|  |
| --- |
|  |
| МИНОБРНАУКИ РОССИИ |
| Федеральное государственное бюджетное образовательное учреждение высшего образования  **«МИРЭА – Российский технологический университет»**  **РТУ МИРЭА** |

Институт: Информационных технологий (ИТ)

Кафедра: Базовая кафедра №234 — Управляющих ЭВМ

**КУРСОВАЯ РАБОТА**

|  |  |
| --- | --- |
| по дисциплине | Программное обеспечение систем управления технологическими процессам |
|  | (наименование дисциплины) |

|  |  |
| --- | --- |
| **Тема курсовой работы:** | Система управления для мониторинга и контроля работы системы вентиляции в зданиях |

|  |  |  |  |
| --- | --- | --- | --- |
| **Студент группы:** | ИКМО-05-23 Миронов Д.С. |  |  |
|  | (учебная группа, фамилия, имя, отчество) |  | (подпись) |

|  |  |  |  |
| --- | --- | --- | --- |
| **Руководитель** | доцент, к.т.н., Глухов А.В. |  |  |
| **курсовой работы:** | (должность, звание, ученая степень, фамилия, имя, отчество) |  | (подпись) |

**Работа предоставлена к защите до « » \_\_\_ 20\_\_г.**

**Допущена к защите до « » \_\_\_\_ 20\_\_ г.**

Москва 2023

**ОГЛАВЛЕНИЕ**

[**ТЕХНИЧЕСКОЕ ЗАДАНИЕ** 5](#_Toc194623514)

[**Общие требования:** 5](#_Toc194623515)

[**Задания на проектирование:** 5](#_Toc194623516)

[**ВВЕДЕНИЕ** 7](#_Toc194623517)

[**Актуальность темы** 7](#_Toc194623518)

[1. Номенклатура применяемых счетчиков ресурсов 8](#_Toc194623519)

[1.1. Создание объектов 8](#_Toc194623520)

[1.1.1. Создание окон 8](#_Toc194623521)

[1.1.1. Создание пользователя 10](#_Toc194623522)

[1.1.1. Создание протокола Modbus TCP 10](#_Toc194623523)

[1.1.1. Запуск системы 12](#_Toc194623524)

[**СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ** 14](#_Toc194623525)

**ТЕХНИЧЕСКОЕ ЗАДАНИЕ**

**Общие требования:**

Для работы необходимо использовать управляющие функции, которые будут осуществлять отображение и контроль технологического процесса. Параметры технологического процесса получать и отправлять на сервер. Спецификация сервера указана в задании.

Для каждого отслеживаемого параметра необходимо реализовать аварийное сообщение. Для аналоговых параметров реализовать верхние\нижние предупредительные и аварийные уставки.

Выдавать аварийные сообщение на экран АРМ. Сопровождать аварийное сообщение звуковой сигнализацией. Архивировать контролируемые параметры и все сообщения технологического процесса.

Реализовать аутентификацию пользователей и разграничение прав доступа различных категорий пользователей.

Создать сценарии аварийного отказа оборудования.

Разделить события на 2 группы — аварийные и события оборудования. События управления оборудованием вести в отдельном окне и выдавать по запросу оператора.

Реализовать графическое отображение технологического процесса. Отобразить схему взаимодействия компонентов технологического оборудования с АРМ оператора.

**Задания на проектирование:**

Разработать систему управления для мониторинга и контроля работы системы вентиляции в зданиях. Данные получать с сервера Modbus-TCP. АСУ должна обеспечивать мониторинг основных параметров работы системы вентиляции, таких как температура, влажность, уровень CO2, скорость воздушного потока, давление и т.д. SCADA-система должна обеспечивать возможность управления системой вентиляции внутри здания. Управление системой включает: регулирование скорости вентиляторов, контроль открытия и закрытия клапанов, регулирование расхода воздуха для разных зон в здании. SCADA-система должна обеспечивать мониторинг качества воздуха в здании, такого как содержание CO2, VOC (летучие органические соединения), пыль и другие загрязнители в воздухе.

SCADA-система должна обладать функцией обнаружения аварийных ситуаций, таких как отказ вентилятора, перекрытие вентиляционных отверстий, понижение или повышение давления воздуха и т.д. При обнаружении аварийной ситуации система должна немедленно оповестить оператора и предпринять необходимые меры, например, отправить оповещение или команду на восстановление нормальной работы.

**ВВЕДЕНИЕ**

## **Актуальность темы**

В современных условиях автоматизация технологических процессов играет ключевую роль в повышении эффективности и безопасности промышленных объектов. Распределенные системы управления, объединяющие разнообразные датчики и мониторы, позволяют оперативно собирать и анализировать данные, что значительно улучшает качество управления и позволяет своевременно реагировать на внештатные ситуации.

1. Создание системы управления
   1. Создание объектов

Создаем объекты для системы.

