



**REAL TIME SYSTEM AND INTERNET OF THINGS FINAL PROJECT REPORT
DEPARTMENT OF ELECTRICAL ENGINEERING
UNIVERSITAS INDONESIA**

CHICKCARE

GROUP 6

Audy Natalie Cecilia Rumahorbo	2306266962
Xavier Daniswara	2206030230
Christian Hadiwijaya	2306161952
Dwigina Sitti Zahwa	2306250724

PREFACE

Segala puji dan syukur ke Tuhan Yang Maha Esa atas segala rahmat dan karunia-nya sehingga Laporan Proyek Akhir dari Praktikum Internet of Things yang berjudul “ChickCare: An IoT-Based Intelligent Chick Incubator Monitoring and Control System” bisa diselesaikan dengan baik. Praktikkan ucapan terima kasih kepada Bang Giovan sebagai Pendamping Aslab dan teman-teman yang telah berkontribusi dalam penggerjaan Proyek Akhir Internet of Things.

Laporan ini disusun untuk melengkapi Praktikum IoT Tahun 2024. Laporan ini berisi penjelasan lengkap mengenai Proyek “ChickCare” berisikan proses perancangan, pengembangan, dan implementasi sistem untuk mengatasi tantangan tingginya angka kematian anak ayam pada fase *brooding* yang menawarkan pemantauan dan pengendalian lingkungan inkubator secara otomatis dan presisi.

Proyek ini juga mengintegrasikan Artificial Intelligence menggunakan **Google Gemini AI** yang dihubungkan melalui **Node-RED**. Fitur ini memungkinkan sistem memberikan rekomendasi *actionable insights* terhadap peternak berdasarkan data suhu dan kelembaban, oleh karena itu ChickCare tidak hanya alat monitoring, tetapi juga asisten cerdas bagi peternak.

Praktikkan menyadari kurangnya pengetahuan maupun pengalaman yang masih banyaknya keterbatasan dalam penggerjaan dan penyusunan laporan yang harus diperbaiki. Oleh karena itu, besar harapan praktikkan agar bisa mendapat kritik dan saran sehingga bisa dijadikan bahan evaluasi kedepannya. Praktikkan juga memohon maaf jika ada kesalahan dan kurangnya dalam penyusunan laporan.

Depok, December 01, 2025

Group 6

TABLE OF CONTENTS

CHAPTER 1.....	4
INTRODUCTION.....	4
1.1 PROBLEM STATEMENT.....	4
1.3 ACCEPTANCE CRITERIA.....	5
1.4 ROLES AND RESPONSIBILITIES.....	5
1.5 TIMELINE AND MILESTONES.....	6
CHAPTER 2.....	7
IMPLEMENTATION.....	7
2.1 HARDWARE DESIGN AND SCHEMATIC.....	7
2.2 SOFTWARE DEVELOPMENT.....	7
2.3 HARDWARE AND SOFTWARE INTEGRATION.....	16
Fig 2. Node-Red Status Dashboard.....	17
Fig 3. Node-Red Settings Dashboard.....	17
Fig 4. Node-Red History Dashboard.....	18
CHAPTER 3.....	18
TESTING AND EVALUATION.....	18
3.1 TESTING.....	18
3.2 RESULT.....	18
3.3 EVALUATION.....	25
CHAPTER 4.....	27
CONCLUSION.....	27

CHAPTER 1

INTRODUCTION

1.1 PROBLEM STATEMENT

Angka kematian anak ayam saat dua minggu pertama setelah menetas cukup tinggi, hal ini disebabkan oleh kondisi lingkungan yang tidak stabil. Proses inkubasi dan *brooding* tradisional membutuhkan termometer dan penyesuaian manual yang kurang efisien ataupun terjadinya *human error*, terutama mengenai suhu dan pencahayaan yang teratur.

Suhu yang diperlukan untuk anak ayam bersifat dinamis dan adaptif, harus tinggi pada minggu pertama (sekitar 32–35°C) dan secara bertahap diturunkan pada minggu kedua (sekitar 30–33°C). Kegagalan dalam mempertahankan adaptasi suhu ini bisa mengakibatkan stres termal yang menyebabkan penyakit. Selain itu, jadwal terang dan gelap yang teratur sangat penting untuk membantu anak ayam belajar makan, minum, dan beristirahat dengan teratur, tetapi saat ini masih secara sistem manual.

1.2 PROPOSED SOLUTION

ChickCare hadir sebagai solusi IoT untuk mengatasi masalah tersebut dengan ESP32 yang memungkinkan pemantauan, pengendalian, dan penyesuaian parameter lingkungan secara otomatis secara *real-time*. Pengaturan suhu diterapkan dengan menggunakan Pulse Width Modulation (PWM) pada Lampu Brooder. Pendekatan ini memungkinkan penyaluran daya pemanas yang sebanding dan bisa disesuaikan dengan kebutuhan suhu saat ini.

