**Smart Egg Incubator**

****

Session: 2022 – 2026

**Submitted by:**

Muhammad Salman 2022-CS-138

Muhammad Taha Saleem 2022-CS-139

Muhammad Abubakar 2022-CS-125

Muhammad Abubakar Jami 2022-CS-132

**Submitted to:**

Sir Tehseen-ul-Hassan Shah

Department of Computer Science

**University of Engineering and Technology**

**Lahore Pakistan**

Contents

[Project Photo 3](#_Toc155353309)

[Description 3](#_Toc155353310)

[Methodology Used: AVR Module (ESP32) 4](#_Toc155353311)

[**Programming Language** 4](#_Toc155353312)

[Data Flow Diagram 6](#_Toc155353313)

[AVR Flow Diagram 6](#_Toc155353314)

[IOT Flow Diagram 7](#_Toc155353315)

[**`**  **FlowChart** 7](#_Toc155353316)

[**Circuit Diagram** 8](#_Toc155353317)

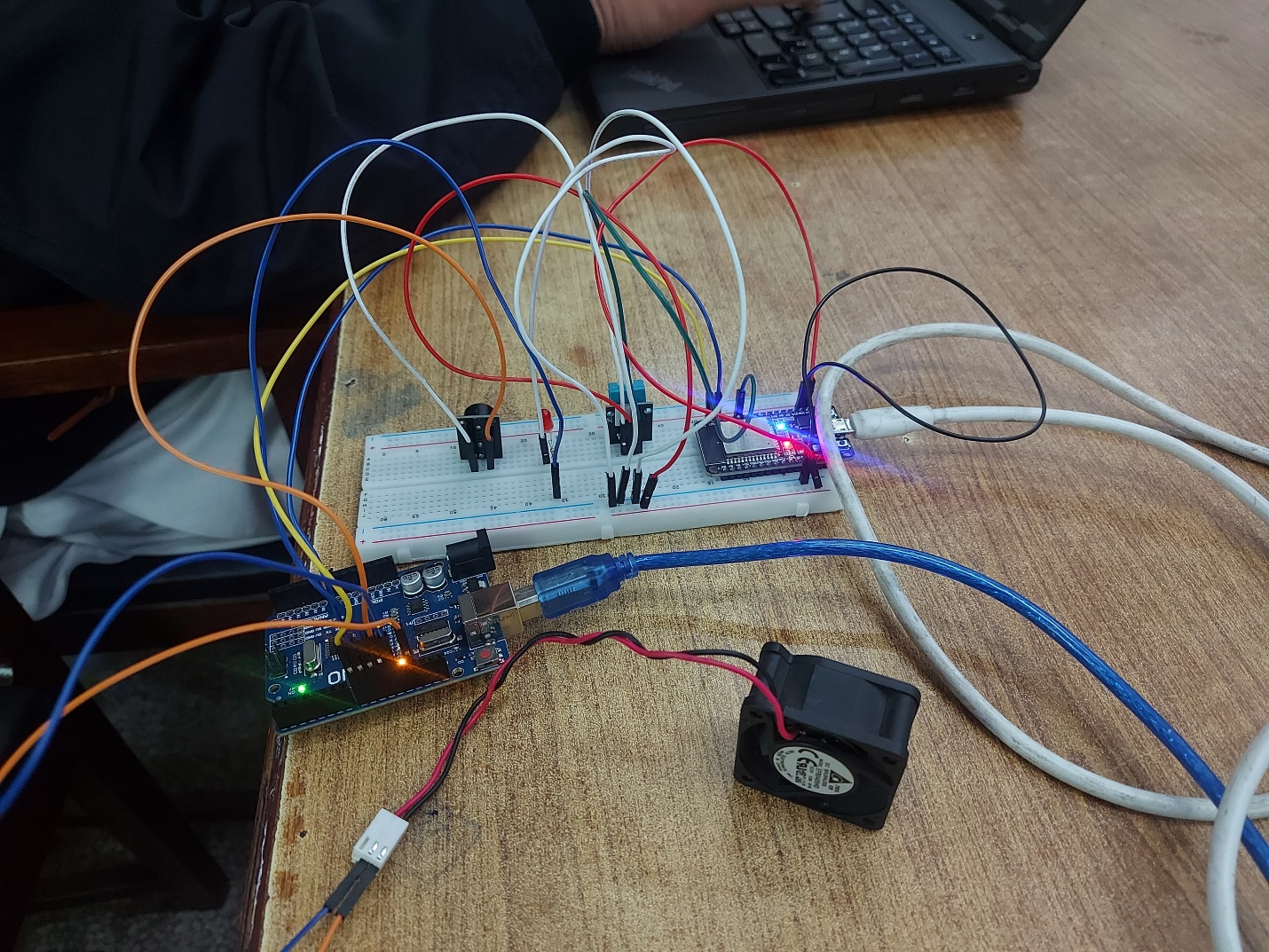
[AVR module code 11](#_Toc155353318)

