



МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение
высшего образования

«МИРЭА – Российский технологический университет»

РТУ МИРЭА

Институт Информационных технологий

Кафедра Математического обеспечения и стандартизации информационных
технологий

Отчет по практическим работам №9-12

по дисциплине «Технологические основы Интернета вещей»

Выполнили:

Студенты группы ИКБО-20-23

Кузнецов Л.А.
Комисарик М.А.

Проверил:

Жматов Дмитрий Владимирович

2025 г.

ОГЛАВЛЕНИЕ

1. Практическая работа №9	3
2. Практическая работа №10	10
3. Практическая работа №11	13
4. Практическая работа №12	17
ВЫВОД.....	20

ПРАКТИЧЕСКАЯ ЧАСТЬ

1. Практическая работа №9

Часть 1. Регистрация на платформе Rightech IoT Cloud

Для выполнения работ была осуществлена регистрация на сайте dev.rightech.io.

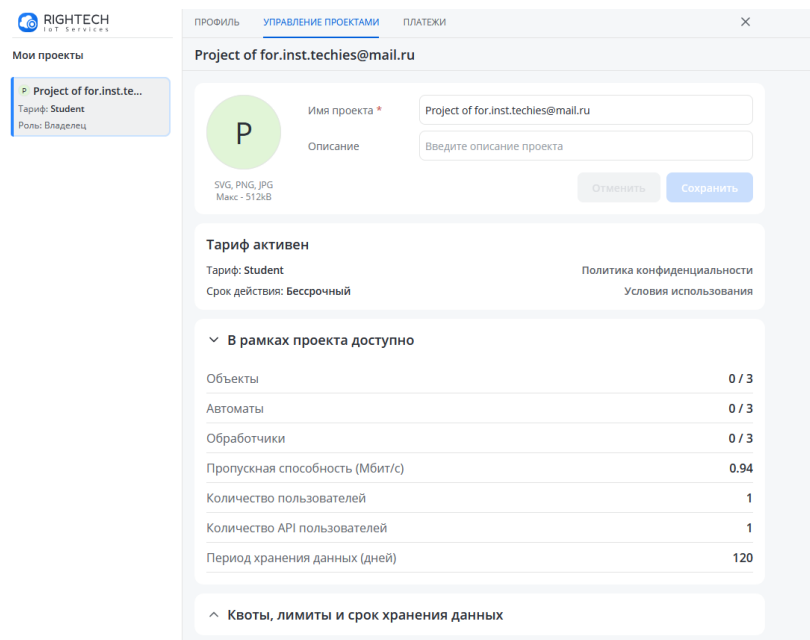


Рисунок 1.1.1 – Вид профиля зарегистрированного пользователя на dev.rightech.io

Часть 2. Создание виртуальных устройств в облаке

В облаке были созданы виртуальные устройства согласно варианту.

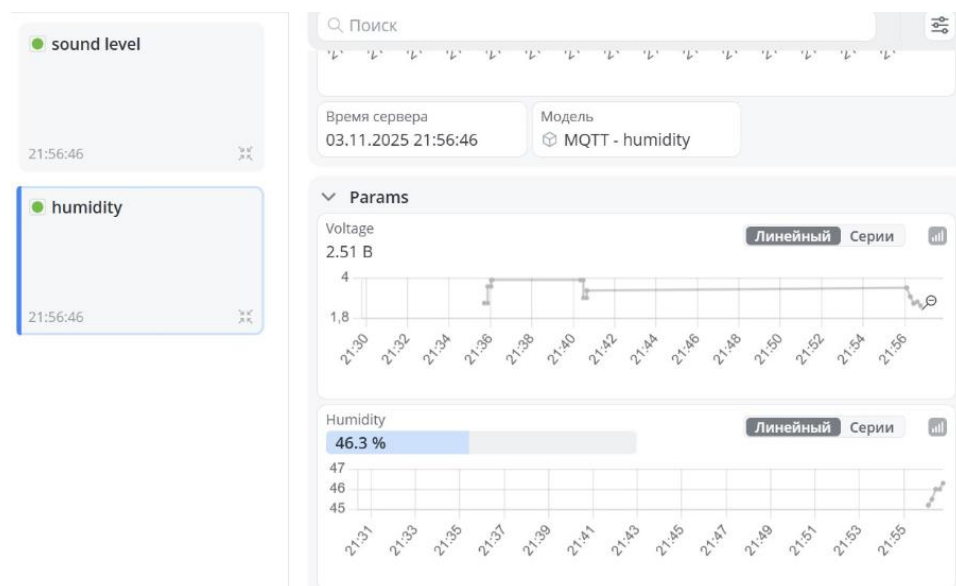


Рисунок 1.2.1 – Показатель напряжения на датчик влажности

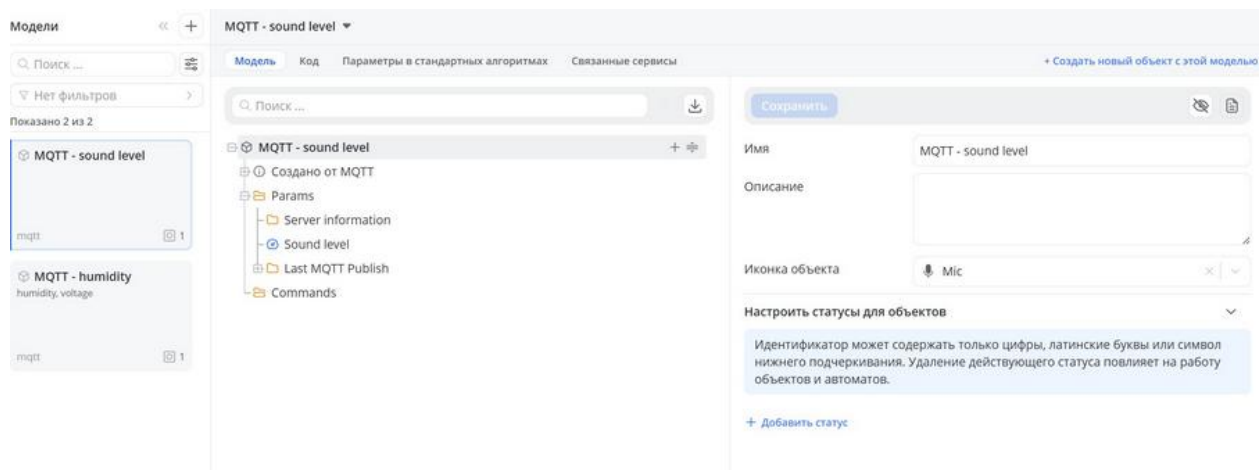


Рисунок 1.2.2 – Модель датчика громкости звука часть 1

Сохранить
🔍

Тип	🔍 Аргумент
Идентификатор	sound_level
Имя	Sound level
Описание	
Тип данных	number : Число
Источник	/devices/sensors/s2/sound level
	Источник данных зависит от протокола
Линейный	<input checked="" type="checkbox"/>
При разборе	⬆
Отображение	⬇
	Настройка отображения параметра в меню Объект
Единица измерения	децибелы (дБ)
Количество знаков после запятой	2