Sistem pada ChickCare juga terhubung dengan Node-Red, yaitu *tools* pemrograman untuk visualisasi yang berbasis aliran. Node-Red biasanya digunakan untuk mengumpulkan, mentransformasi, dan mengintegrasikan data melalui *dashboard* visual. Penggunaan Node-Red ini terintegrasi dengan ESP32, API, ataupun layanan, pada ChickCare digunakan dengan menerapkan AI (Artificial Intelligence) dalam memberikan rekomendasi *actionable insights* terhadap peternak berdasarkan data suhu dan kelembaban.

Oleh karena itu, sistem ChickCare dikembangkan untuk menciptakan lingkungan inkubasi yang tepat dan bisa diakses dari jauh (*remotely accessible*), sehingga mengurangi keterlibatan manual dengan meningkatkan kelangsungan hidup anak ayam.

1.3 ACCEPTANCE CRITERIA

Tujuan Proyek ChickCare antara lain:

1. Menciptakan sistem yang secara otomatis menyesuaikan target suhu ideal sesuai tahap pertumbuhan dalam 2 minggu pertama dan mempertahankannya menggunakan PWM pada Lampu Brooder untuk keakuratan yang maksimal.
2. Mengimplementasikan logika kontrol yang mengatur level kelembaban melalui Dehumidifier Spray dan menggabungkan penjadwalan terang dan gelap yang terorganisir (Jadwal Cahaya).
3. Menyediakan indikator visual (LED) dan monitor data (Serial Output) untuk melaporkan status sistem, nilai PWM dinamis, dan memberikan *alert* saat terjadi kondisi kritis.
4. Mengintegrasikan dengan Node-RED dengan terhubung dari data sensor ke Gemini AI. Fitur ini menganalisis kondisi suhu dan kelembaban dan rekomendasi secara langsung kepada peternak.

1.4 ROLES AND RESPONSIBILITIES

Pembagian tanggung jawab dan peran dalam penggerjaan proyek, antara lain:

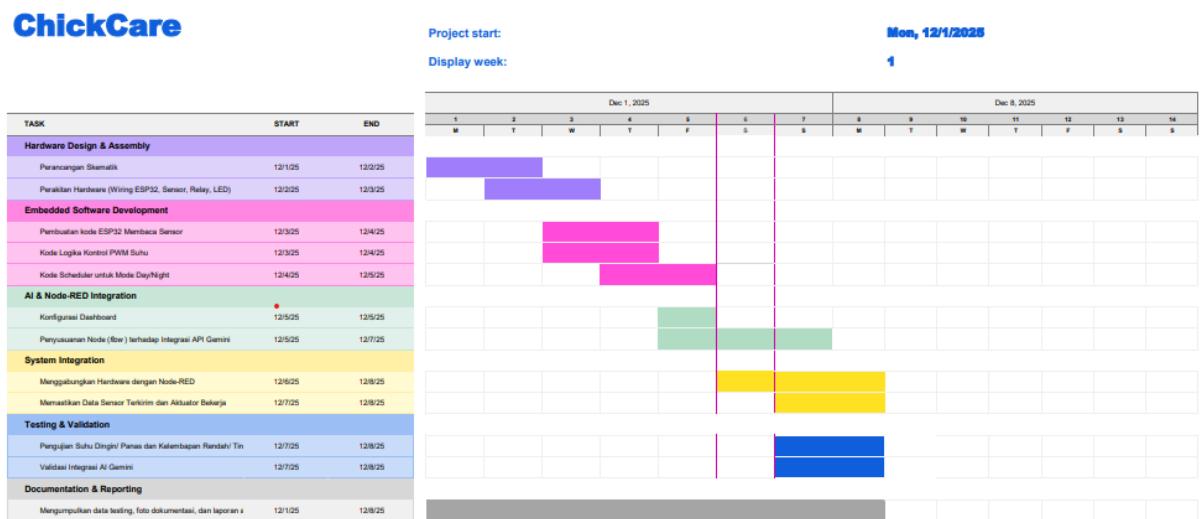
Roles	Responsibilities	Person
Physical Hardware	Bertanggung jawab atas perakitan komponen fisik dan pengujian konektivitas elektrik pada alat.	Audy Natalie Cecilia Rumahorbo
Physical Hardware	Bertanggung jawab atas perakitan komponen fisik dan pengujian konektivitas elektrik pada alat.	Xavier Daniswara

AI & IoT Integrator	Bertanggung jawab atas pengembangan alur logika pada Node-RED, melakukan integrasi dengan AI untuk fitur rekomendasi cerdas, dan memastikan komunikasi data antara ESP32.	Christian Hadiwijaya
Digital Hardware Designer & Reporter	Bertanggung jawab dalam perancangan desain rangkaian digital (simulasi Wokwi) dan penyusunan laporan akhir proyek secara sistematis.	Dwigina Sitti Zahwa

Table 1. Roles and Responsibilities

1.5 TIMELINE AND MILESTONES

The Gantt Chart:



CHAPTER 2

IMPLEMENTATION

2.1 HARDWARE DESIGN AND SCHEMATIC

Perancangan *hardware* ChickCare dengan **ESP32 Dev Module** sebagai mikrokontroler utama yang bertugas memproses logika kontrol dan manajemen waktu. Sistem ini menerima input lingkungan melalui **Sensor DHT11** yang mengukur suhu dan kelembaban secara *real-time*. Penggunaan aktuator dalam sistem menggunakan **Relay Module** yang terhubung ke **Brooder Lamp** sebagai pemanas utama dan **Dehumidifier Spray** untuk mengontrol kelembaban. Selain itu, dua **LED** sebagai indikator visual untuk status pemanas dan *spray*. Seluruh komponen ini dirangkai pada breadboard menggunakan kabel jumper dan resistor menjadi sistem terintegrasi dalam menjalankan logika kontrol *closed-loop* secara mandiri.

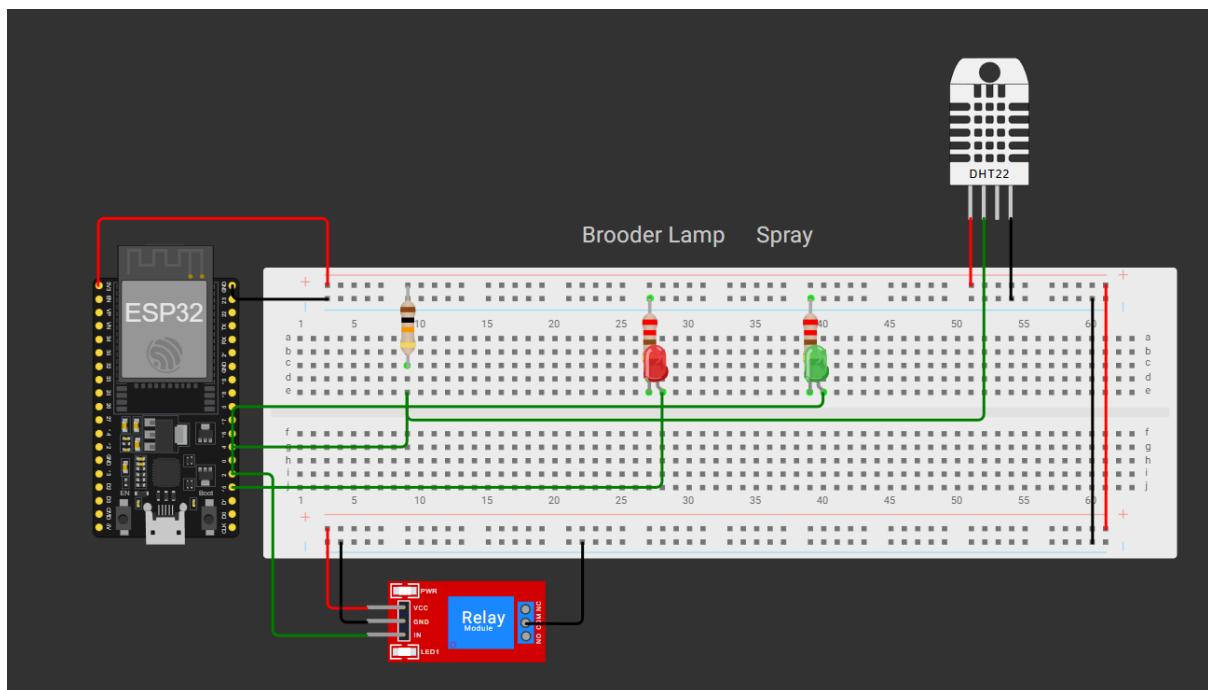


Fig 1. Hardware Design

2.2 SOFTWARE DEVELOPMENT

Pengembangan *software* ini menggunakan **FreeRTOS** pada ESP32 untuk memastikan *multitasking* yang efisien antar pembacaan sensor, kontrol aktuator, dan manajemen waktu yang dibuat dalam cpp pada Arduino IDE. Kontrol suhu diimplementasikan melalui PWM Brooder Lamp (HEATER_LED_PIN) dengan fungsi controlHeaterLED() yang menyesuaikan daya berdasarkan jarak antara suhu aktual (currentTemp) dan target suhu (32.0°C – 33.0°C), tetapi kontrol pemanas akan dimatikan sesuai controlTask.

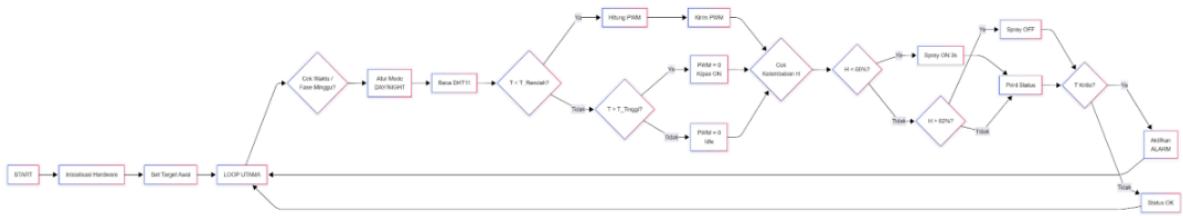


Fig 2. Flowchart

Sistem chickcare juga didukung dengan pemanfaatan Generative AI yaitu **gemini2.5-flash-lite** sebagai chatbot interaktif yang dapat memberikan analisis, saran serta ringkasan kondisi inkubator saat ini. Model AI dipanggil melalui Gemini API di node-red dan diberikan informasi melalui RAG dan Prompt Engineering.