[IoT module code 15](#_Toc155353319)

[MQTT Dashboard Screenshot 19](#_Toc155353320)

[References 20](#_Toc155353321)

# Project Photo



# Description

The Smart Egg Incubator project represents an innovative fusion of technology and agriculture, employing a microcontroller, such as Arduino or Raspberry Pi, along with a dedicated temperature and humidity sensor. This integrated system is designed to meticulously monitor and control the environmental conditions within the incubator, ensuring an optimal environment for successful egg incubation.

**Temperature and Humidity Sensing:**

At the core of our system is a highly accurate temperature and humidity sensor. This sensor continuously gathers real-time data, providing a constant stream of information critical for monitoring and managing the incubation environment. The precision of these measurements is fundamental to the success of the incubation process.

**Data Transmission Platforms**

1. **MQTT (Message Queuing Telemetry Transport):**

MQTT serves as a pivotal communication channel, enabling seamless and instantaneous data exchange. This protocol facilitates the transmission of real-time updates on temperature and humidity within the incubator. The immediate communication allows for prompt responses to any fluctuations, ensuring that the environment remains within the desired parameters for optimal egg development.

1. **ThingSpeak:**

Our project integrates ThingSpeak, a versatile platform that enhances data visualization and analysis. ThingSpeak transforms raw data into informative charts and graphs, providing a visual representation of temperature and humidity trends. This visual analysis is invaluable for users, offering insights into the performance of the incubator over time. ThingSpeak's user-friendly interface facilitates a deeper understanding of incubation conditions, contributing to informed decision-making.

**Comprehensive Approach to Smart Incubation:**

In summary, our integrated system embodies a holistic approach to smart egg incubation. The real-time monitoring and control, facilitated by MQTT, ensures immediate responsiveness to environmental changes. Simultaneously, ThingSpeak's visualization tools offer an accessible interface for users to comprehend trends and patterns, optimizing the overall effectiveness of our Smart Egg Incubator. This comprehensive solution maximizes the likelihood of a successful and controlled egg incubation process without the need for Firebase, providing a user-friendly and efficient system for egg hatchery management.

# Methodology Used: AVR Module (ESP32)

**Programming Language:** C/C++ in Arduino IDE

Our Smart Egg Incubator project leverages the capabilities of the AVR module, specifically the ESP32 microcontroller, to ensure precise monitoring and control of the incubation environment. The methodology encompasses the following key steps:

**Temperature and Humidity Sensing:**

* Integration of a dedicated temperature and humidity sensor (e.g., DHT11) with the AVR module (ESP32).
* Employing the ESP32's GPIO pins to interface with the sensor and retrieve real-time data on temperature and humidity within the incubator.

**Control Logic Implementation:**

* Developing control algorithms within the ESP32 to analyze the received temperature and humidity data.
* Defining threshold values for optimal incubation conditions.
* Implementing logic for activating the fan in case of temperature rise beyond the threshold, ensuring prompt cooling.
* Integrating a buzzer for audible alerts when environmental conditions deviate from the desired range.

**MQTT Integration:**

* Configuring the ESP32 to establish a secure and efficient connection with an MQTT broker.
* Implementing MQTT communication protocols to publish and subscribe to topics related to temperature and humidity.
* Transmitting real-time data updates to the MQTT broker for seamless communication with other connected devices or cloud platforms.