Рисунок 1.2.3 – Модель датчика громкости звука часть 2

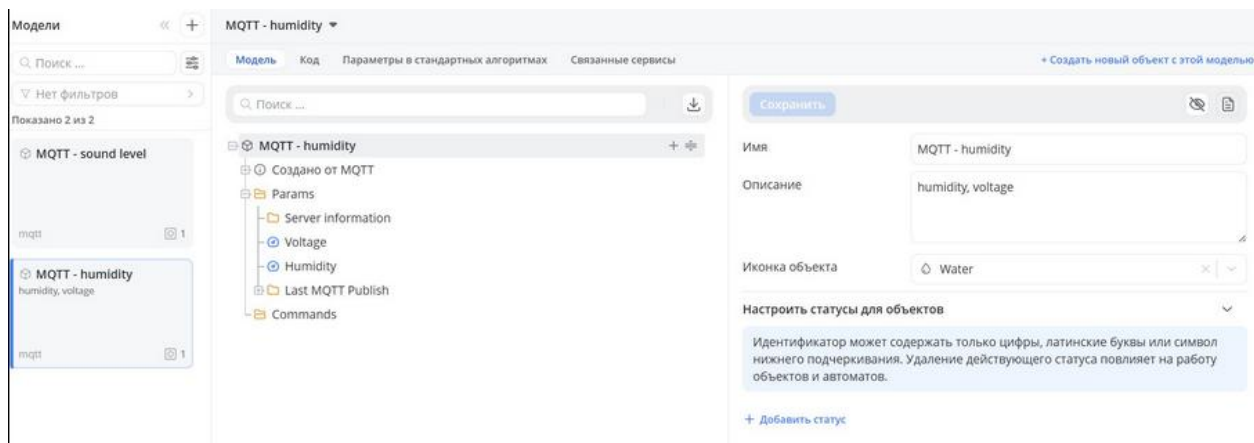


Рисунок 1.2.4 – Модель датчика влажности часть 1

Тип	Аргумент
Идентификатор	humidity
Имя	Humidity
Описание	
Тип данных	number : Число
Источник	/devices/sensors/s1/humidity
	Источник данных зависит от протокола
Линейный	<input checked="" type="checkbox"/>
При разборе	^
Отображение	^
	Настройка отображения параметра в меню Объект
Единица измерения	проценты (%)
Количество знаков после запятой	1

Рисунок 1.2.5 – Модель датчика влажности часть 2

Также с моделями были созданы объекты датчиков.

The screenshot shows the configuration page for a sensor object named 'sound level'. On the left, a sidebar lists two objects: 'sound level' (selected) and 'humidity'. The main panel has tabs for 'Журнал', 'Редактирование' (active), 'Команды', and 'Сервисы'. A 'Сохранить' button is at the top. The configuration fields are as follows:

- Модель ***: MQTT - sound level
- Идентификатор ***: mqtt-mihakom050-s2 (with a note: ClientID, используемый при подключении устройства)
- Имя ***: sound level (with a note: произвольное, отображается в интерфейсе пользователя)
- Описание**: (empty text area with a note: описание объекта)
- Использовать как бота**: toggle switch (turned off)
- Статус**: (empty text field)

At the top of the configuration area, there is a 'Протокол' section with the following details: MQTT (описание подключения), MQTT / TCP: dev.rightech.io:1883, and MQTTS / TLS: dev.rightech.io:8883.

Рисунок 1.2.6 – Объект датчика громкости (шума)

The screenshot shows the configuration page for a sensor object named 'humidity'. On the left, a sidebar lists two objects: 'sound level' and 'humidity' (selected). The main panel has tabs for 'Журнал', 'Редактирование' (active), 'Команды', and 'Сервисы'. A 'Сохранить' button is at the top. The configuration fields are as follows:

- Модель ***: MQTT - humidity
- Идентификатор ***: mqtt-mihakom050-s1 (with a note: ClientID, используемый при подключении устройства)
- Имя ***: humidity (with a note: произвольное, отображается в интерфейсе пользователя)
- Описание**: humidity, voltage (with a note: описание объекта)

At the top of the configuration area, there is a 'Протокол' section with the following details: MQTT (описание подключения), MQTT / TCP: dev.rightech.io:1883, and MQTTS / TLS: dev.rightech.io:8883.

Рисунок 1.2.7 – Объект датчика влажности

Часть 3. Отправка данных в облако

Отправка в облако проводилась при помощи эмулятора написанного на языке Python.

```
emulator.py x
1 import dataclasses
2 import random
3 import time
4
5 import paho.mqtt.client as mqtt
6 from datetime import datetime
7
8
9 @dataclasses.dataclass
10 class SensorData:
11     timestamp: str
12     humidity: float
13     sound_level: float
14     voltage: float
15
16     def as_dict(self):
17         return dataclasses.asdict(self)
18
19     def __str__(self):
20         return (f"[{self.timestamp[11:19]}] "
21             f"Вн-ть: {self.humidity:5.1f}% | "
22             f"Громкость: {self.sound_level:6.2f}dB | "
23             f"Напряжение: {self.voltage:4.2f}V")
24
25
26 class MultiClientSensorEmulator:
27     def __init__(self, broker_host, broker_port, client_id_s1, client_id_s2, username=None, password=None):
28         self.broker_host = broker_host
29         self.broker_port = broker_port
30
31         self.client_s1 = mqtt.Client(client_id=client_id_s1)
32         self.client_s1.on_connect = self.on_connect
33         if username:
34             self.client_s1.username_pw_set(username, password)
35
36         self.client_s2 = mqtt.Client(client_id=client_id_s2)
37         self.client_s2.on_connect = self.on_connect
38         if username:
39             self.client_s2.username_pw_set(username, password)
40
```