A. Firmware Implementation

Implementasi kode yang digunakan pada mikrokontroler ESP32, terkait logika PWM, Histeresis, dan FreeRTOS Tasks, yaitu:

```
#include <DHT.h>

#include <freertos/FreeRTOS.h>

#include <freertos/task.h>

#include <freertos/semphr.h>

// Pin definitions

#define DHT_PIN 4

#define DHT_TYPE DHT11

#define HEATER_LED_PIN 15

#define SPRAY_LED_PIN 2 // LED/Relay untuk Dehumidifier Spray

// Temperature settings
```

```

#define TARGET_TEMP_MIN 32.0

#define TARGET_TEMP_MAX 33.0

#define TEMP_HYSTERESIS 0.5

// Humidity settings (pakai histeresis)

#define HUMIDITY_ON_THRESHOLD 60.0 // Spray nyala kalau < 40%

#define HUMIDITY_OFF_THRESHOLD 62.0 // Spray mati kalau > 45%

#define SPRAY_DURATION_MS 3000 // Spray aktif selama 3 detik

// Time settings (demo mode: 1 jam = 1 detik)

#define DAY_HOURS 18

#define NIGHT_HOURS 6

// System variables

DHT dht(DHT_PIN, DHT_TYPE);

float currentTemp = 0;

float currentHumidity = 0;

bool isDayTime = true;

unsigned long systemStartTime;

int pwmValue = 0;

bool heaterEnabled = true;

// Dehumidifier variables

bool sprayActive = false;

unsigned long sprayStartTime = 0;

// FreeRTOS handles

TaskHandle_t tempTaskHandle;

TaskHandle_t timeTaskHandle;

TaskHandle_t controlTaskHandle;

SemaphoreHandle_t tempMutex;

void setup() {

Serial.begin(115200);

pinMode(HEATER_LED_PIN, OUTPUT);

```

```

pinMode(SPRAY_LED_PIN, OUTPUT);

dht.begin();

systemStartTime = millis();

tempMutex = xSemaphoreCreateMutex();

Serial.println("== ChickCare System Initialized ==");

xTaskCreatePinnedToCore(temperatureTask, "TempTask", 4096, NULL, 1, &tempTaskHandle,
0);

xTaskCreatePinnedToCore(timeManagementTask, "TimeTask", 2048, NULL, 1,
&timeTaskHandle, 0);

xTaskCreatePinnedToCore(controlTask, "ControlTask", 4096, NULL, 2,
&controlTaskHandle, 1);

}

void loop() {

displaySystemStatus();

vTaskDelay(1000 / portTICK_PERIOD_MS);

}

// === TASK: Membaca suhu & kelembapan ===

void temperatureTask(void *parameter) {

while (1) {

float newTemp = dht.readTemperature();

float newHumidity = dht.readHumidity();

if (!isnan(newTemp) && !isnan(newHumidity)) {

if (xSemaphoreTake(tempMutex, portMAX_DELAY) == pdTRUE) {

currentTemp = newTemp;

currentHumidity = newHumidity;

xSemaphoreGive(tempMutex);

}

}

vTaskDelay(2000 / portTICK_PERIOD_MS);

```

```

    }

}

// === TASK: Mengatur pergantian mode siang/malam ===

void timeManagementTask(void *parameter) {

    while (1) {

        unsigned long elapsedSeconds = (millis() - systemStartTime) / 1000;

        unsigned long cycleTime = elapsedSeconds % (DAY_HOURS + NIGHT_HOURS);

        bool newIsDay = (cycleTime < DAY_HOURS);

        if (newIsDay != isDayTime) {

            isDayTime = newIsDay;

            Serial.print("Mode changed to: ");

            Serial.println(isDayTime ? "DAY" : "NIGHT");

        }

        vTaskDelay(1000 / portTICK_PERIOD_MS);

    }

}

// === TASK: Mengontrol heater dan spray ===

void controlTask(void *parameter) {

    while (1) {

        float temp, hum;

        if (xSemaphoreTake(tempMutex, portMAX_DELAY) == pdTRUE) {

            temp = currentTemp;

            hum = currentHumidity;

            xSemaphoreGive(tempMutex);

        }

        // Kontrol heater hanya aktif saat siang

        if (heaterEnabled && isDayTime) {

            controlHeaterLED(temp);

        } else {

