Smart Office Monitoring Report

1. Team Information

Team ID	6058544998518381825
Mr. Pratik Ashok Girhale	Team Leader
Miss. Sanskruti Rajesh Gawali	Team Member
Mr. Sahil Sanjay Sahare	Team Member
Miss. Rashmi Ramesh Chaudhari	Team Member

2. Introduction

In current scenario human being facing a very physical and mental issue and in the corporate life it adds over problems. Due to that they face many medical issues and due to unstable temperature, humidity, air quality it increases day by day. In our workplace we need a good environment because if the environment is good the output is automatically coming good.

In this project we are going to implement automatic healing of the any condition it automatic set the level of temperature for the human being as well as set the air quality and the other issue. It also monitors the water level to protect the loss of water.

In this project industrial IOT is very important to execute the model

3. Methodology

- Firstly we find out the problems.
- Then think about the solution on it.
- Then we find one solution and draw the block diagram according to it.
- After that we make an circuit and give the code to it.
- After that we get the results
- We use many IOT platform to execute our model.

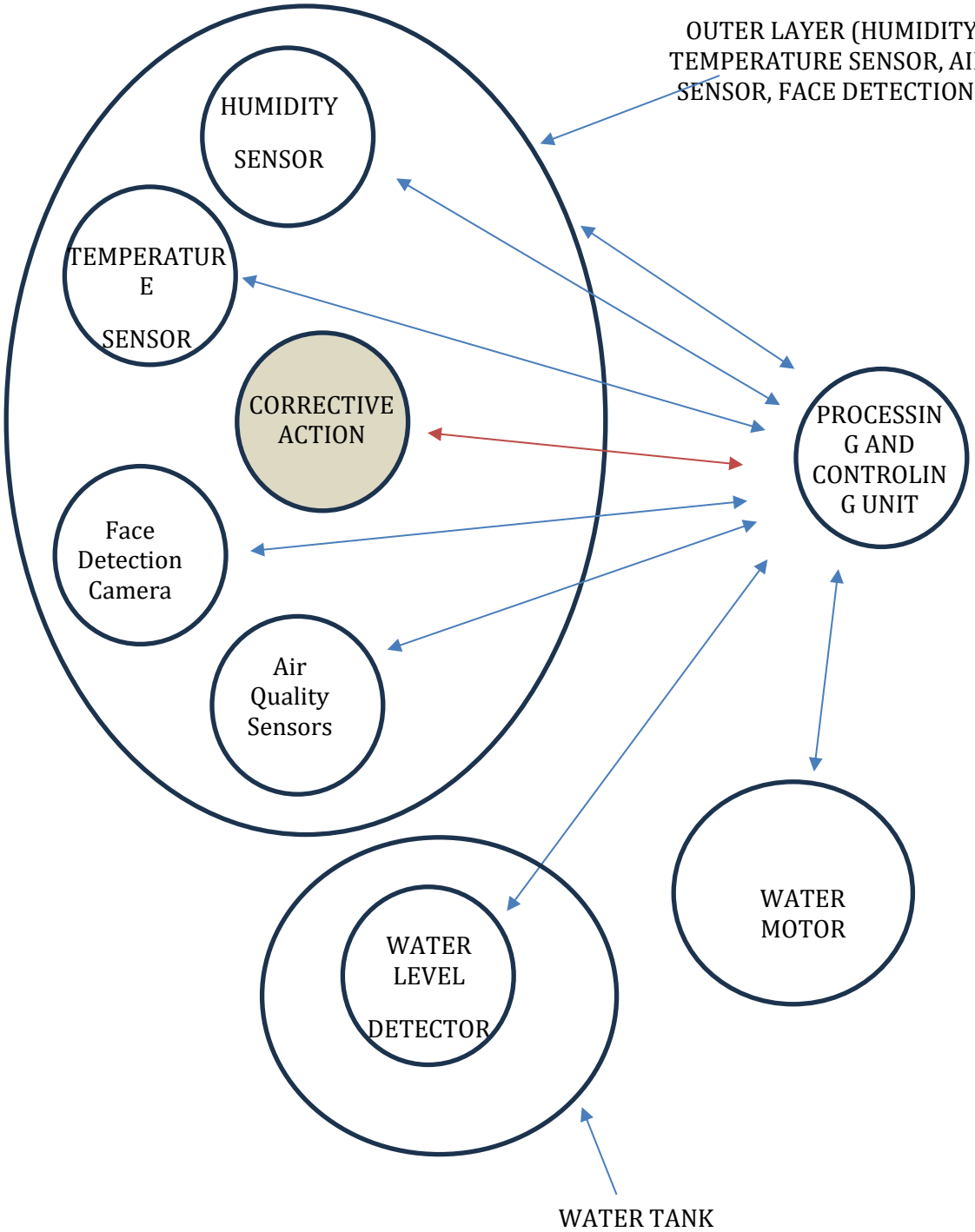

Fig. Block diagram of Smart Office Monitoring

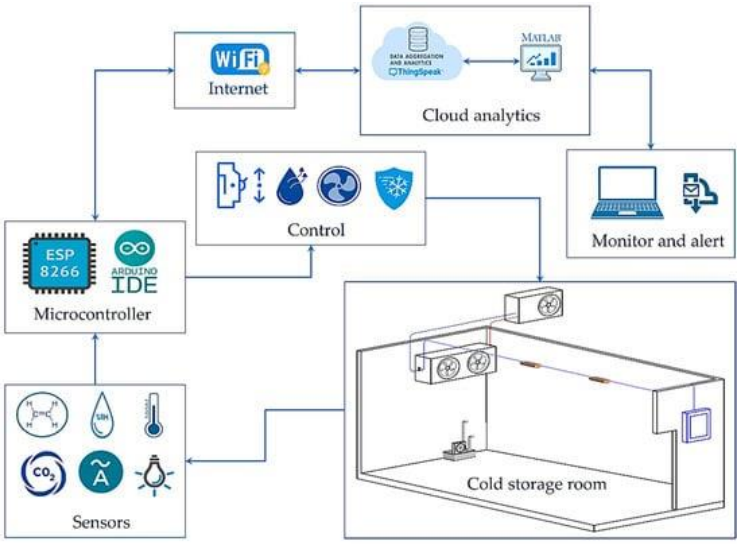

Fig. Circuit diagram of Smart Office Monitoring

Program

```
#include <WiFi.h>

#include <PubSubClient.h>

#include <DHT.h>

#include <NewPing.h>

#include <MQ135.h>


// WiFi and MQTT credentials

const char* ssid = "yourSSID";

const char* password = "yourPassword";

const char* mqtt_server = "mqtt_broker_address";  // Replace with MQTT broker IP or URL


WiFiClient espClient;

PubSubClient client(espClient);
```

```
// Pin Definitions
#define DHTPIN 4        // DHT22 sensor pin (GPIO 4)
#define PIRPIN 12       // PIR sensor pin (GPIO 12)
#define TRIGPIN 13      // Ultrasonic sensor trigger pin (GPIO 13)
#define ECHOPIN 14      // Ultrasonic sensor echo pin (GPIO 14)
#define RELAYPIN 15     // Relay pin to control HVAC (GPIO 15)
#define MQ135PIN 34     // MQ135 sensor analog pin (GPIO 34)

// Setup DHT sensor
DHT dht(DHTPIN, DHT22);

// Setup ultrasonic sensor (Water Tank Level)
NewPing sonar(TRIGPIN, ECHOPIN, 200);  // Max distance 200cm

// Setup MQ135 Air Quality Sensor
MQ135 gasSensor(MQ135PIN);

void setup() {
 Serial.begin(115200);

 // WiFi setup
 setup_wifi();

 // MQTT setup
 client.setServer(mqtt_server, 1883);
```

```
  // DHT sensor setup
  dht.begin();