Рисунок 1.3.1 – Код эмулятора с подключением к брокеру по адресу dev.rigtech.io часть 1

```
emulator.py x
26 class MultiClientSensorEmulator:
27     def __init__(self, broker_host, broker_port, client_id_s1, client_id_s2, username=None, password=None):
28         self.sensor_readings = {
29             'humidity': {'min': 30.0, 'max': 90.0, 'current': 45.0, 'trend': 0.2},
30             'sound_level': {'min': 30.0, 'max': 95.0, 'current': 40.0, 'trend': 0.5}, # Работает с float
31             'voltage': {'min': 0.0, 'max': 5.0, 'current': 3.3, 'trend': -0.05}
32         }
33
34         self.mqtt_topics = {
35             'humidity': '/devices/sensors/s1/humidity',
36             'sound_level': '/devices/sensors/s2/sound_level',
37             'voltage': '/devices/sensors/s1/voltage'
38         }
39
40     def on_connect(self, client, userdata, flags, rc):
41         client_id = client.client_id.decode('utf-8')
42         if rc == 0:
43             print(f"Клиент '{client_id}' успешно подключился к {self.broker_host}")
44         else:
45             print(f"Ошибка подключения клиента '{client_id}'. Код: {rc}")
46             if rc == 5:
47                 print("-> Ошибка аутентификации: проверьте логин и пароль.")
48
49     def update_sensor_readings(self):
50         for sensor, params in self.sensor_readings.items():
51             change = params['trend'] + random.uniform(-0.5, 0.5)
52             params['current'] += change
53
54             if params['current'] < params['min']:
55                 params['current'] = params['min']
56                 params['trend'] = abs(params['trend'])
57             elif params['current'] > params['max']:
58                 params['current'] = params['max']
59                 params['trend'] = -abs(params['trend'])
60
61             if random.random() < 0.1:
62                 params['trend'] = random.uniform(-1, 1)
63
64     def generate_sensor_data(self) -> SensorData:
65         self.update_sensor_readings()
```

Рисунок 1.3.2 – Код эмулятора с подключением к брокеру по адресу dev.rigtech.io часть 2

```

emulator.py
class MultiClientSensorEmulator:
    def generate_sensor_data(self) -> SensorData:
        return SensorData(
            timestamp=datetime.now().isoformat(),
            humidity=round(self.sensor_readings['humidity']['current'], 1),
            sound_level=round(self.sensor_readings['sound_level']['current'], 2),
            voltage=round(self.sensor_readings['voltage']['current'], 2)
        )

    def publish(self, sensor_data: SensorData):
        self.client_s1.publish(self.mqtt_topics['humidity'], sensor_data.humidity)
        self.client_s1.publish(self.mqtt_topics['voltage'], sensor_data.voltage)
        self.client_s2.publish(self.mqtt_topics['sound_level'], sensor_data.sound_level)
        print(sensor_data)

    def start_emulation(self, interval):
        print("Запуск эмуляции датчиков...")
        try:
            self.client_s1.connect(self.broker_host, self.broker_port, keepalive=60)
            self.client_s2.connect(self.broker_host, self.broker_port, keepalive=60)

            self.client_s1.loop_start()
            self.client_s2.loop_start()

            while True:
                data = self.generate_sensor_data()
                self.publish(data)
                time.sleep(interval)

```

Рисунок 1.3.3 – Код эмулятора с подключением к брокеру по адресу dev.rigtech.io часть 3

```

109     except KeyboardInterrupt:
110         print("\n\nЭмуляция остановлена пользователем.")
111     except Exception as e:
112         print(f"Произошла ошибка: {e}")
113     finally:
114         self.client_s1.loop_stop()
115         self.client_s1.disconnect()
116         self.client_s2.loop_stop()
117         self.client_s2.disconnect()
118         print("Соединения с MQTT брокером закрыты.")
119
120
121 if __name__ == "__main__":
122     emulator = MultiClientSensorEmulator(
123         broker_host="dev.rigtech.io",
124         broker_port=1883,
125         client_id_s1="mqtt-mihakom050-s1",
126         client_id_s2="mqtt-mihakom050-s2"
127     )
128     emulator.start_emulation(interval=10)
129

```

Рисунок 1.3.4 – Код эмулятора с подключением к брокеру по адресу dev.rigtech.io часть 4

```

emulator.py
import dataclasses
import random
import time

import paho.mqtt.client as mqtt
from datetime import datetime

@dataclasses.dataclass
class SensorData:
    timestamp: str
    humidity: float
    sound_level: float
    voltage: float

    def as_dict(self):
        return dataclasses.asdict(self)

    def __str__(self):
        return f"[{self.timestamp[11:19]}] "
        f"[{self.humidity:9.10}%] "
        f"[Громкость: {self.sound_level:0.2f}dB] "
        f"[Напряжение: {self.voltage:0.2f}V]"

class MultiClientSensorEmulator:
    def __init__(self):
        self.broker_host = "dev.rigtech.io"
        self.broker_port = 1883
        self.client_id_s1 = "mqtt-mihakom050-s1"
        self.client_id_s2 = "mqtt-mihakom050-s2"
        self.mqtt_topics = {
            "humidity": "humidity",
            "voltage": "voltage",
            "sound_level": "sound_level"
        }
        self.client_s1 = None
        self.client_s2 = None
        self.interval = 10

    def generate_sensor_data(self) -> SensorData:
        return SensorData(
            timestamp=datetime.now().isoformat(),
            humidity=round(random.random() * 100, 1),
            sound_level=round(random.random() * 100, 2),
            voltage=round(random.random() * 5, 2)
        )

    def publish(self, sensor_data: SensorData):
        self.client_s1.publish(self.mqtt_topics['humidity'], sensor_data.humidity)
        self.client_s1.publish(self.mqtt_topics['voltage'], sensor_data.voltage)
        self.client_s2.publish(self.mqtt_topics['sound_level'], sensor_data.sound_level)
        print(sensor_data)

    def start_emulation(self, interval):
        print("Запуск эмуляции датчиков...")
        try:
            self.client_s1.connect(self.broker_host, self.broker_port, keepalive=60)
            self.client_s2.connect(self.broker_host, self.broker_port, keepalive=60)

            self.client_s1.loop_start()
            self.client_s2.loop_start()

            while True:
                data = self.generate_sensor_data()
                self.publish(data)
                time.sleep(interval)

```

```

Run
C:\Users\Nesip\AppData\Local\Programs\Python\Python312\python.exe C:\Users\Nesip\Downloads\emulator.py
DeprecationWarning: Callback API version 1 is deprecated, update to latest version
self.client_s1 = mqtt.Client(client_id=self.client_id_s1)
DeprecationWarning: Callback API version 1 is deprecated, update to latest version
self.client_s2 = mqtt.Client(client_id=self.client_id_s2)
Запуск эмуляции датчиков...
[12:41:02] 88-ты: 45.5% | Громкость: 40.74dB | Напряжение: 3.28V
Клиент 'mqtt-mihakom050-s2' успешно подключен к dev.rigtech.io
Клиент 'mqtt-mihakom050-s1' успешно подключен к dev.rigtech.io
[12:41:12] 88-ты: 46.1% | Громкость: 40.88dB | Напряжение: 3.30V
[12:41:22] 88-ты: 46.2% | Громкость: 41.20dB | Напряжение: 3.35V

