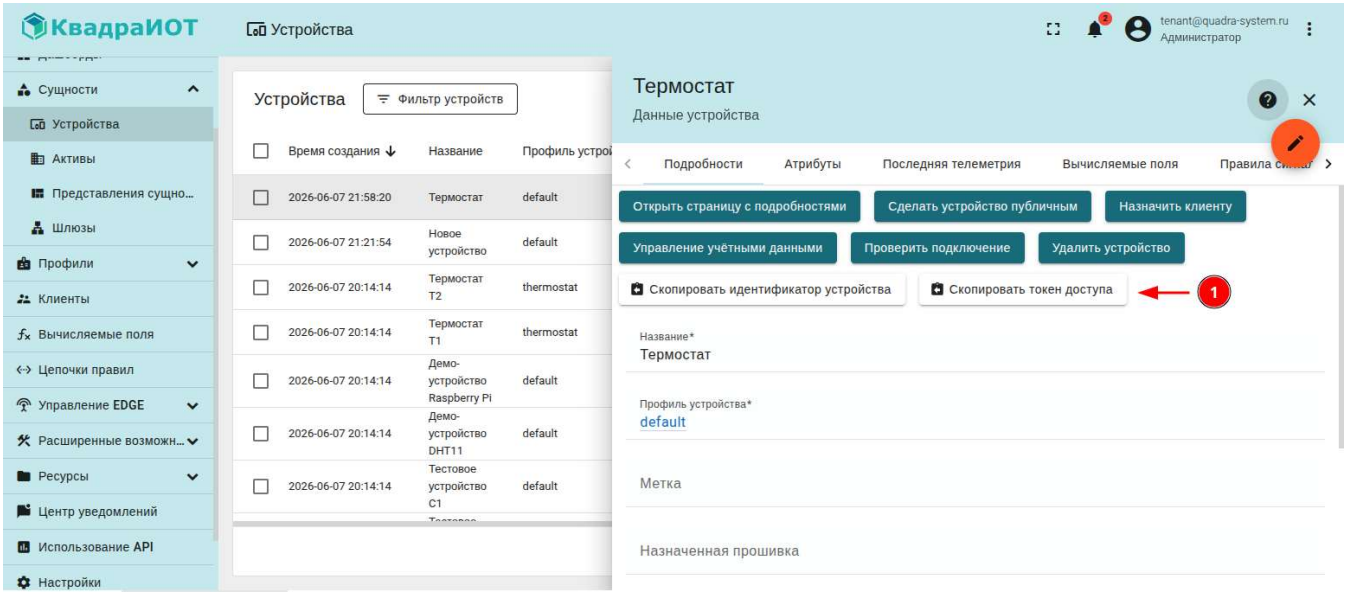


Рисунок 34 – Замена переменной «**$ACCESS_TOKEN**»

В окне «Последняя телеметрия» (Рисунок 35) можно убедиться, что корректные значения температуры сохранились, а недопустимые отфильтровались согласно ранее настроенной цепочки правил.

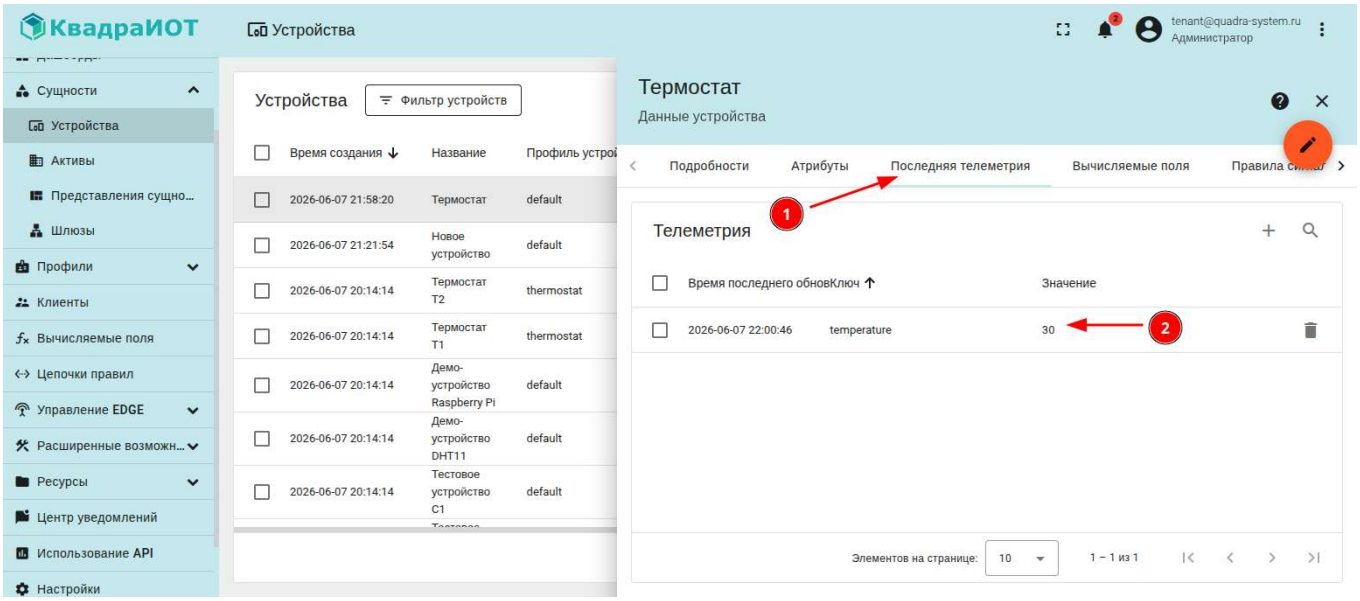


Рисунок 35 – Сохраненные данные устройства

3 Контакты

Для контактов с командой разработчиков просьба обращаться по электронной почте: info@quadra-system.ru.