Понадобятся следующие ресурсы:

* Температура
* Влажность
* Уровень CO2
* Скорость воздушного потока
* Давление
* Скорость вентилятор
* ВКЛ/ВЫКЛ затвор
* ВКЛ/ВЫКЛ фильтр
* Мощность водяного калорифера (кВт)
* Установка пользовательской температуры
* Пользовательская температура
* Вентиляция
* Фильтр
* Водяной калорифер (кВт)
* Sound

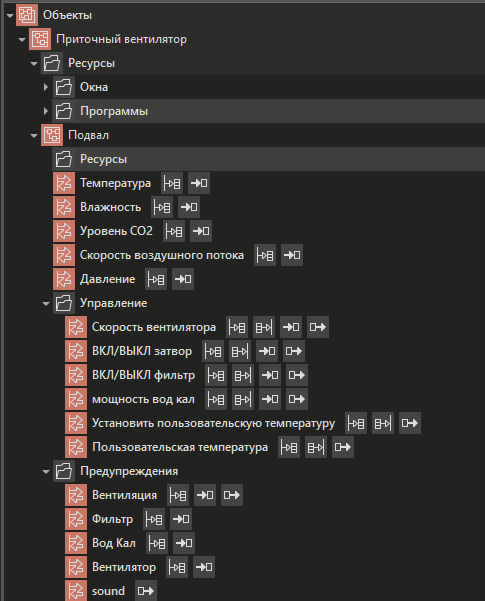


Рисунок – 1

* + 1. Создание окон

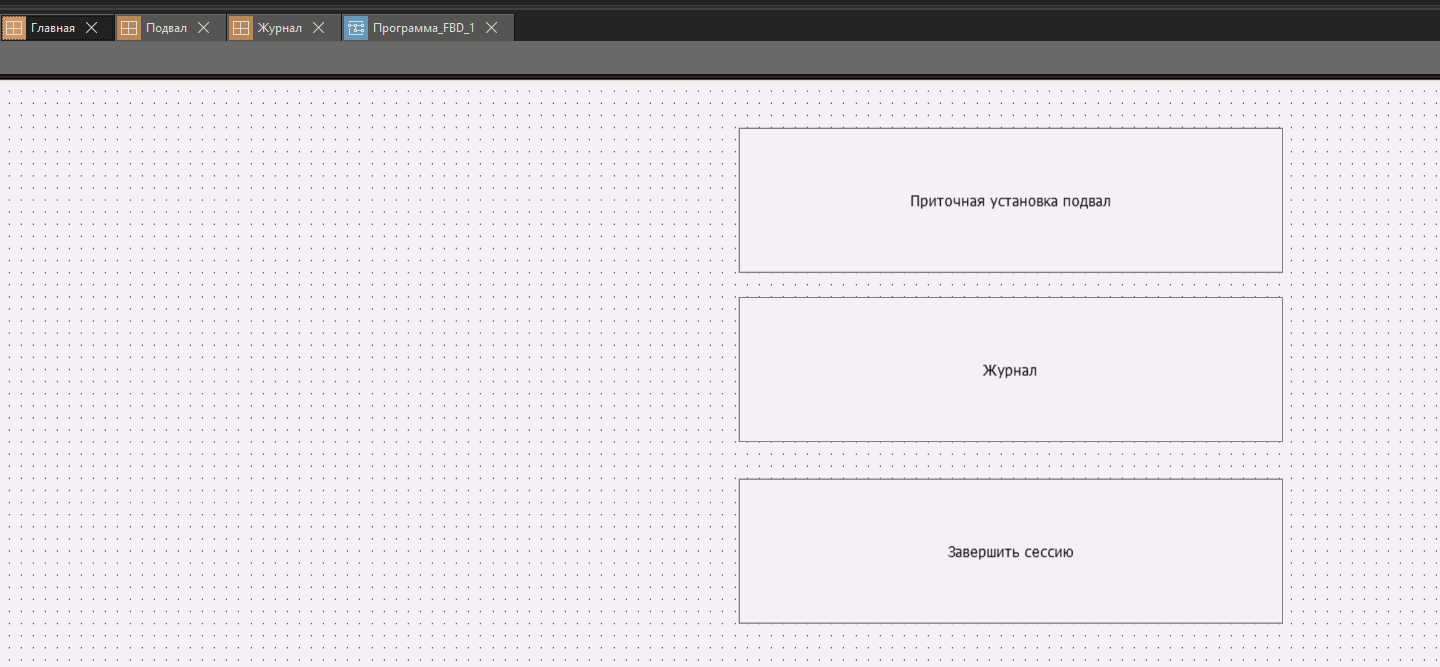


Рисунок – 2

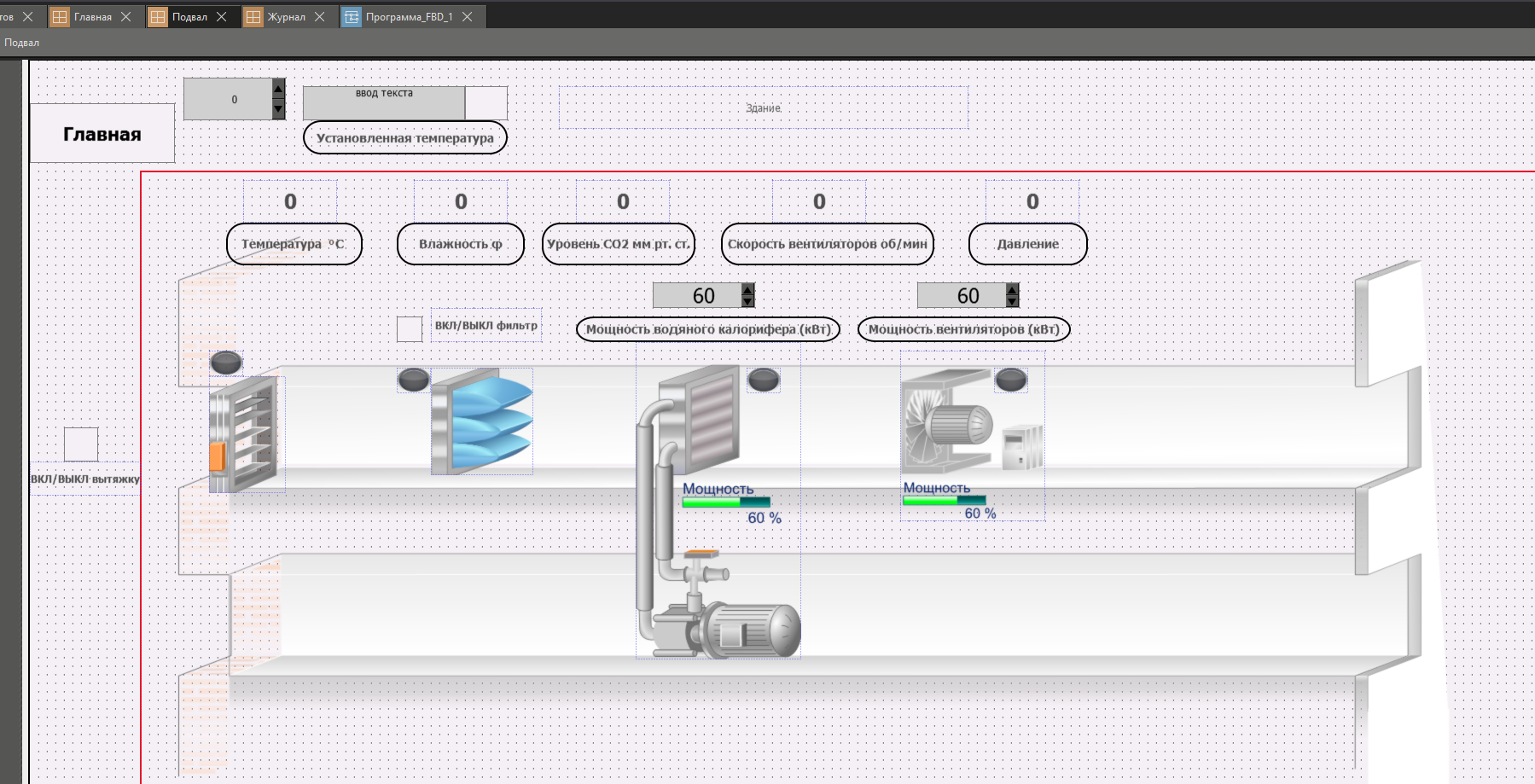


Рисунок – 3