```

Рисунок 1.3.5 – Код эмулятора с подключением к брокеру по адресу dev.rigtech.io часть 5

Корректность работы эмулятора была проверена в IDE, а также на сайте dev.rightech.io.

```
C:\Users\Noip\Downloads\emulator.py:31: DeprecationWarning: Callback API version 1 is deprecated, update to latest version
  self.client_s1 = mqtt.Client(client_id=client_id_s1)
C:\Users\Noip\Downloads\emulator.py:36: DeprecationWarning: Callback API version 1 is deprecated, update to latest version
  self.client_s2 = mqtt.Client(client_id=client_id_s2)
Запуск эмуляции датчиков...
[12:39:24] Вл-ть: 45.2% | Громкость: 40.66dB | Напряжение: 2.98V
Клиент 'mqtt-mihakom050-s1' успешно подключился к dev.rightech.io
Клиент 'mqtt-mihakom050-s2' успешно подключился к dev.rightech.io
[12:39:34] Вл-ть: 45.6% | Громкость: 41.26dB | Напряжение: 3.11V
[12:39:44] Вл-ть: 45.4% | Громкость: 42.34dB | Напряжение: 2.91V
```

Рисунок 1.3.6 – Итоговый вид результата работы эмулятора

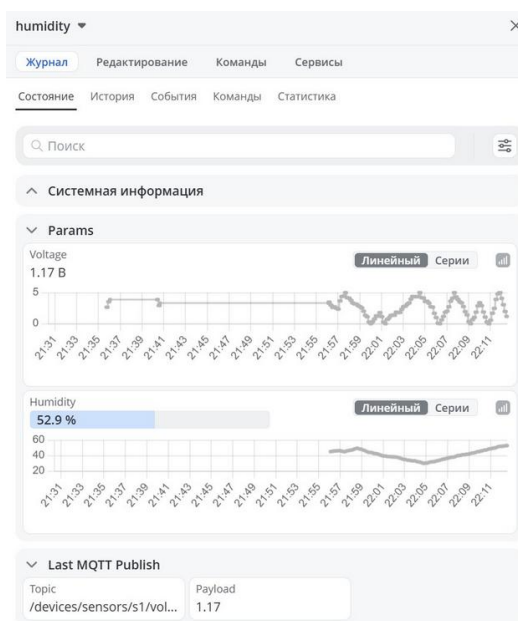


Рисунок 1.3.7 – Получение данных влажности и напряжения

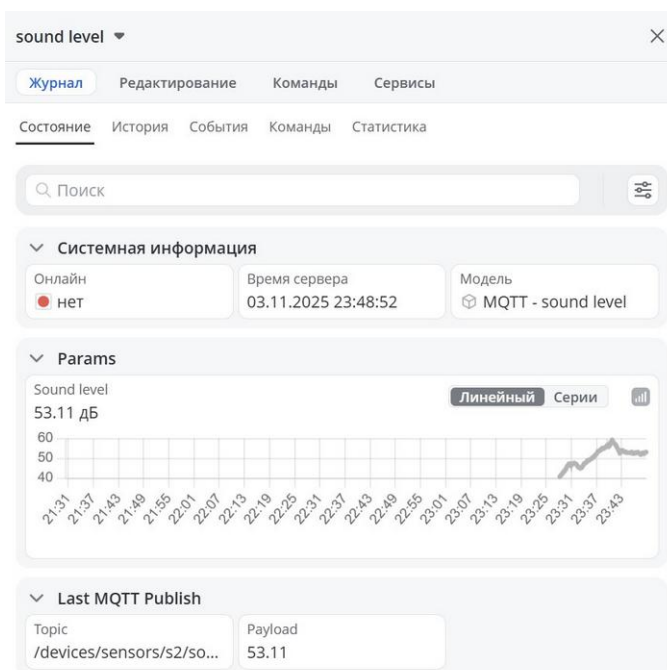


Рисунок 1.3.8 – Получение данных громкости звука

```

Запуск эмуляции датчиков...
[23:30:33] Вл-ть: 44.0% | Громкость: 47.29dB | Напряжение: 5.00V
[23:30:44] Вл-ть: 44.2% | Громкость: 47.31dB | Напряжение: 4.25V
[23:30:54] Вл-ть: 44.6% | Громкость: 46.22dB | Напряжение: 3.88V
[23:31:04] Вл-ть: 45.5% | Громкость: 46.52dB | Напряжение: 3.27V
[23:31:14] Вл-ть: 46.1% | Громкость: 47.27dB | Напряжение: 3.00V
[23:31:24] Вл-ть: 46.9% | Громкость: 47.41dB | Напряжение: 3.15V
[23:31:34] Вл-ть: 47.8% | Громкость: 47.63dB | Напряжение: 3.80V
[23:31:44] Вл-ть: 48.4% | Громкость: 47.59dB | Напряжение: 4.09V
[23:31:54] Вл-ть: 47.9% | Громкость: 47.50dB | Напряжение: 3.87V
[23:32:04] Вл-ть: 48.0% | Громкость: 47.14dB | Напряжение: 4.19V
[23:32:14] Вл-ть: 48.2% | Громкость: 47.03dB | Напряжение: 4.18V

```

Рисунок 1.3.9 – Отчёт об отправке данных эмулятором

На рисунках 1.3.7-1.3.8 представлены полученные облаком данные, а на рисунках 1.3.6 и 1.3.9 – процесс проверки корректности работы кода эмулятора.

Часть 2

2. Практическая работа №10

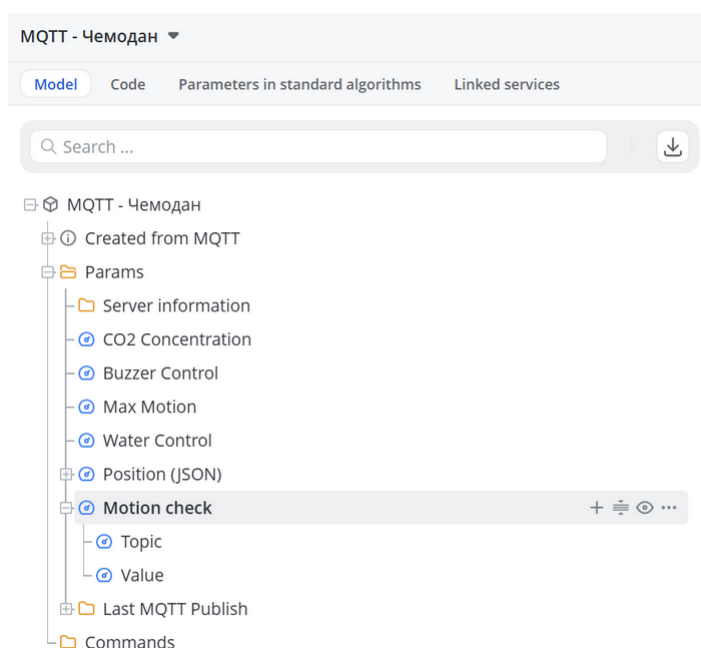


Рисунок 2.1 – Настройка моделей

По варианту необходимо воспроизвести сценарии:

- Включения и выключения звукового сигнала по концентрации CO₂;

– Включения и выключения воды по датчику движения.