  // PIR sensor setup
  pinMode(PIRPIN, INPUT);

  // Relay setup for HVAC control
  pinMode(RELAYPIN, OUTPUT);
  digitalWrite(RELAYPIN, LOW);  // Ensure HVAC is off initially

  // Initialize the gas sensor
  gasSensor.begin();
}

void setup_wifi() {
  delay(10);
  Serial.println();
  Serial.print("Connecting to WiFi...");
  WiFi.begin(ssid, password);

  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }

  Serial.println("WiFi connected");
```

```
}

void reconnect() {
  while (!client.connected()) {
    Serial.print("Attempting MQTT connection...");

    if (client.connect("SmartOfficeClient")) {
      Serial.println("connected");
      // Subscribe to topics if needed
    } else {
      Serial.print("Failed, rc=");
      Serial.print(client.state());
      delay(5000);
    }
  }
}

void loop() {
  if (!client.connected()) {
    reconnect();
  }
  client.loop();

  // Collect data from sensors
  float temperature = dht.readTemperature();
  float humidity = dht.readHumidity();
```

```
int occupancy = digitalRead(PIRPIN);
int waterLevel = sonar.ping_cm();  // Water level in cm
float airQuality = gasSensor.getPPM();  // Air quality (CO2, VOCs, etc.)

// If any sensor reading fails, return early
if (isnan(temperature) || isnan(humidity)) {
  Serial.println("Failed to read from DHT sensor!");
  return;
}

if (occupancy == HIGH) {
  Serial.println("Occupancy detected.");
  // Turn on HVAC if occupancy detected and temp > 25°C (you can customize this logic)
  if (temperature > 25) {
    digitalWrite(RELAYPIN, HIGH);  // HVAC ON
  }
} else {
  Serial.println("No occupancy detected.");
  // Turn off HVAC if no occupancy
  digitalWrite(RELAYPIN, LOW);   // HVAC OFF
}

// Prepare data payload
String payload = "{\"temperature\": " + String(temperature) +
                 ", \"humidity\": " + String(humidity) +
                 ", \"occupancy\": " + String(occupancy) +
```

```
          ", \"waterLevel\": " + String(waterLevel) +
          ", \"airQuality\": " + String(airQuality) + "}";

 // Publish sensor data to MQTT broker
 client.publish("office/sensors", payload.c_str());

 // Output to Serial Monitor for debugging
 Serial.print("Temperature: ");
 Serial.print(temperature);
 Serial.print(" C, Humidity: ");
 Serial.print(humidity);
 Serial.print(" %, Occupancy: ");
 Serial.print(occupancy == HIGH ? "Detected" : "Not detected");
 Serial.print(", Water Level: ");
 Serial.print(waterLevel);
 Serial.print(" cm, Air Quality: ");
 Serial.println(airQuality);

 // Wait for 10 seconds before the next reading
 delay(10000);
}
```

