Питон:

import time

import serial

from email.mime.multipart import MIMEMultipart

from email.mime.text import MIMEText

import smtplib

import imaplib

import email

import random

msg = MIMEMultipart()

TO = 'smart.house.diplom@yandex.ru' #pass LJeTI2021yandex

GMAIL\_USER = 'smart.house.diplom@gmail.com'

GMAIL\_PASS = 'LJeTI2021'

password = GMAIL\_PASS

msg['From'] = GMAIL\_USER

msg['To'] = TO

msg['Subject'] = "Пожар!"

server = smtplib.SMTP('smtp.gmail.com: 587')

server.starttls()

server.login(msg['From'], password)

ser = serial.Serial('COM3', 9600)

mail = imaplib.IMAP4\_SSL('imap.gmail.com')

mail.login(GMAIL\_USER, GMAIL\_PASS)

flag = 0

timing = time.time()

while True:

if flag == 0:

message\_ser = str(ser.readline())

print(message\_ser)

if message\_ser[2] == 'F' :

server = smtplib.SMTP('smtp.gmail.com: 587')

server.starttls()

server.login(msg['From'], password)

msg = MIMEMultipart()

msg['From'] = GMAIL\_USER

msg['To'] = TO

msg['Subject'] = "Пожар!"

num = random.randint(100, 999)

message = 'Датчик обнаружил возгорание! Если Вы наблюдате пожар, оправте в ответ этот код: '+str(num)

msg.attach(MIMEText(message, 'plain'))

server.sendmail(msg['From'], msg['To'], msg.as\_string())

print("successfully sent email to:"+ msg['To'])

timing = time.time()

flag = 1

if message\_ser[2] == 'H':

server = smtplib.SMTP('smtp.gmail.com: 587')

server.starttls()

server.login(msg['From'], password)

msg = MIMEMultipart()

msg['From'] = GMAIL\_USER

msg['To'] = TO

msg['Subject'] = "Взлом!"

num = random.randint(100, 999)

message = 'Произошла попытка взлома двери!'

msg.attach(MIMEText(message, 'plain'))

server.sendmail(msg['From'], msg['To'], msg.as\_string())

print("successfully sent email to:"+ msg['To'])

if message\_ser[2] == 'W':

server = smtplib.SMTP('smtp.gmail.com: 587')

server.starttls()

server.login(msg['From'], password)

msg = MIMEMultipart()

msg['From'] = GMAIL\_USER

msg['To'] = TO

msg['Subject'] = "Протечка!"

num = random.randint(100, 999)

message = 'Обнаружена протечка в системе водоснабжения!'

msg.attach(MIMEText(message, 'plain'))

server.sendmail(msg['From'], msg['To'], msg.as\_string())

print("successfully sent email to:"+ msg['To'])

if message\_ser[2] == 'A' :

server = smtplib.SMTP('smtp.gmail.com: 587')

server.starttls()

server.login(msg['From'], password)

msg = MIMEMultipart()

msg['From'] = GMAIL\_USER

msg['To'] = TO

msg['Subject'] = "Климат"

num = random.randint(100, 999)

message = 'Температура:'+message\_ser.split(":")[1]+' Влажность:'+message\_ser.split(":")[2]

msg.attach(MIMEText(message, 'plain'))

server.sendmail(msg['From'], msg['To'], msg.as\_string())

print("successfully sent email to:"+ msg['To'])

elif flag == 1:

mail.list()

mail.select("inbox")

result, data = mail.search(None, "ALL")

ids = data[0]

id\_list = ids.split()

latest\_email\_id = id\_list[-1]

result, data = mail.fetch(latest\_email\_id, "(RFC822)")

raw\_email = data[0][1]

raw\_email\_string = raw\_email.decode('utf-8')

email\_message = email.message\_from\_string(raw\_email\_string)

body = email\_message.get\_payload()

if int(body[:3]) == num: #подтвердили

flag = 3

else:

if time.time() - timing > 50.0:

flag = 2

elif flag == 2:

new\_mes = float(message\_ser.split(":")[1].split("\\")[0])

if new\_mes >= 20.0:

msg = MIMEMultipart()

msg['From'] = GMAIL\_USER

msg['To'] = TO

msg['Subject'] = "Пожар!"

num = random.randint(100, 999)

message = 'Дома поднилась слишком высокая температура! Вы не ответели на певрое уведомление. Если Вы НЕ наблюдаете пожар, оправте в ответ этот код: '+str(num)+', в противном случае будут вызваны пожарные.'

msg.attach(MIMEText(message, 'plain'))

server.sendmail(msg['From'], msg['To'], msg.as\_string())

print("successfully sent email to:"+ msg['To'])

timing = time.time()

flag = 4

else:

flag = 0

elif flag == 3:

ser.write('CONFIRM'.encode('utf-8'))

flag = 0

timing = time.time()

elif flag == 4:

mail.list()

mail.select("inbox")

result, data = mail.search(None, "ALL")

ids = data[0]

id\_list = ids.split()

latest\_email\_id = id\_list[-1]

result, data = mail.fetch(latest\_email\_id, "(RFC822)")

raw\_email = data[0][1]

raw\_email\_string = raw\_email.decode('utf-8')

email\_message = email.message\_from\_string(raw\_email\_string)

body = email\_message.get\_payload()

if int(body[:3]) == num: #подтвердили

flag = 0

else:

if time.time() - timing > 50.0:

flag = 3

time.sleep(0.5)

Арудино:

//Подключение библиотек

#include "DHT.h"

#include <Wire.h>

#include <SPI.h>

#include <MFRC522.h> // библиотека "RFID".