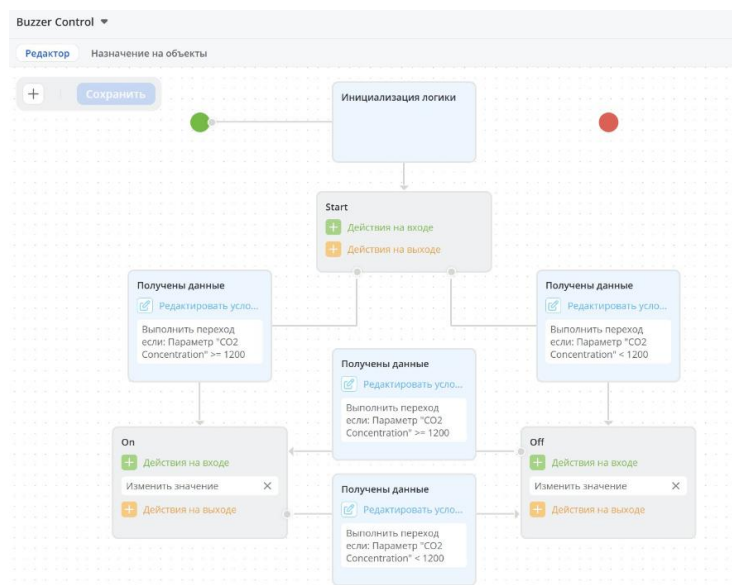


Рисунок 2.2 – Воспроизведение сценария включения и выключения звукового сигнала по концентрации CO2



Рисунок 2.3 – Результаты тестирования воспроизведения сценария включения и выключения звукового сигнала по концентрации CO2

На рисунках 2.2 – 2.3 происходит подключение к получению данных с датчика с CO2, после чего в зависимости от изменения значений датчика происходит включение (при концентрации CO2 большей 1200) и выключение (при концентрации CO2 не меньшей 1200) звукового сигнала.

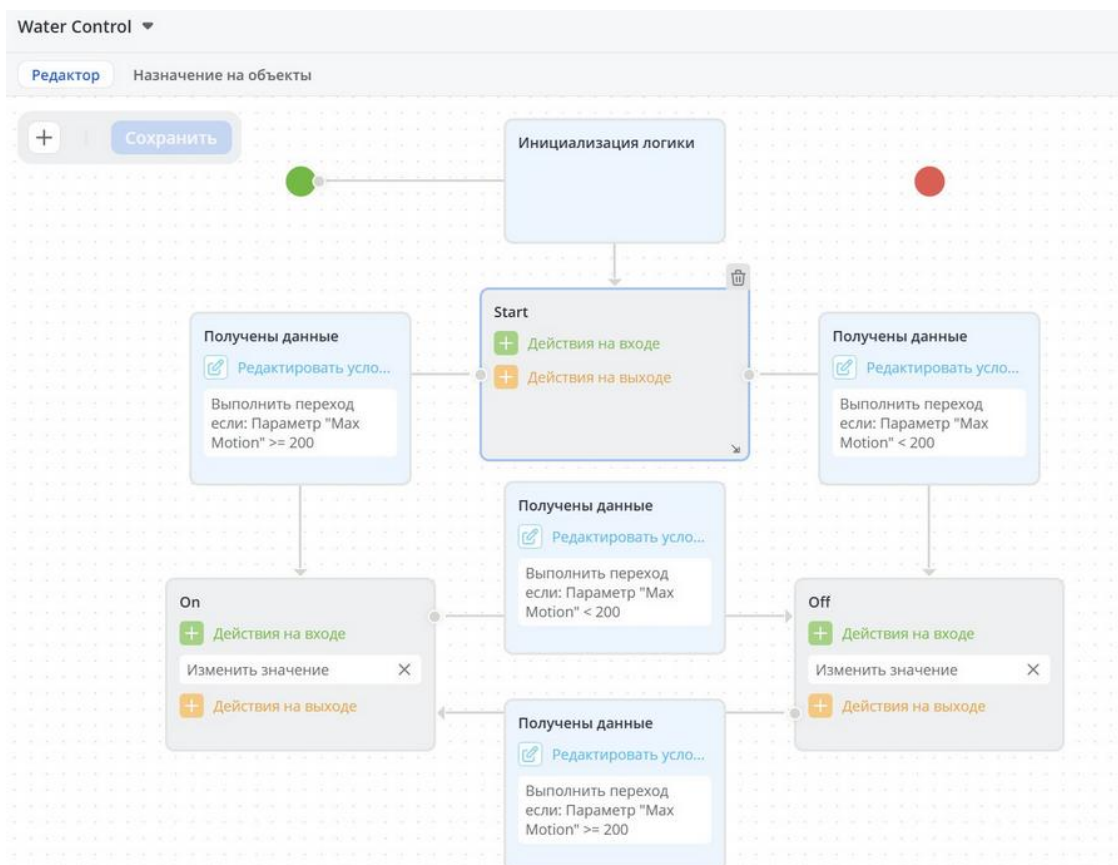


Рисунок 2.4 – Воспроизведение сценария включения и выключения воды по датчику движения

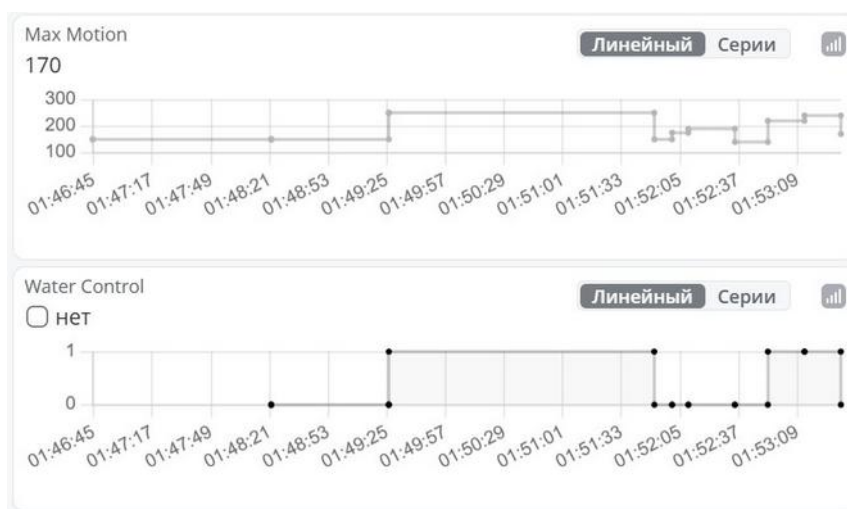


Рисунок 2.5 – Результаты тестирования воспроизведения сценария включения и выключения воды по датчику движения