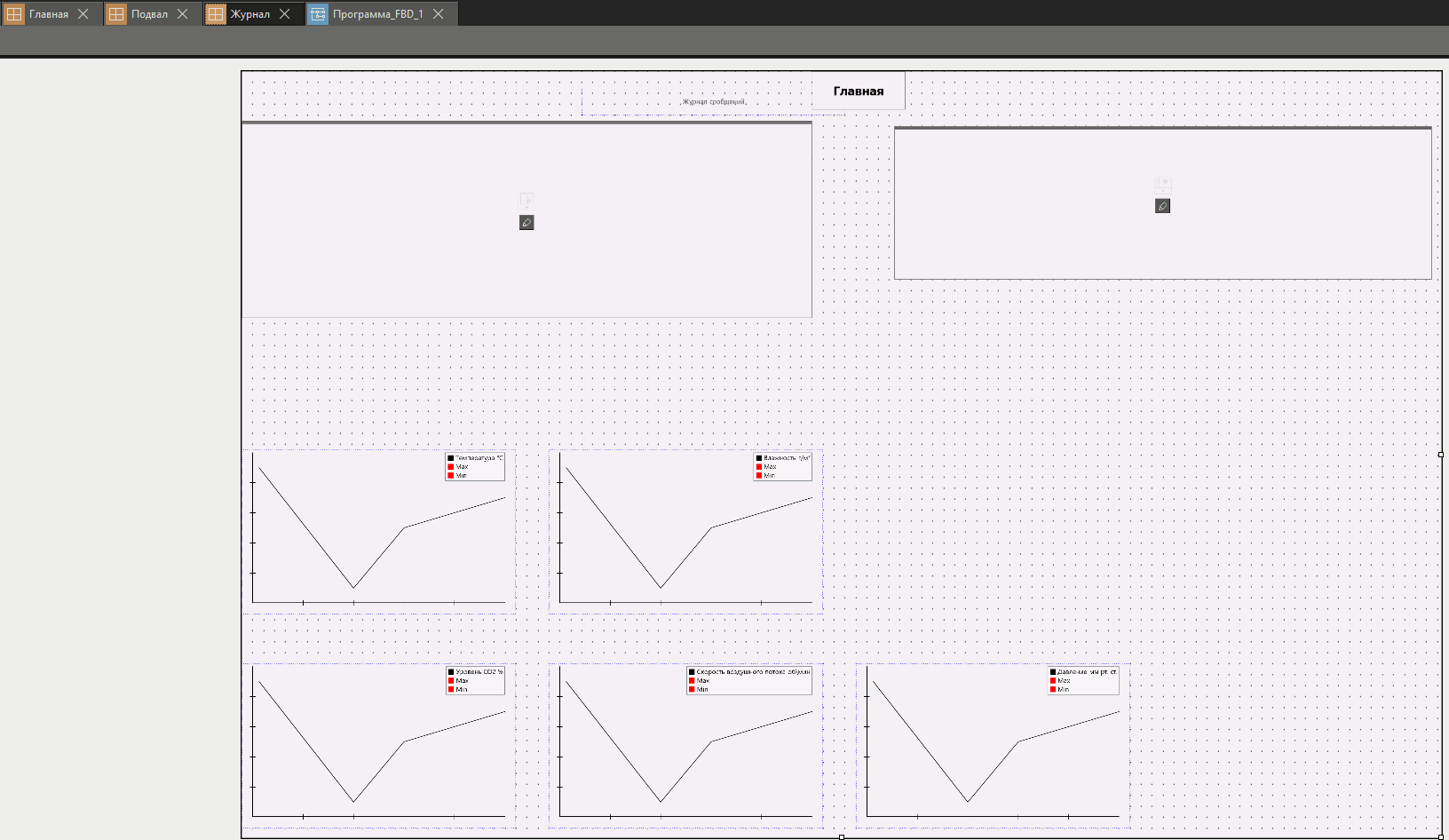


Рисунок – 4

* + 1. Создание пользователя

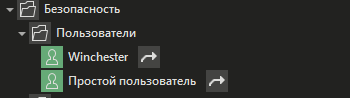


Рисунок – 5

* + 1. Создание протокола Modbus TCP

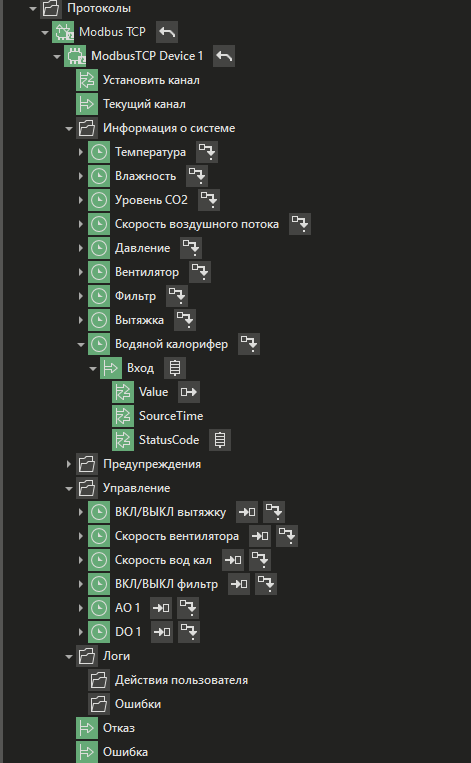


Рисунок – 6

|  |  |
| --- | --- |
| **Название** | **Адрес** |
| Влажность | 1 |
| Температура | 2 |
| CO2 | 4 |
| Давление | 5 |
| Скорость воздушного потока | 6 |
| вкл выкл вент | 11 |
| мощность вент | 12 |
| мощность вод кал | 13 |
| вкл выкл фильтр |  |
| Установить температуру | 20 |
| Установить температуру | 21 |