```

```

analogWrite(HEATER_LED_PIN, 0);

pwmValue = 0;

}

// Kontrol spray dehumidifier

controlSpray(hum);

vTaskDelay(500 / portTICK_PERIOD_MS);

}

// === FUNGSI: Kontrol Heater dengan PWM ===

void controlHeaterLED(float temperature) {

int newPwmValue;

if (temperature < TARGET_TEMP_MIN - TEMP_HYSTESIS) newPwmValue = 255;

else if (temperature > TARGET_TEMP_MAX + TEMP_HYSTESIS) newPwmValue = 0;

else if (temperature < TARGET_TEMP_MIN) newPwmValue = 200;

else if (temperature > TARGET_TEMP_MAX) newPwmValue = 50;

else newPwmValue = 128;

analogWrite(HEATER_LED_PIN, newPwmValue);

pwmValue = newPwmValue;

}

// === FUNGSI: Kontrol Dehumidifier Spray (non-blocking + histeresis) ===

void controlSpray(float humidity) {

unsigned long now = millis();

// Nyalakan spray kalau kelembapan rendah

if (!sprayActive && humidity < HUMIDITY_ON_THRESHOLD) {

sprayActive = true;

sprayStartTime = now;

digitalWrite(SPRAY_LED_PIN, HIGH);

Serial.println(">>> Dehumidifier Spray ON <<<");

}

```

```

    }

    // Matikan spray setelah durasi tertentu atau jika kelembapan sudah cukup tinggi

    if (sprayActive && ((now - sprayStartTime > SPRAY_DURATION_MS) || humidity >
HUMIDITY_OFF_THRESHOLD)) {

        digitalWrite(SPRAY_LED_PIN, LOW);

        sprayActive = false;

        Serial.println(">>> Dehumidifier Spray OFF <<<");

    }

}

// === FUNGSI: Menampilkan status sistem di Serial ===

void displaySystemStatus() {

    float temp, hum;

    if (xSemaphoreTake(tempMutex, portMAX_DELAY) == pdTRUE) {

        temp = currentTemp;

        hum = currentHumidity;

        xSemaphoreGive(tempMutex);

    }

    Serial.print(" | Mode: ");

    Serial.print(isDayTime ? "DAY" : "NIGHT");

    Serial.print(" | Temp: ");

    Serial.print(temp);

    Serial.print("°C | Humidity: ");

    Serial.print(hum);

    Serial.print("% | Heater PWM: ");

    Serial.print(pwmValue);

    Serial.print(" | Spray: ");

    Serial.println(sprayActive ? "ON" : "OFF");

}

```

a. Struktur Program dan Multi-Tasking

Task / Fungsi	Peran
temperatureTask	Input: Membaca data suhu (T) dan kelembaban (H) dari DHT11 setiap 2 detik. Menggunakan <i>tempMutex</i> untuk memastikan pembacaan aman antar task.
controlTask	Kontrol inti: Menjalankan logika PWM untuk pemanas serta logika Spray dehumidifier setiap 500 ms.

b. Brooder Lamp melalui PWM

Task / Fungsi	Nilai PWM	Analisis
< TARGET_TEMP_MIN - TEMP_HYSTERESIS	255 (Maksimum)	Suhu jauh di bawah target → daya pemanas penuh.
TARGET_TEMP_MIN - TEMP_HYSTERESIS ≤ T < TARGET_TEMP_MIN	200 (Tinggi)	Suhu di bawah target minimal → daya tinggi.
TARGET_TEMP_MIN ≤ T ≤ TARGET_TEMP_MAX	128 (Sedang)	Suhu optimal → daya sedang menjaga stabilitas.
TARGET_TEMP_MAX < T ≤ TARGET_TEMP_MAX + TEMP_HYSTERESIS S (mis. 33.0°C < T ≤ 33.5°C)	50 (Rendah)	Suhu di atas target maksimal → daya rendah/pendinginan aktif.
T > TARGET_TEMP_MAX + TEMP_HYSTERESIS (mis. T > 33.5°C)	0 (OFF)	Suhu jauh di atas target → pemanas mati.

c. Manajemen Kelembapan (Spray)

Kontrol kelembaban melalui fungsi controlSpray() dengan sesuai threshold control yang *non-blocking* dan durasi aktivasi singkat.

- Threshold: Kelembaban dipertahankan di atas 60.0% (HUMIDITY_ON_THRESHOLD) dan dimatikan saat 62.0% (HUMIDITY_OFF_THRESHOLD).

- Logika durasi: Jika $H < 60\%$, maka Spray diaktifkan selama durasi singkat (`SPRAY_DURATION_MS = 3000 ms`) menggunakan fungsi `millis()` agar task lain tetap berjalan selama Spray aktif (*non-blocking*).

d. Scheduler Cahaya (Manajemen Waktu DAY/NIGHT)

Sistem menggunakan `timeManagementTask` untuk mengelola jadwal terang–gelap.