**ThingSpeak Integration:**

* Utilizing the ESP32's connectivity features to integrate with the ThingSpeak platform.
* Implementing data transfer protocols to upload temperature and humidity data to ThingSpeak's cloud.
* Configuring ThingSpeak channels to store and visualize data through charts and graphs, offering a user-friendly interface for data analysis.

**Power Management:**

* Implementing power-efficient strategies to prolong the life of the incubator system.
* Considering sleep modes for the ESP32 to conserve energy during idle periods, especially when the incubator is not actively monitored.

**Testing and Calibration:**

* Rigorous testing of the entire system to ensure the accuracy of temperature and humidity readings.
* Calibrating sensors and fine-tuning control algorithms to guarantee precise and reliable operation.

By adopting this methodology centered around the AVR module, specifically the ESP32 microcontroller, our Smart Egg Incubator achieves a robust and intelligent system for efficient egg incubation, combining real-time monitoring, control, and cloud connectivity for comprehensive data analysis.

AVR Module (Arduino UNO)

**Programming Language:** AVR Assembly in Microchip Studio

Communication

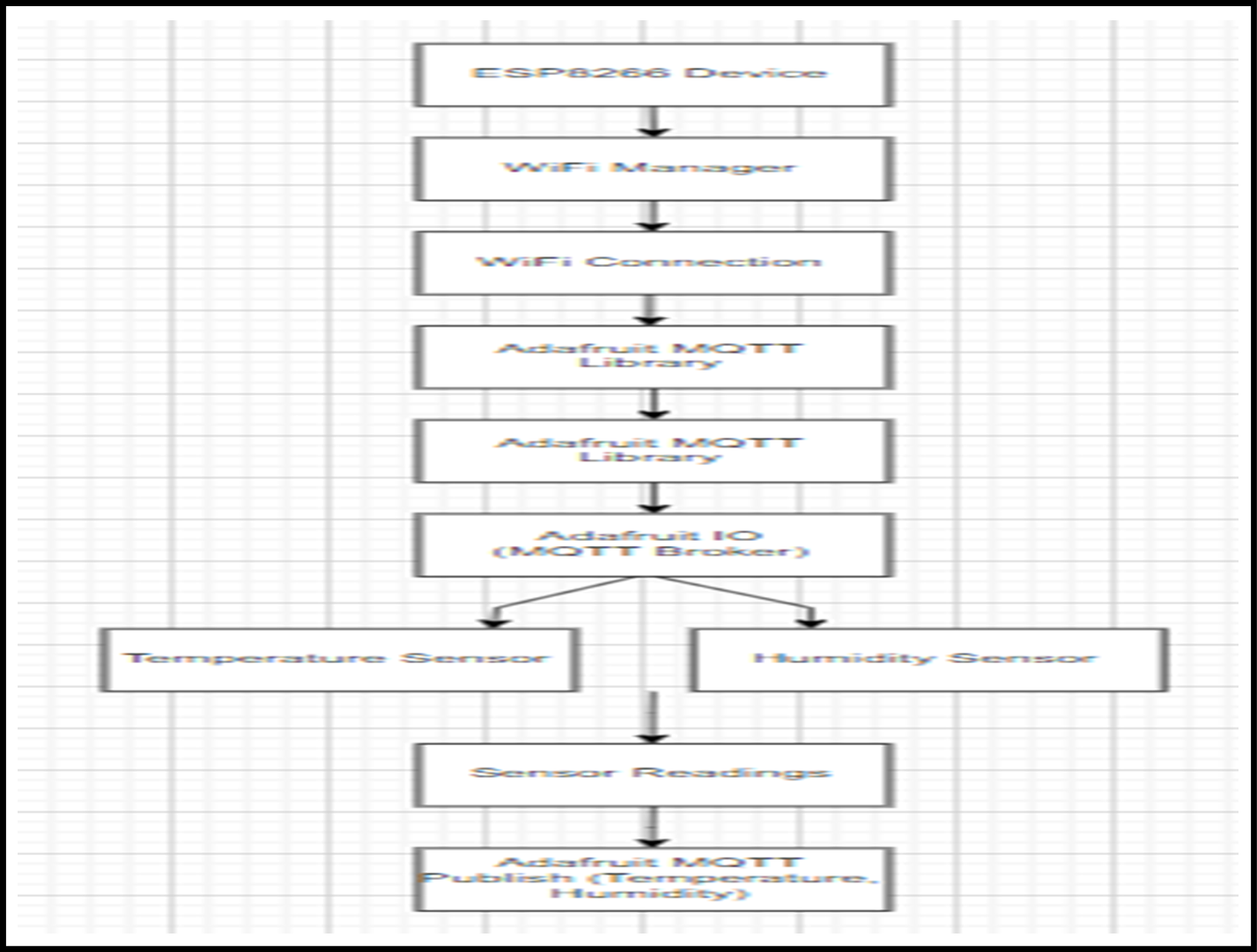
Data transfer between Arduino UNO and ESP32 is achieved using UART serial communication through Tx and Rx pins.