На рисунках 2.4 – 2.5 происходит подключение к получению данных с датчика движения, после чего в зависимости от изменения значений датчика происходит включение (при зафиксированном перемещении не меньше 200 интенсивности движения) и выключение (при зафиксированном перемещении меньше 200 интенсивности движения) воды.

```

20bridges.conf x
1 #rightech connect config
2 connection rightech
3 address dev.rightech.io:1883
4 clientid mqtt-mihakom050-case
5 try_private false
6 start_type automatic
7 topic /devices/wb-msw-v3_64/controls/CO2 both
8 topic /devices/wb-msw-v3_64/controls/Buzzer/on both
9 topic /devices/wb-msw-v3_64/controls/Max Motion both
10 topic /devices/wb-mwac-v2_25/controls/Output K1 both

```

Рисунок 2.6 – Файл конфигурации для подключения к топикам

3. Практическая работа №11

Согласно варианту были составлены скрипты с различными тревогами.

В первом случае при превышении концентрации CO2 определённого значения отправляется уведомление с текущим уровнем CO2.

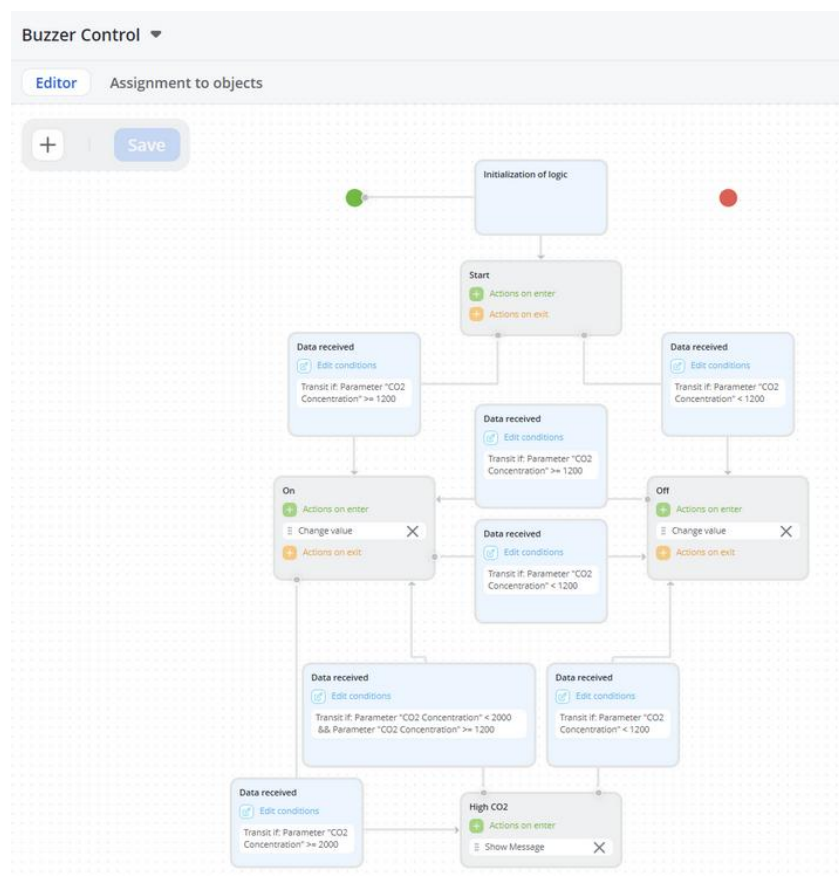


Рисунок 3.1 – Правило автомата первого задания

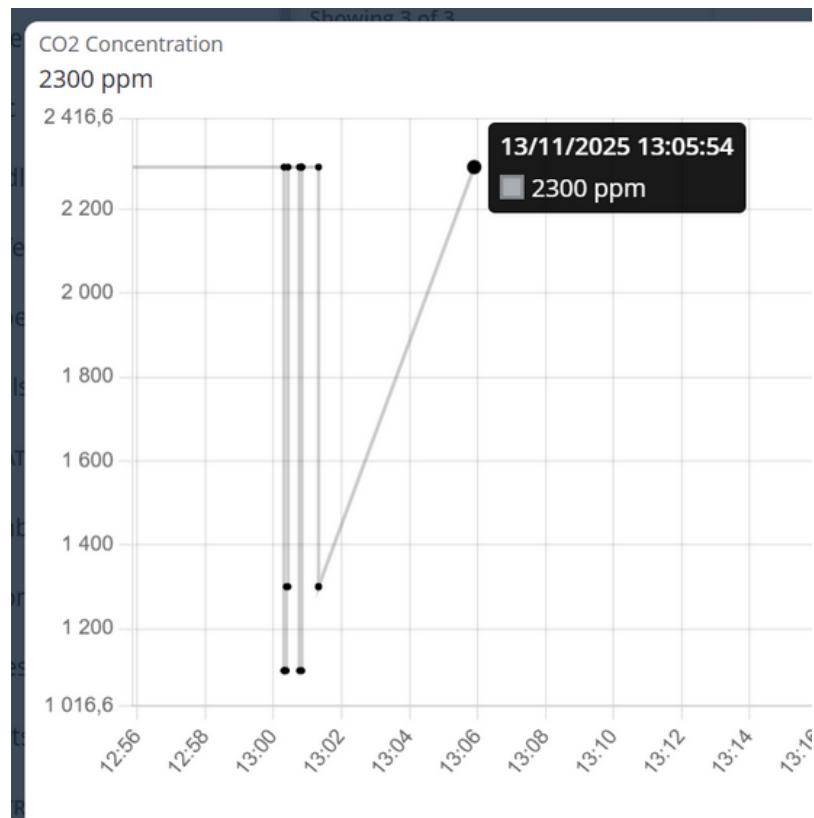


Рисунок 3.2 – Зафиксированные значения датчика концентрации CO2

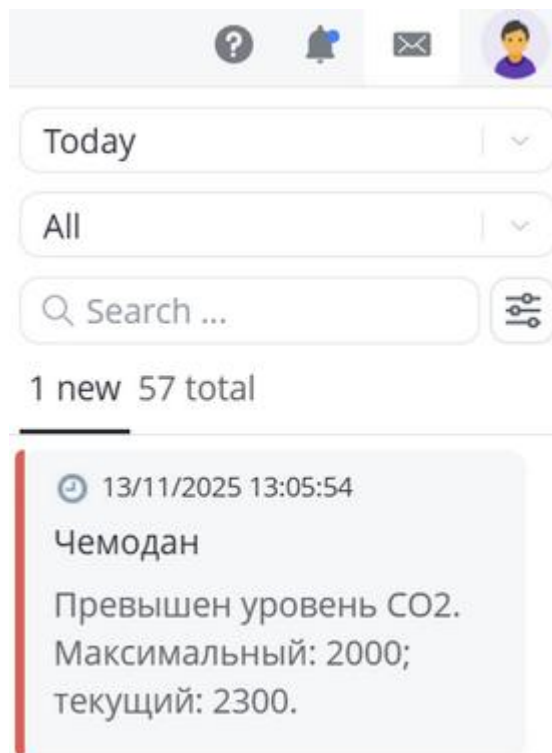


Рисунок 3.3 – Результат тестирования автомата первого задания

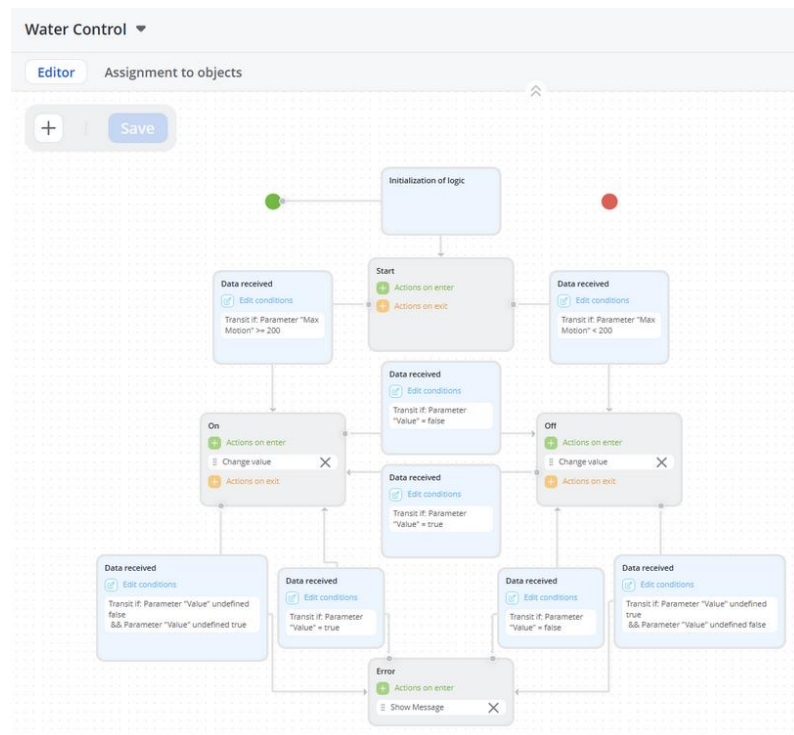


Рисунок 3.4 – Правило автомата для второго задания

Во втором задании при передаче данных, в которых нет motion, пользователю передаётся сообщение об ошибке.

Handlers

Search ...

No filters

Showing 1 of 1

Анализатор движе...

JS

1

Анализатор движения

Edit Handler management

Save Testing

```

1 export function process(topic, payload) {
2   const threshold = 200;
3   if (topic !== "/devices/wb-msw-v3_64/controls/Max Motion")
4   {
5     return { topic: topic }
6   }
7   return { topic: topic, value: payload > threshold}
8 }