* + 1. Запуск системы

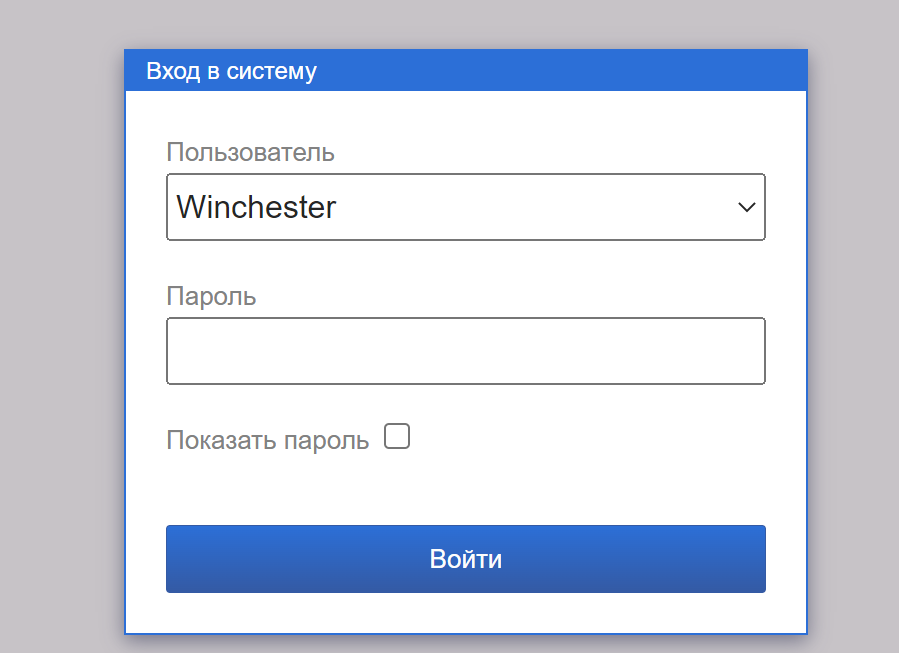


Рисунок – 7

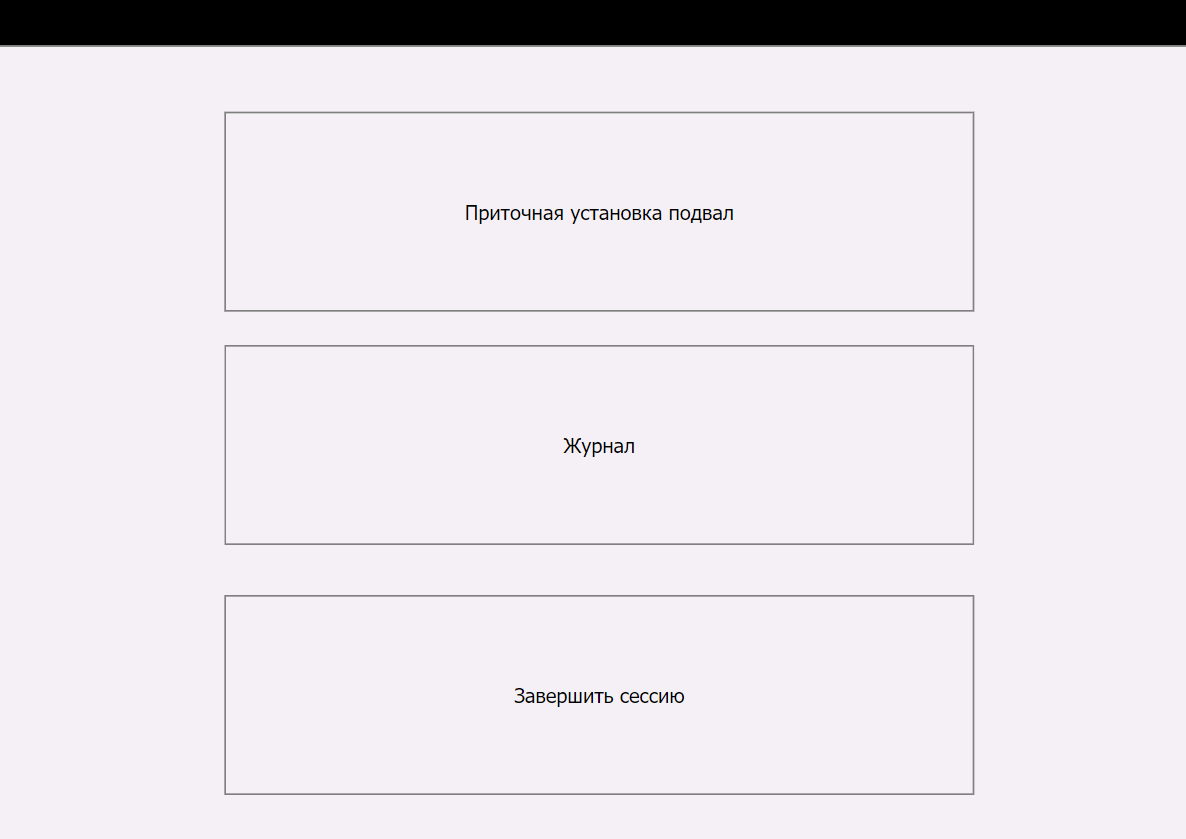


Рисунок – 8

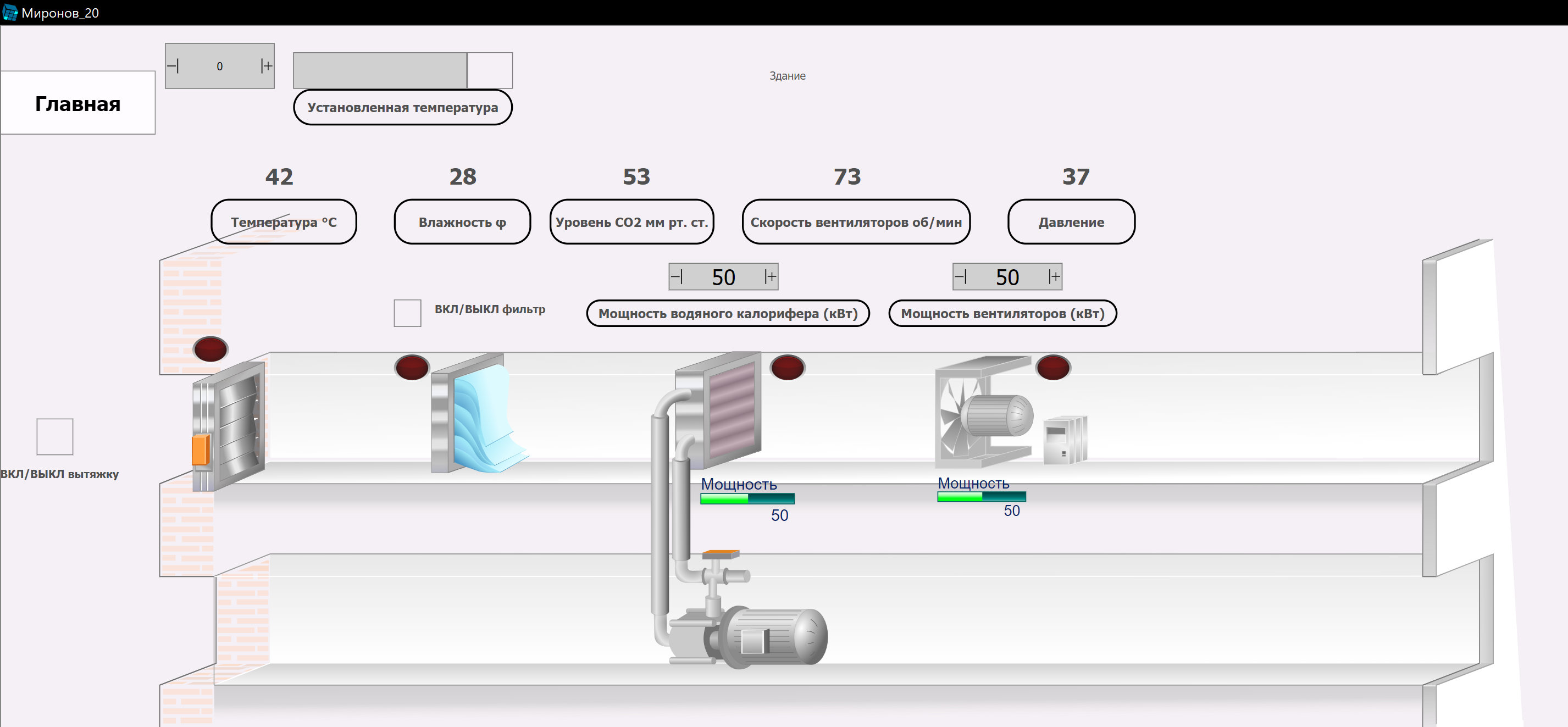


Рисунок – 9

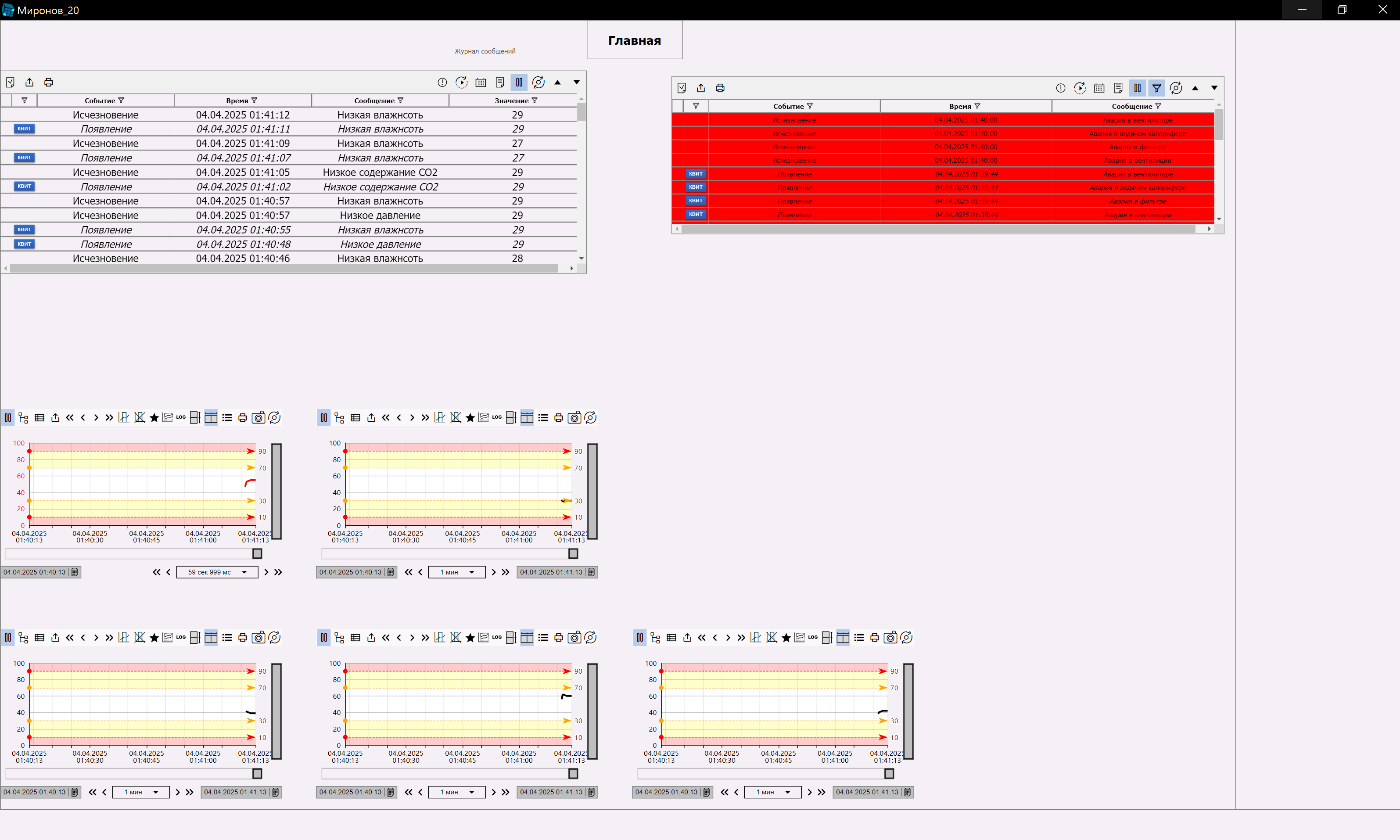


Рисунок – 10

Код сервера TCP

import time

from pymodbus.server import StartTcpServer

from pymodbus.datastore import ModbusSlaveContext, ModbusServerContext, ModbusSequentialDataBlock

from threading import Thread

import random

from datetime import datetime

#выводить журнал + (реализовать верхние\нижние предупредительные и )