# Data Flow Diagram

# AVR Flow Diagram



# IOT Flow Diagram



## **` FlowChart**

Data Collection

Initialization

Start

Temperature Threshold?

No

Yes

Control Actions

Humidity Threshold?

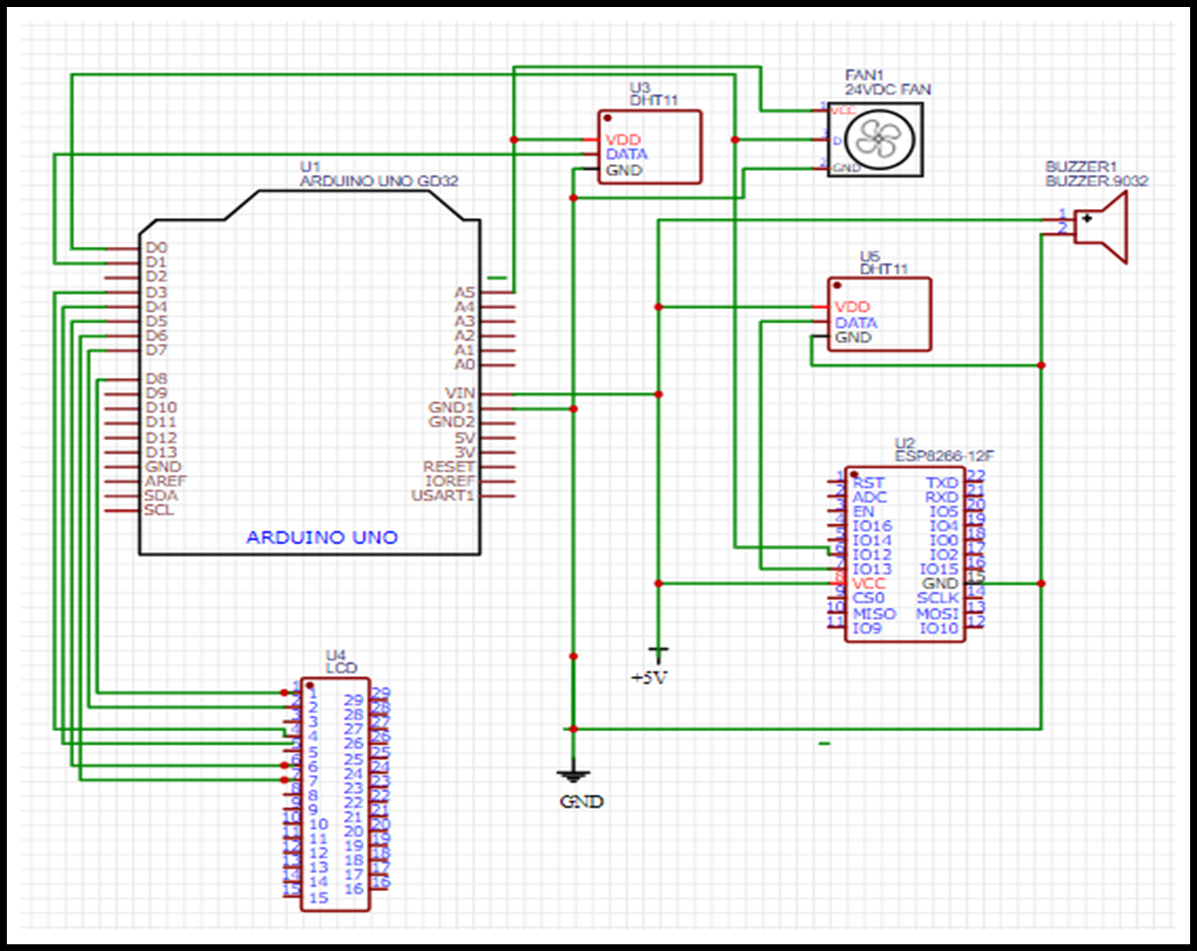
No

Yes

End

Remote Communication

## **Circuit Diagram**



Working of Components

Arduino UNO

• **Role:** The Arduino UNO serves as the microcontroller for the AVR module.

• Working:

* Monitors the temperature using a connected temperature sensor (DHT11).
* Controls an LED based on low temperature conditions for maintaining optimal warmth in the egg incubator.
* Activates an exhaust fan based on humidity values received from the ESP32.
* Drives a buzzer for audible notifications.

ESP32 Development Board

• **Role:** The ESP32 serves as the microcontroller for the IoT module.

• **Working:**

* Obtains DHT11 humidity and temperature values from the egg incubator.
* Sends realtime temperature, humidity, LED status, and fan status data to the ThingSpeak cloud for remote monitoring.
* Sends data to a smartphone via MQTT protocol for realtime monitoring.
* Allows manual control of the LED and exhaust fan through the MQTT Dash app.

5V Fan

• **Role:** The 5V fan is used for controlling the environmental conditions inside the egg incubator.

• Working:

* The fan is connected to the AVR module (Arduino UNO).
* The fan is activated or deactivated based on humidity levels received from the ESP8266.
* The fan helps regulate humidity by controlling airflow within the incubator.

DHT11 Sensor

• **Role:** The DHT11 sensor is used for measuring humidity and temperature.

• Working:

* Connected to both the AVR module (Arduino UNO) and the IoT module (ESP8266).