```

Рисунок 3.5 – Обработчик для второго правила

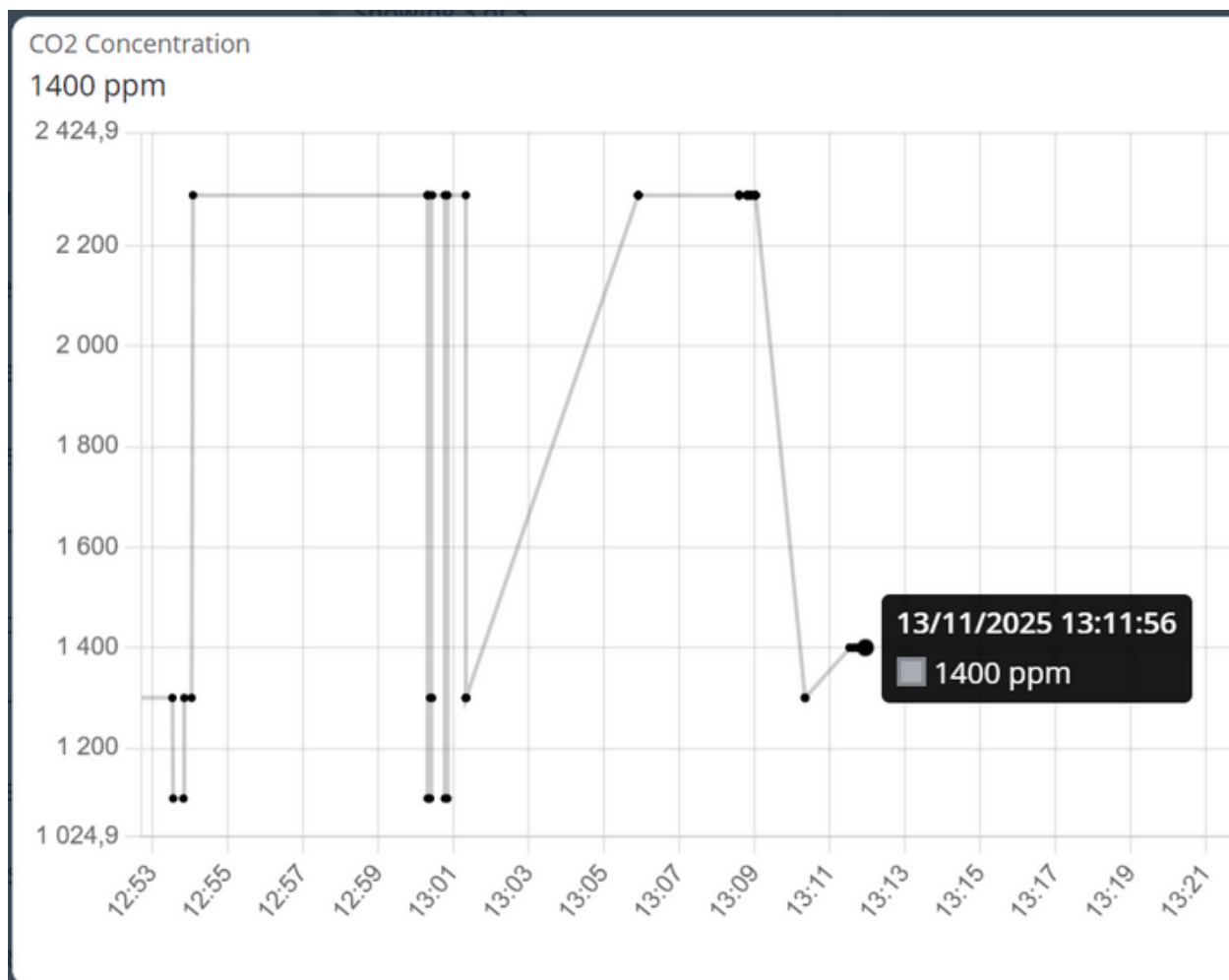


Рисунок 3.6 – Переданные данные в обработчик

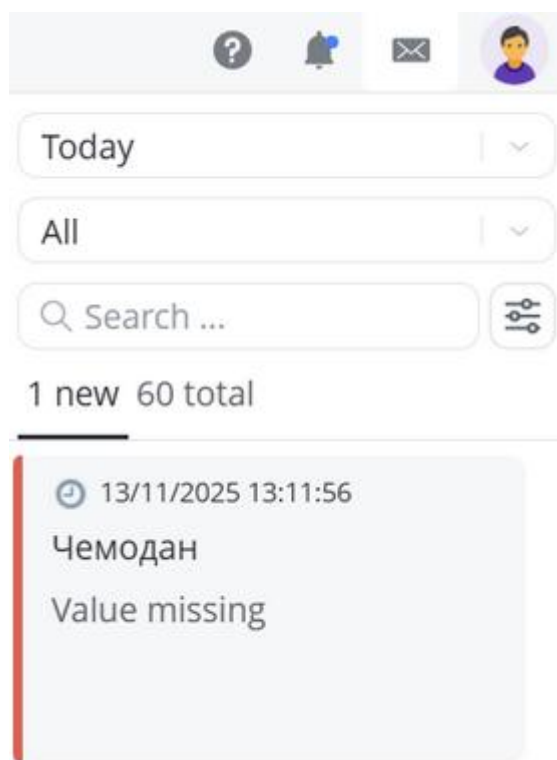


Рисунок 3.7 – Результат тестирования автомата второго задания

4. Практическая работа №12

Перед началом выполнения практической работы произошло настрой SMTP сервера.

Project of mihakom050@gmail.com / Project settings

Project Team Roles Access settings API Tokens Integrations

Сообщения

+ Create new

SMTP is a communication protocol for electronic mail transmission. Add an SMTP-server to be able to send email from [Automatons](#)

Message channel name * Сообщения

Description Describe the integration

Host * smtp.gmail.com

Port * 587

Username * mihakom050@gmail.com

Password *

Sender IoT Message

Connection security STARTTLS

Save Remove

Рисунок 4.1 – Настройка SMTP сервера
Далее было настроено отправляемое сообщение.

Send e-mail

SMTP gateway Сообщения

From mihakom050@gmail.com

Add parameter

To mihakom050@gmail.com

Add parameter

Subject High CO2

Add parameter

Mail body (text)

Превышен уровень CO2. Максимальный: 2000; текущий: {object.state["co2"]}).

Add parameter

Рисунок 4.2 – Настройка отправки ошибочного сообщения с концентрацией CO2