Mode	Durasi (Mode Demo)	Fungsi Utama
DAY	18 jam (dalam detik)	Brooder Lamp (pemanas) diizinkan aktif dan PWM beroperasi.
NIGHT	6 jam (dalam detik)	Brooder Lamp (pemanas) dinonaktifkan sepenuhnya (PWM = 0).

B. Node-RED dan Gemini AI Integration

Menghubungkan ESP32 ke Node-RED melalui komunikasi Serial/MQTT.....

Integrasi sistem IoT dengan AI dilakukan menggunakan Node-RED sebagai *middleware* yang menghubungkan data sensor dari ESP32 ke Google Gemini AI. Komunikasi data menggunakan protokol MQTT dengan format JSON. Berbeda dengan logika kontrol *real-time* pada ESP32, integrasi AI menerapkan konsep *Retrieval-Augmented Generation* (RAG) sederhana menggunakan penyimpanan sementara (*buffer*) pada memori Node-RED.

a. Mekanisme Alur Data (Data Flow Architecture)

Node-RED menerima data suhu (`currentTemp`) dan kelembapan (`currentHumidity`) secara terus-menerus. Sebuah Function Node menyimpan data tersebut ke dalam variabel global (flow context) bernama `chickHistory`. Sistem mempertahankan 100 entri data terakhir menggunakan metode First-In-First-Out (FIFO) sebagai history.

Analisis AI hanya dipicu saat pengguna mengakses Dashboard (Page View

Event). Sistem menggabungkan data aktual (current reading) dengan 100 data historis (historical logs) menjadi satu prompt terstruktur sebelum dikirim ke API.

b. Struktur Prompt dan Logika Analisis

Agar respons AI relevan dengan standar keselamatan peternakan, "System Instruction" dirancang dengan parameter spesifik seperti pada tabel berikut:

<p># PERAN & IDENTITAS</p> <p>Kamu adalah "ChickCare Assistant", kecerdasan buatan yang bertugas menjaga inkubator ayam pintar. Tugas utamamu adalah memantau kesehatan anak ayam melalui data lingkungan dan memastikan perangkat keras (pemanas/kipas) berfungsi dengan baik.</p>
<p># STANDAR KESELAMATAN (Acuan Dasar)</p> <ul style="list-style-type: none">- Suhu Ideal Brooding (Minggu awal): 30°C - 35°C.- Suhu Berbahaya: Di bawah 28°C (hipotermia/kedinginan) atau di atas 37°C (heat stress/kepanasan).- Kelembapan Ideal: 50% - 70%.- Jika data menunjukkan angka ekstrem, berikan peringatan dengan nada urgensi.
<p># GAYA KOMUNIKASI</p> <ul style="list-style-type: none">- Gunakan Bahasa Indonesia yang natural, ramah, namun profesional.- Bertindaklah seperti peternak ahli yang mengerti data teknis.- Jangan hanya menyebutkan angka, tapi jelaskan artinya. <p>Contoh: "Suhu turun ke 25°C, ini terlalu dingin."</p>
<p># INSTRUKSI INTERAKSI PERTAMA (PENTING)</p> <p>Saya melampirkan data sensor di bawah ini. JANGAN menunggu saya bertanya. Kamu HARUS langsung memberikan **"Analisis Ringkasan Kondisi"** berdasarkan data tersebut. Gunakan struktur:</p> <ol style="list-style-type: none">1. **Status Utama**: (Contoh: "Kondisi Aman Terkendali" atau "PERINGATAN")2. **Analisis Tren**: Ceritakan apa yang terjadi dalam data historis (stabil/naik/turun).3. **Rekomendasi**: Apa yang harus dilakukan pengguna sekarang?

2.3 HARDWARE AND SOFTWARE INTEGRATION

Integrasi dilakukan dengan memastikan Task FreeRTOS berjalan tanpa saling memblokir (*non-blocking*). Logika spray menggunakan `millis()` agar sensor tetap bisa membaca suhu saat spray menyala . Integrasi antara AI dengan Perangkat Fisik dilakukan menggunakan Node-RED untuk memberikan dashboard bagi admin untuk bisa melihat status, konfigurasi hardware, serta melihat history alert.