* In the AVR module, the DHT11 sensor measures the environmental temperature.
* In the IoT module, the DHT11 sensor measures humidity and temperature values.
* Data from the sensor is used for environmental control and monitoring.

Buzzer Module

• **Role:** The buzzer is used for providing audible notifications.

• Working:

* Connected to the AVR module (Arduino UNO).
* Emits two beeps when the exhaust fan turns on.
* Emits three beeps when the fan turns off.
* Provides audible feedback for specific system events.

Breadboard, Jumper Wires

• **Role:** Provide a platform for circuit connections and flexibility.

• Working:

* Components are connected on a breadboard using jumper wires.
* Enables easy prototyping and experimentation with different circuit configurations.

ThingSpeak Cloud

• **Role:** Online platform for storing and visualizing IoT data.

• Working:

* Data from the IoT module (ESP8266) is sent to ThingSpeak for real-time monitoring.
* Allows users to track temperature, humidity, LED status, and fan status remotely.

MQTT Dash App

• **Role:** Mobile application for manual control and monitoring.

• Working:

* Allows users to manually turn on/off the LED and exhaust fan.
* Provides a real-time display of temperature, humidity, LED status, and fan status.
* Utilizes the MQTT protocol for communication with the IoT module (ESP8266).

In summary, each component plays a specific role in monitoring and controlling the egg incubator environment. The system combines both the AVR and IoT modules to achieve automated control, remote monitoring, and manual intervention for maintaining optimal conditions for egg incubation.