Components use

A humidity sensor monitoring system is designed to measure and track the moisture level in the air. These systems are commonly used in various applications, including agriculture, HVAC (heating, ventilation, and air conditioning), and indoor climate control, to maintain optimal humidity levels.

Here are the key components and concepts involved in a humidity sensor monitoring system:

1. Humidity Sensors: These are the core components that detect humidity levels. There are different types of humidity sensors, including capacitive, resistive, and thermal conductivity sensors. Capacitive sensors measure changes in capacitance caused by humidity, while resistive sensors measure the change in resistance of a hygroscopic material.

2. Microcontroller: This is the brain of the system. It processes the data received from the humidity sensor and can control other devices based on the readings. Common microcontrollers used include Arduino, Raspberry Pi, or any other programmable logic controller.

3. Data Logging: The system can log humidity data over time, allowing users to track changes and trends. This can be done using onboard memory in the microcontroller or by sending data to a cloud service for remote monitoring.

4. Display Unit: To provide real-time feedback, a display unit such as an LCD or LED screen can be integrated into the system. This allows users to see the current humidity level at a glance.

5. Alerts and Automation: Many systems include features for sending alerts when humidity levels fall outside of a predefined range. This can be done through notifications on a smartphone app or through email. Additionally, the system can be programmed to automatically adjust HVAC settings or activate dehumidifiers/humidifiers based on the readings.

6. Power Supply: The system needs a reliable power source, which can be batteries, USB power, or a direct connection to the mains, depending on the design and application.

A temperature monitoring system is designed to measure and track temperature variations in various environments, such as homes, industrial settings, or laboratories. Here's a detailed breakdown of its components and how it works:

1. Temperature Sensors: The core component of any temperature monitoring system is the sensor. Common types include:

- Thermocouples: These are made of two different metals joined at one end, producing a voltage that corresponds to temperature.

- RTDs (Resistance Temperature Detectors): These sensors use the principle that the resistance of certain metals changes with temperature.

-Thermistors: These are temperature-sensitive resistors that change resistance with temperature changes, typically more sensitive than RTDs.

2. Microcontroller: This is the brain of the system. It processes the data from the temperature sensors. Popular choices include Arduino, Raspberry Pi, or other

programmable microcontrollers. It reads the sensor data, processes it, and can make decisions based on temperature thresholds.

3. Data Logging: The system can log temperature readings over time, which is crucial for tracking trends and analyzing data. This can be done using an SD card module or by sending data to a cloud service for storage and analysis.

4. Display Unit: A display, such as an LCD or LED screen, shows real-time temperature readings. This allows users to easily monitor the current temperature at a glance.

5. Alerts and Notifications: Many temperature monitoring systems include alert features that notify users when temperatures exceed or fall below predetermined thresholds. This can be done via email, SMS, or app notifications.

6. Power Supply: The system needs a reliable power source. It could be powered by batteries, USB, or a direct electrical connection, depending on the application.

7. Connectivity: Some advanced systems can connect to the internet (IoT) to allow remote monitoring and control. This is especially useful in applications like smart homes or industrial settings.

4. Process Steps

Break down the project into its key phases or steps:
- Step 1: We research in many industries office such as production as well as any other and find out problem in it and start work on it
- **Step 2**: Then we provide solution design on it and execute our model working.
- Step 3: We implement the code which give to access the project
- Step 4: Then we test our model and check whether it work or not and any other fault in it also think about future scope.

5. Results/Observations

- It results with the protection of inner environment of office
- It protect the water loss as well as other loss.
- It provide discipline in employees

6. Conclusion

From this project we concluded that the problems face in the environment of the our workplace(office) sometime it is not good for our health it is dangerous for the our immune system.

To overcome it we are going to conduct this project in this project we concluded the solution of the various problem by using modern technology and the smart sensors. We monitor all the activities which related with the office.

7. References

1. Iyer Priya Eswar Padaleeswaaran, IOT Based Humidity and Temperature Monitoring using Arduino Uno with an feasibility of Sound Decibel Meter, (Reg.No. 37120032) November 2019 to April 2020.
2. Piyush Mahale, Sarang Panse, Pratiksha Rajule IOT based water level monitoring system, Volume:04/Issue:03/March-2022 Impact Factor- 6.752
3. www.tinkercad.com
4. www.thinkboard.com
5. www.wikipedia.com