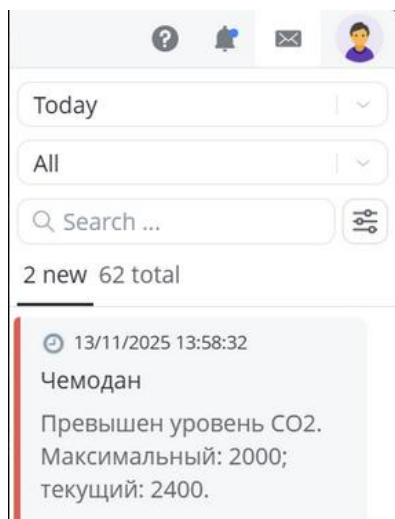


Рисунок 4.3 – Пример итогового ошибочного сообщения с концентрацией CO₂

И в конечном итоге был проведён тест корректности работы топиков.



Рисунок 4.4 – Получено письмо с ошибкой о концентрации CO₂
Далее было настроено следующее сообщение в соответствии с заданием.

Send e-mail

SMTP gateway

Сообщения

From

mihakom050@gmail.com

Add parameter

To

for.inst.techies@mail.ru

Add parameter

Subject

Invalid motion check value

Add parameter

Mail body (text)

Expected a boolean value. Got {{object.state["motion_check_value"]}}

Add parameter

Рисунок 4.5 – Настройка отправки ошибочного сообщения с наличием motion

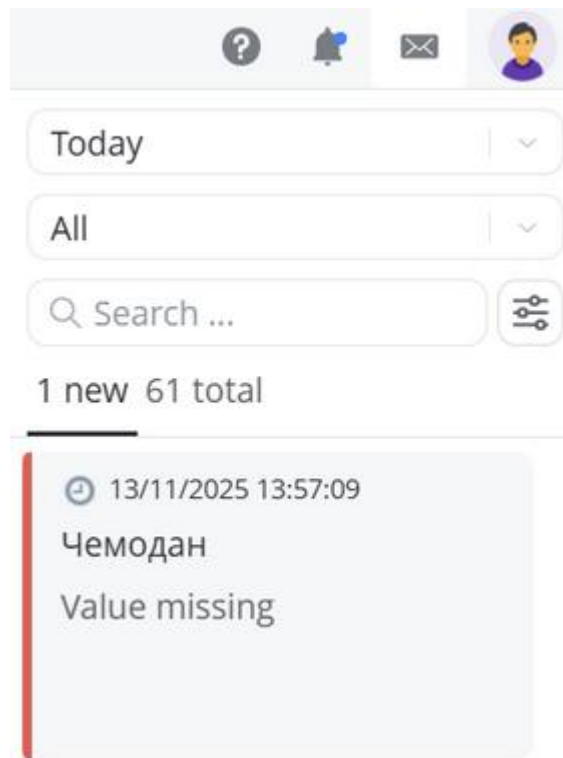


Рисунок 4.6 – Пример итогового ошибочного сообщения с motion

При этом была проведена проверка корректности работы сообщений отправкой сообщения на почту.

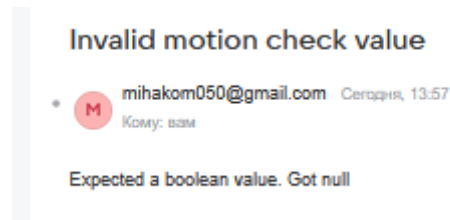


Рисунок 4.7 – Получено письмо с ошибкой о motion

ВЫВОД

В ходе данных практических работ было проведено ознакомление с функциональными возможностями сайта `dev.rightech.io` и проведена работа с динамически меняющимися данными. Полученные знания были закреплены путём выполнения практических заданий.