# добавить второго пользователя

# первый этаж - стартовый

# добавить звук

# лист дейсвтий пользователя

# журнал аварий

# Создаем таблицу регистров (100 значений) с правильной инициализацией

store = ModbusSlaveContext(

di=ModbusSequentialDataBlock(0, [0] \* 100), # Дискретные входы

co=ModbusSequentialDataBlock(0, [0] \* 100), # Катушки (битовые значения)

hr=ModbusSequentialDataBlock(0, [50] \* 100), # 100 значений в Holding Registers

ir=ModbusSequentialDataBlock(0, [0] \* 100) # Входные регистры

)

context = ModbusServerContext(slaves=store, single=True)

# Таймеры отказа датчиков

sensor\_failure\_timers = {

'Вытяжка': 20000, # 20

'Фильтр': 25000, # 25

'Вод\_кал': 30000, # 30

'Вентилятор': 35000, # 35

}

address\_error = {

'Вытяжка': 15,

'Фильтр': 16,

'Вод\_кал': 17,

'Вентилятор': 18,

}

address\_error\_status = {

'Вытяжка': False,

'Фильтр': False,

'Вод\_кал': False,

'Вентилятор': False,

}

# Счетчик времени для плавного уменьшения влажности

humidity\_decrease\_counter = 0

# Флаги отказов датчиков

sensor\_failures = {

"humidity": False,

"temperature": False,

"irrigation": False,

"ventilation": False,

"check\_fan":False,

"check\_speed\_fan":60,

"check\_water\_cal":60,

"check\_filtr":False

}

local\_state= {}

logs\_actions= []

logs\_system=[]

logs\_errors =[]

def main\_function( humidity\_decrease\_counter ):

while True:

# Чтение значений из Modbus регистров

humidity = store.getValues(3, 1, count=1)[0] # Влажность (адрес 1)

temperature = store.getValues(3, 2, count=1)[0] # Температура (адрес 2)

c\_o\_2 = store.getValues(3, 4, count=1)[0] # CO2 (адрес 4)

pressure = store.getValues(3, 5, count=1)[0] # Давление (адрес 5)

airflow\_speed = store.getValues(3, 6, count=1)[0] # Скорость воздушного потока (адрес 6)

check\_fan = store.getValues(3, 11, count=1)[0] # вкл выкл вент (адрес 11)

check\_speed\_fan = store.getValues(3, 12, count=1)[0] # мощность вент (адрес 12)

check\_water\_cal = store.getValues(3, 13, count=1)[0] # мощность вод кал (адрес 13)

check\_filtr = store.getValues(3, 14, count=1)[0] # вкл выкл фильтр (адрес 14)

temperature\_user = store.getValues(3, 21, count=1)[0] # Установить температуру (адрес 21) # int

temperature\_user\_use = store.getValues(3, 20, count=1)[0] # Установить температуру (адрес 20) # bool

if temperature\_user\_use==1:

temperature\_user\_use=True

else:

temperature\_user\_use=False

sensor\_failures['check\_fan']= True if isinstance(check\_fan, int) else check\_fan

sensor\_failures['check\_speed\_fan']= check\_speed\_fan

sensor\_failures['check\_water\_cal']= check\_water\_cal

sensor\_failures['check\_filtr']= True if isinstance(check\_fan, int) else check\_filtr

local\_state ={ 'Вытяжка': sensor\_failures['check\_fan'],

'мощность вент': sensor\_failures['check\_speed\_fan'],

'мощность вод кал': sensor\_failures['check\_water\_cal'],

'Вкл/Выкл фильтр': sensor\_failures['check\_filtr'],

'Time':str(datetime.now())

}

if logs\_system and temperature\_user\_use:

temp = logs\_system[-1]

if list(temp.values()) != list(local\_state.values()):

logs\_system.append(local\_state)

else:

logs\_system.append(local\_state)

# Моделирование отказа датчиков

for name, time\_time in sensor\_failure\_timers.items():

if sensor\_failure\_timers[name] > 0:

sensor\_failure\_timers[name] -= 1

store.setValues(3, address\_error[name], [False]) # Сигнал об отказе

elif sensor\_failure\_timers[name] == 0:

store.setValues(3, address\_error[name], [True]) # Сигнал об отказе

address\_error\_status[name]=True

logs\_errors.append( address\_error | { 'Time':str(datetime.now())} )

if temperature\_user\_use and temperature\_user: # если пользовательская настройка

if temperature\_user > temperature:

sensor\_failures['check\_fan']= True

if check\_speed\_fan>40:

check\_speed\_fan=check\_speed\_fan-1

elif check\_speed\_fan<40:

check\_speed\_fan=check\_speed\_fan+1

sensor\_failures['check\_speed\_fan']= check\_speed\_fan

if check\_water\_cal>40:

check\_water\_cal=check\_water\_cal-1

elif check\_water\_cal<40:

check\_water\_cal=check\_water\_cal+1

sensor\_failures['check\_water\_cal']= check\_water\_cal

sensor\_failures['check\_filtr']= True

temperature =temperature + 1

if humidity>40:

humidity = humidity -1

elif humidity<40:

humidity = humidity +1

if c\_o\_2>40:

c\_o\_2 = c\_o\_2 -1

elif c\_o\_2<40:

c\_o\_2 = c\_o\_2 +1

if pressure>40:

pressure = pressure -1

elif pressure<40:

pressure = pressure +1

if airflow\_speed>40:

airflow\_speed = airflow\_speed -1

elif c\_o\_2<40:

airflow\_speed = airflow\_speed +1

elif temperature\_user < temperature:

sensor\_failures['check\_fan']= True

if check\_speed\_fan<70:

check\_speed\_fan=check\_speed\_fan+1

elif check\_speed\_fan>70:

check\_speed\_fan=check\_speed\_fan-1

sensor\_failures['check\_speed\_fan']= check\_speed\_fan

if check\_water\_cal<70:

check\_water\_cal=check\_water\_cal+1

elif check\_water\_cal>70:

check\_water\_cal=check\_water\_cal-1

sensor\_failures['check\_water\_cal']= check\_water\_cal

sensor\_failures['check\_filtr']= True

temperature =temperature - 1

if humidity>30:

humidity = humidity -1

elif humidity<40:

humidity = humidity +1

if c\_o\_2>30:

c\_o\_2 = c\_o\_2 -1

elif c\_o\_2<30:

c\_o\_2 = c\_o\_2 +1

if pressure>30:

pressure = pressure -1

elif pressure<30:

pressure = pressure +1

if airflow\_speed>30:

airflow\_speed = airflow\_speed -1

elif c\_o\_2<30:

airflow\_speed = airflow\_speed +1

else:

sensor\_failures['check\_filtr']= False

sensor\_failures['check\_fan']= False

humidity = humidity+ random.randint(1,7) if '+' == random.choice(["+","-"]) else humidity- random.randint(1,7)

temperature =temperature + random.randint(1,7) if '+' == random.choice(["+","-"]) else temperature - random.randint(1,7)