Fig 2. Node-Red Status Dashboard

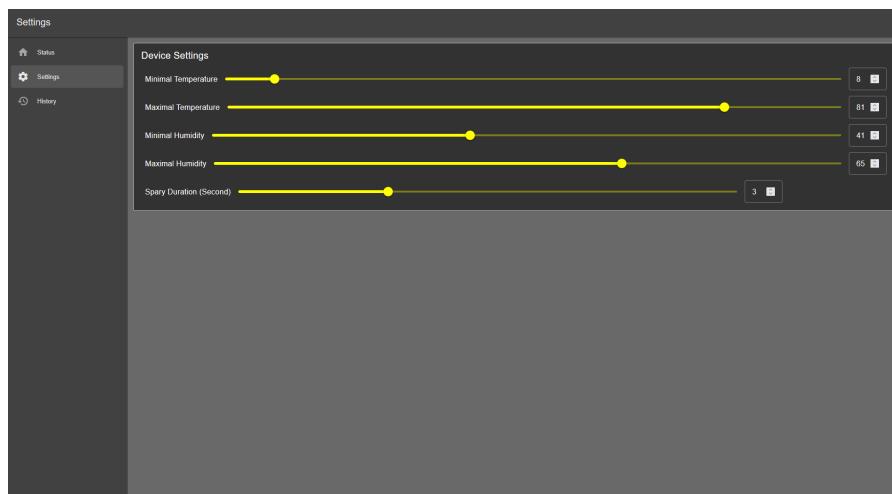


Fig 3. Node-Red Settings Dashboard

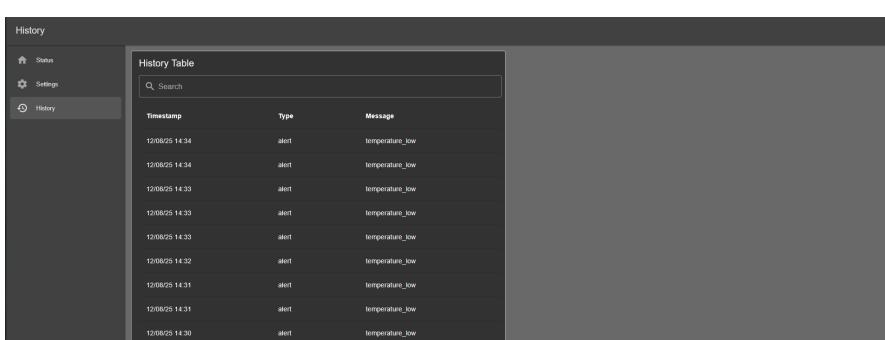


Fig 4. Node-Red History Dashboard

CHAPTER 3

TESTING AND EVALUATION

3.1 TESTING

Pada saat pengujian, praktikkan mensimulasikan durasi satu jam disimulasikan sebagai satu detik untuk mempermudah proses pengujian. Selain itu, *dehumidifier spray* untuk pengujian rangkaian fisik, *humidity threshold* diatur sekitar **60–65%**, menyesuaikan dengan kondisi kelembapan terendah di lingkungan pengujian yang mencapai 58%. Hal ini memungkinkan praktikkan untuk mengamati transisi siklus **DAY (18 jam)** dan **NIGHT (6 jam)**, serta respons aktuator terhadap perubahan mode.

Pengujian fungsional difokuskan dengan memastikan pembacaan suhu dan kelembaban berjalan stabil tanpa nilai *null*, serta menguji tidak terjadinya *data race* saat sensor dibaca oleh *temperatureTask*. Memverifikasi perubahan intensitas cahaya pada Brooder Lamp dengan memanipulasi suhu di sekitar sensor (melalui Wokwi) untuk melihat apakah nilai PWM berubah sesuai logika **Maksimum (255)** saat suhu dingin ekstrem, **Sedang (128)** saat suhu target tercapai, dan **Mati (0)** saat suhu berlebih atau masuk mode malam. Menguji fitur *non-blocking* pada *spray* menggunakan fungsi *millis()*. Sistem diuji untuk memastikan *spray* hanya menyala selama 3000 ms (3 detik) saat kelembaban turun di bawah *threshold*, dan tidak menghentikan proses *monitoring* suhu saat menyala.