# AVR module code

.include "m328pdef.inc"

.include "delay\_Macro.inc"

.include "UART\_Macros.inc"

.def temp = r20

.def humidity = r21

.cseg

.org 0x0000

; UART Configuration

SBI DDRD,1 ; Set PD1 (TX) as Output

CBI PORTD,1 ; TX Low (initial state)

CBI DDRD,0 ; Set PD0 (RX) as Input

SBI PORTD,0 ; Enable Pull-up Resistor on RX

Serial\_begin; Initialize UART Protocol

LDI temp,30

LDI humidity,70

LDI r31,2

SBI DDRB,3

CBI PORTB,3

SBI DDRB,4

CBI PORTB,4

Loop:

// read humidity

// call readTempratureValue

LDI temp, 25 // SUPPOSED VALUE

// Read humidity

// call readHumiditySensor

LDI humidity, 65 // SUPPOSED VALUE

CPI temp,20 // compare with a threshhold value

BRLO low\_temp\_detected

CPI temp, 25

BRSH high\_temp\_detected

// Normal temperature

rjmp normal\_temperature

low\_temp\_detected:

//call LED\_ON

//call Fanoff

CBI PORTB,4

// Beep the buzzer 3 times

LDI R31, 2

B3:

SBI PORTB,3

delay 400

CBI PORTB,3

delay 400

CPI r31, 0

DEC r31

BRGE B3

rjmp main\_exit

high\_temp\_detected:

// call LED\_OFF

// Fan On

SBI PORTB,4

// Beep the buzzer 2 times

LDI R31, 1

B2:

SBI PORTB,3

delay 400

CBI PORTB,3

delay 400

CPI r31, 0

DEC r31

BRGE B2

rjmp main\_exit

normal\_temperature:

//call LED\_OFF

//call FanOff

CBI PORTB,4

rjmp check\_humidity

check\_humidity:

cpi humidity, 70 // compare humidity with a threshold value

brsh high\_humidity\_detected

// Normal humidity

rjmp normal\_humidity

high\_humidity\_detected:

//call FanOn

CBI PORTB,4

// Beep the buzzer 2 times

LDI R31, 1

B1:

SBI PORTB,3

delay 400

CBI PORTB,3

delay 400

CPI r31, 0

DEC r31

BRGE B1

rjmp main\_exit

normal\_humidity:

//call FanOff

CBI PORTB,4

// Beep the buzzer 3 times

LDI R31, 2

B4:

SBI PORTB,3

delay 400

CBI PORTB,3

delay 400

CPI r31, 0

DEC r31

BRGE B4

main\_exit:

# IoT module code

#include <WiFi.h>

#include <PubSubClient.h>

#include <ESP32Firebase.h>

#include "ThingSpeak.h"

WiFiClient wifiClient;

PubSubClient client; // Pass the WiFi client to PubSubClient constructor

Firebase firebase("https://pelagic-cocoa-382522-default-rtdb.asia-southeast1.firebasedatabase.app/");

unsigned long myChannelNumber = 2;

const char \* myWriteAPIKey = "CO6KKYSWDKXSQW11";

const int LDRPin = 4;

const char \*ssid = "SAAD";

const char \*password = "atzn2722";

void setup() {

Serial.begin(9600);

pinMode(LDRPin, INPUT);

/\*WiFi.mode(WIFI\_STA);

WiFi.disconnect();

delay(1000);\*/

// Connect to WiFi

WiFi.begin(ssid, password);

while (WiFi.status() != WL\_CONNECTED) {

delay(1000);

Serial.println("Connecting to WiFi...");

}

Serial.println("Connected to WiFi");

// Connect to MQTT broker

//connectToMQTT();

ThingSpeak.begin(wifiClient);

}

void loop() {

// put your main code here, to run repeatedly:

/\*if (!client.connected()) {

// Reconnect to MQTT broker if connection is lost

connectToMQTT();

}\*/

int ldrValue = digitalRead(LDRPin);

Serial.print("LDR Value: ");

Serial.println(ldrValue);

// Convert ldrValue to char array

char buffer[10];

itoa(ldrValue, buffer, 10);

// Publish LDR value to MQTT

/\*if (client.publish("ashhad", buffer)) {

Serial.println("Message sent successfully");

} else {

Serial.println("Failed to send message");

}\*/

sendToTS(ldrValue);

firebase.pushInt("Intensity", ldrValue);

Serial.println("Value sent to Firebase");

delay(1000);

}

void sendToTS(int ldrValue) {

int x = ThingSpeak.writeField(myChannelNumber, 1, ldrValue, myWriteAPIKey);

if(x == 200)

{

Serial.println("Channel update successful.");