c\_o\_2 = c\_o\_2 + random.randint(1,7) if '+' == random.choice(["+","-"]) else c\_o\_2 - random.randint(1,7)

pressure =pressure + random.randint(1,7) if '+' == random.choice(["+","-"]) else pressure - random.randint(1,7)

airflow\_speed =airflow\_speed + random.randint(1,7) if '+' == random.choice(["+","-"]) else airflow\_speed - random.randint(1,7)

tmp = {

'Влажность': humidity,

'Температура': temperature,

'CO2': c\_o\_2,

'Давление': pressure,

'Скорость воздушного потока': airflow\_speed,

'Вытяжка': sensor\_failures['check\_fan'],

'мощность вент': sensor\_failures['check\_speed\_fan'],

'мощность вод кал': sensor\_failures['check\_water\_cal'],

'вкл выкл фильтр': sensor\_failures['check\_filtr'],

'Time':str(datetime.now())

}

logs\_system.append( tmp)

# Запись значений в Modbus

store.setValues(3, 1, [humidity]) # Влажность (адрес 1)

store.setValues(3, 2, [temperature]) # Температура (адрес 2)

store.setValues(3, 4, [c\_o\_2]) # CO2 (адрес 4)

store.setValues(3, 5, [pressure]) # Давление (адрес 5)

store.setValues(3, 6, [airflow\_speed]) # Скорость воздушного потока (адрес 6)

store.setValues(3, 11, [sensor\_failures['check\_fan']]) # вкл выкл (адрес 11)

store.setValues(3, 12, [sensor\_failures['check\_speed\_fan']]) # мощность вент (адрес 12)

store.setValues(3, 13, [sensor\_failures['check\_water\_cal']]) # мощность вод кал (адрес 13)

store.setValues(3, 14, [sensor\_failures['check\_filtr']]) # вкл выкл фильтр (адрес 14)

store.setValues(3, 21, [temperature\_user]) #

store.setValues(3, 20, [temperature\_user\_use]) #

print( f"Влажность={humidity}%, Температура={temperature}°C, "

f"CO2 {c\_o\_2}%, "

f"Давление {pressure}мм рт. ст., "

f"Скорость воздушного потока {airflow\_speed}об/мин, "

f"Заглушка {'открыта' if sensor\_failures['check\_fan'] else 'закрыта'} {sensor\_failures['check\_fan']}, "

f"Фильтр {'вкл' if sensor\_failures['check\_filtr'] else 'выкл'} {sensor\_failures['check\_fan']}, "

f"скорость вод кал {check\_water\_cal}, "

f"скорость вент{sensor\_failures['check\_speed\_fan']}, "

f"Errors: {sensor\_failure\_timers}, "

f"temperature\_user {temperature\_user},"

f"temperature\_user\_use {temperature\_user\_use}"

)

time.sleep(2) # Обновление каждые 5 секунд

# Запускаем обновление данных в отдельном потоке

Thread(target=main\_function, daemon=True, args=(humidity\_decrease\_counter, ) ).start()

# Запускаем сервер

print("Modbus-TCP сервер запущен на 127.0.0.1:5020...")

StartTcpServer(context, address=("127.0.0.1", 5020))

**СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

1. YF-S201 датчик расхода воды: [Электронный ресурс] — URL: https://duino.ru/Schetchik-rashoda-vody.html/ (дата обращения: 12.05.2024).
2. ВСЦ-15 класс В LoRaWAN - ВОДОМЕР: [Электронный ресурс] — URL: https://www.vodomer.su/catalog/schetchiki-vody-i-raskhodomery/kvartirnye-schetchiki-vody/schetchik-vody-universalnyy-du-15/ (дата обращения: 12.05.2024).
3. Электронный счетчик Ду15 RS-485 модель 1: [Электронный ресурс] — URL: https://pulsarm.ru/products/schetchik-vody/kvartirnyy-schyetchik-vody-du-15-du-20/elektronnyy-schetchik-du15-rs-485-qn-1-5-m3-ch-l-80mm-prisoediniteli-v-komplekte-/ (дата обращения: 12.05.2024).
4. Меркурий 200: [Электронный ресурс] — URL: https://www.incotexcom.ru/catalogue/200 (дата обращения: 12.05.2024).
5. Электросчетчик СЭ-310: [Электронный ресурс] — URL: http://www.energomera.ru/ru/products/meters/ce301r33 (дата обращения: 12.05.2024).
6. Счетчик газа СГБМ-1,6 Бетар: [Электронный ресурс] — URL: https://xn----7sbajcomicunrr2bq2fc.xn--p1ai/magazin/product/schetchik-gaza-sgbm-1-6-betar (дата обращения: 12.05.2024).
7. Счетчик газа Сигнал СГК-1.6: [Электронный ресурс] — URL: https://clck.ru/3As6ay (дата обращения: 12.05.2024).
8. Счетчик газа «Тепловодомер ВК G4»: [Электронный ресурс] — URL: https://gazovye-schetchiki.ru/bk-g4/(дата обращения: 12.05.2024).
9. APC by Schneider Electric Back-UPS BX950MI: [Электронный ресурс] — URL: https://www.apc.com/kz/ru/product/BX950MI-GR/apc-backups-950va-tower-230v-4x-cee-7-7-schuko-outlets-avr/ (дата обращения: 12.05.2024).
10. Arduino UNO: [Электронный ресурс] — URL: https://duino.ru/arduino-uno-r3.html/ (дата обращения: 12.05.2024).
11. Raspberry Pi 4: [Электронный ресурс] — URL: https://amperka.ru/product/raspberry-pi-4-model-b-4-gb (дата обращения: 12.05.2024).