//Обявленеи констант пинов

#define HUMANPIN A0

#define SVETPIN A1

#define WATERPIN A2

#define DHTPIN 2

#define BUTTONPIN 3

#define SVETPIND 4

#define OOPIN\_ 5

#define DOPIN\_ 6

#define FLAMEPIN 7

#define PSPIN\_ 8

#define RST\_PIN 9

#define SS\_PIN 10

#define DHTTYPE DHT22

//Обявление переменых нужных для работы

long minSecsBetweenEmails = 60;

unsigned long uidDec, uidDecTemp;

long lastSend = -60000;

long lastSend\_klimat = -60000;

bool ps = false;

bool rr = false;

bool sigRFID = false;

bool sigwater = false;

bool oo\_mode = false;

bool human = false;

bool button;

int time\_now = 0;

//Обявленеи модулей

DHT dht(DHTPIN, DHTTYPE);

MFRC522 mfrc522(SS\_PIN, RST\_PIN);

void setup() {

//Инициализация

Serial.begin(9600);

dht.begin();

SPI.begin();

mfrc522.PCD\_Init();

//Режимы пинов

pinMode(OOPIN\_, OUTPUT);

pinMode(DOPIN\_, OUTPUT);

pinMode(PSPIN\_, OUTPUT);

pinMode(DHTPIN, INPUT);

pinMode(WATERPIN, INPUT);

pinMode(SVETPIND, INPUT);

pinMode(SVETPIN, INPUT);

pinMode(FLAMEPIN, INPUT);

pinMode(BUTTONPIN, INPUT);

button = digitalRead(BUTTONPIN);

}

void loop() {

//Проверка активности сигнализации

if (ps || rr || sigRFID || sigwater) {

return;

}

//Считыванеи температуры, влажности, текущего времени и состояния кнопки для режимов освещения

float humidity = dht.readHumidity();

float temperature = dht.readTemperature();

long now = millis();

if (digitalRead(BUTTONPIN) != button) {

if (oo\_mode) {

oo\_mode = false;

} else {

oo\_mode = true;

}

}

//Активизация пожарной сигнализации, в случае пожара

if (Serial.readString() == "CONFIRM") {

ps = true;

digitalWrite(OOPIN\_, HIGH);

digitalWrite(DOPIN\_, HIGH);

digitalWrite(PSPIN\_, HIGH);

}

if (digitalRead(FLAMEPIN) == LOW) {

if (now > (lastSend + minSecsBetweenEmails \* 1000)) {

Serial.println("FLAME:" + String(temperature));

lastSend = now;

}

}

//Активизация сигнализации в случае протечки

if (analogRead(WATERPIN) > 200) {

sigwater = true;

Serial.println("WATER");

digitalWrite(PSPIN\_, HIGH);

}

//Систма сканирования RFID ключей

if ( mfrc522.PICC\_IsNewCardPresent() && mfrc522.PICC\_ReadCardSerial()) {

uidDec = 0;

// Выдача серийного номера метки.

for (byte i = 0; i < mfrc522.uid.size; i++)

{

uidDecTemp = mfrc522.uid.uidByte[i];

uidDec = uidDec \* 256 + uidDecTemp;

}

if (uidDec != 1920115773) {

sigRFID = true;

digitalWrite(PSPIN\_, HIGH);

Serial.println("HUMAN");

}

}

//Получение информации с датчика движения, в случае отвутствия людей выключаем всё освещение

if (analogRead(HUMANPIN) > 500) {

human = true;

} else {

human = false;

}

if (!human) {

digitalWrite(OOPIN\_, LOW);

digitalWrite(DOPIN\_, LOW);

return;

}

// Отравка текужей температуры и влажности

if (now > (lastSend\_klimat + minSecsBetweenEmails \* 1000)) {

lastSend\_klimat = now;

Serial.println("ANSWERKLIMAT:" + String(temperature) + ":" + String(humidity) + ":");

}

//Регулирование освещености по режимам

if (oo\_mode) {

analogWrite(OOPIN\_, analogRead(SVETPIN) / 4);

} else {

digitalWrite(OOPIN\_, LOW);

if (digitalRead(SVETPIND) == HIGH) {

digitalWrite(DOPIN\_, HIGH);

} else {

digitalWrite(DOPIN\_, LOW);

}

}

delay(500);

}