}

else

{

Serial.println("Problem updating channel. HTTP error code " + String(x));

}

}

void connectToMQTT() {

client.setClient(wifiClient);

client.setServer("test.mosquitto.org",1883);

while (!client.connected()) {

Serial.println("Attempting MQTT connection...");

if (client.connect("b14d6440-8a03-46b3-963a-d3384e54e6d8")) {

Serial.println("Connected to MQTT broker");

} else {

Serial.print("MQTT connection failed, rc=");

Serial.print(client.state());

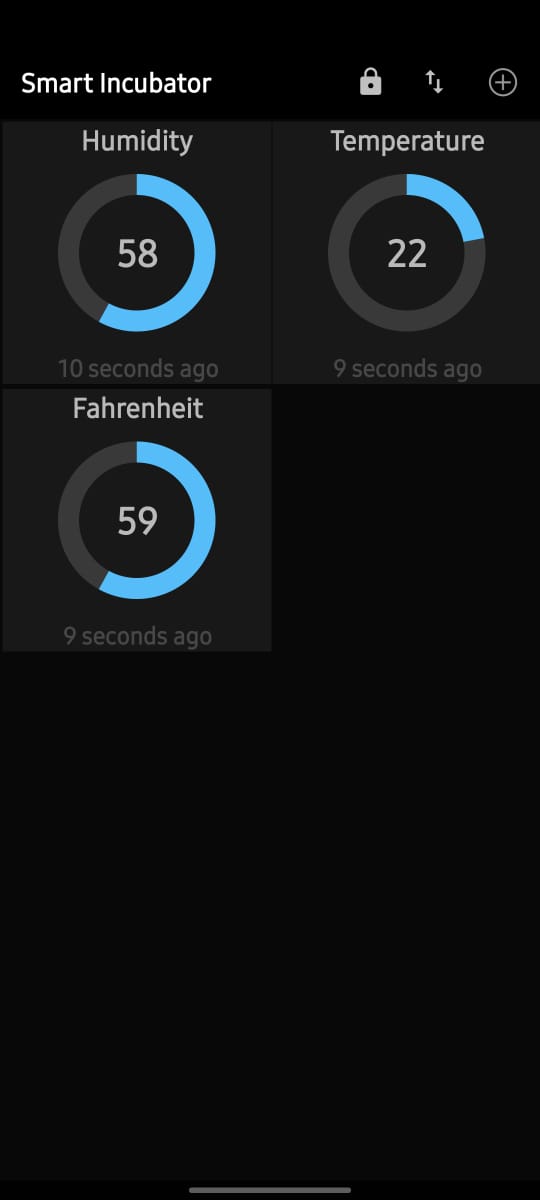
Serial.println(" Retrying in 1 second...");

delay(1000);

}

}

# MQTT Dashboard Screenshot



Dependencies

• **SimpleDHT library:** https://github.com/winlinvip/SimpleDHT

• **ESP8266WiFi library:** https://github.com/esp8266/Arduino

• **WiFiManager library:** https://github.com/tzapu/WiFiManager

• **Adafruit MQTT library**: https://github.com/adafruit/AdafruitM QTTLibrary

Hardware Connections

• DHT sensor connected to pin D3

• LED on pin D5 (LED PIN1)

Project Links

* **GitHub Repo:** <https://github.com/twonum/Smart-Egg-Incubator>

• **LinkedIn Video:** [linkedin.com/in/taha-saleem/](file:///C:\Users\Muhammad%20Taha%20Saleem\Desktop\linkedin.com\in\taha-saleem\)

• **YouTube Video:** <http://www.youtube.com/@outlawed_fantasy>

Conclusion

The Automated Egg Incubator Monitoring and Control System effectively ensures optimal conditions for egg incuba- tion. The integration of both AVR and IoT modules allows for automated control and remote monitoring, providing a comprehensive solution for egg incubation management.

# References

• Adafruit Learning System: <https://learn.adafruit.com/>

• ThingSpeak IoT Platform: <https://thingspeak.com/>

• MQTTDash: <https://play.google.com/store/apps/details?id=net.routix.mqttdashhl=engl=US>