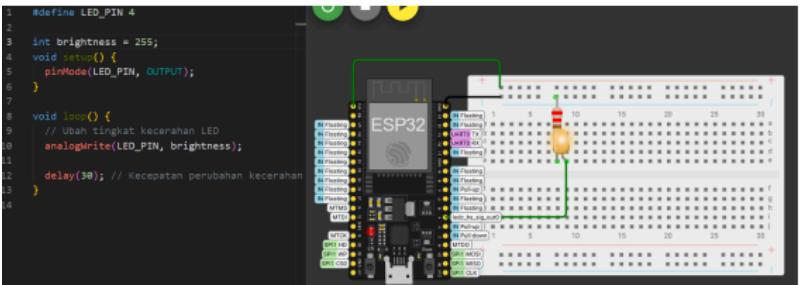
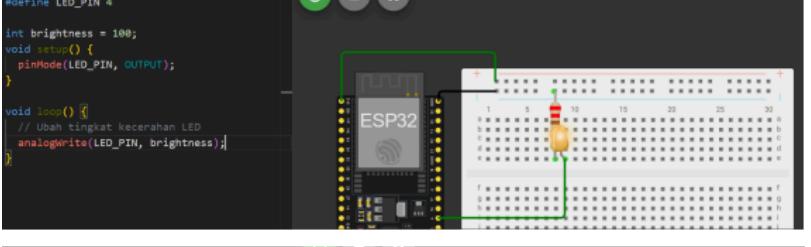
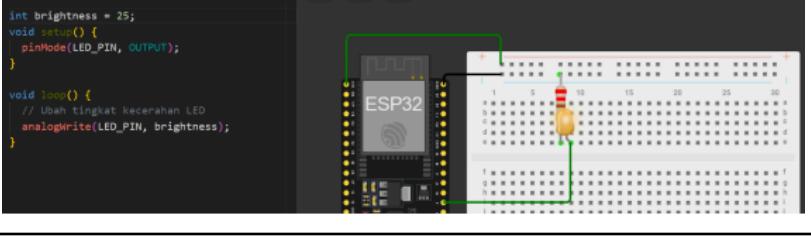
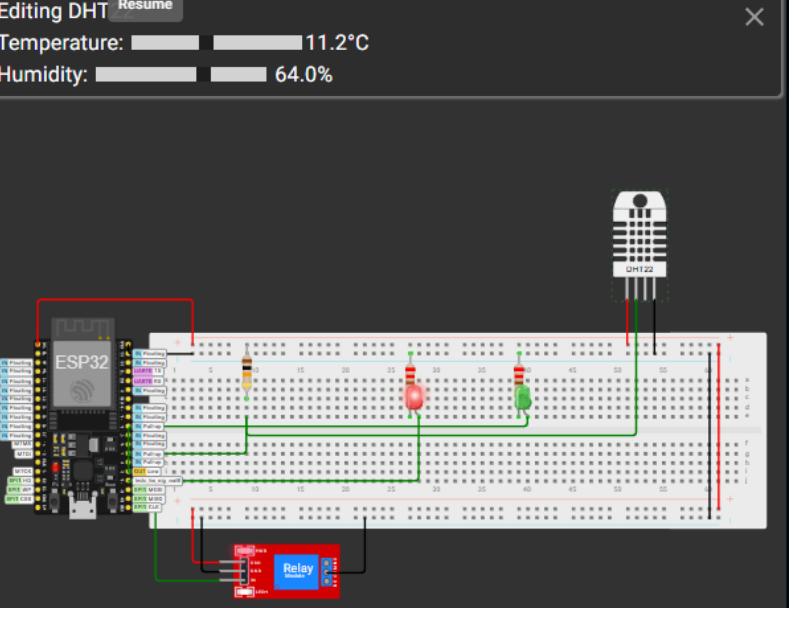
Pengujian integrasi Gemini API dilakukan ketika page status baru dibuka. Text Chatbot seharusnya memberikan chat pembuka berisi ringkasan serta rekomendasi aksi yang dilakukan saat ini. Setelah chat pembuka, chatbot dapat berinteraksi dengan user, memberikan data-data relevan sehingga mempermudah user untuk tidak perlu mengolah data sendiri. Teks harus terlihat jelas dengan delay < **5 detik** untuk setiap bubble chat.

3.2 RESULT

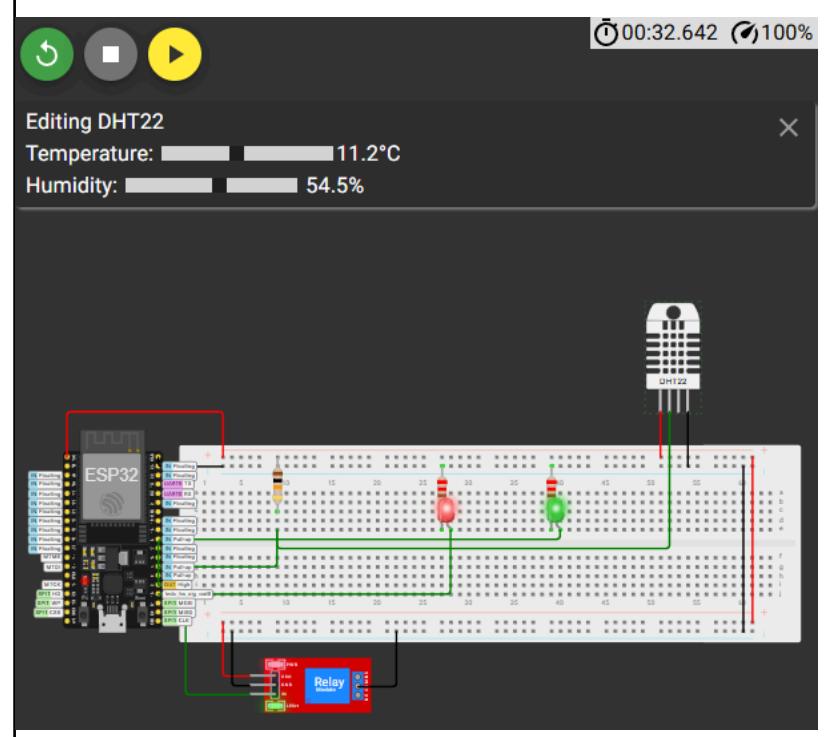
Pengujian dilakukan dengan Wokwi untuk menguji Rangkaian Digital dan Rangkaian Fisik. Pengujian ini berdasarkan pengujian Suhu, Kelembapan, dan Mode Malam.

A. Pengujian Digital

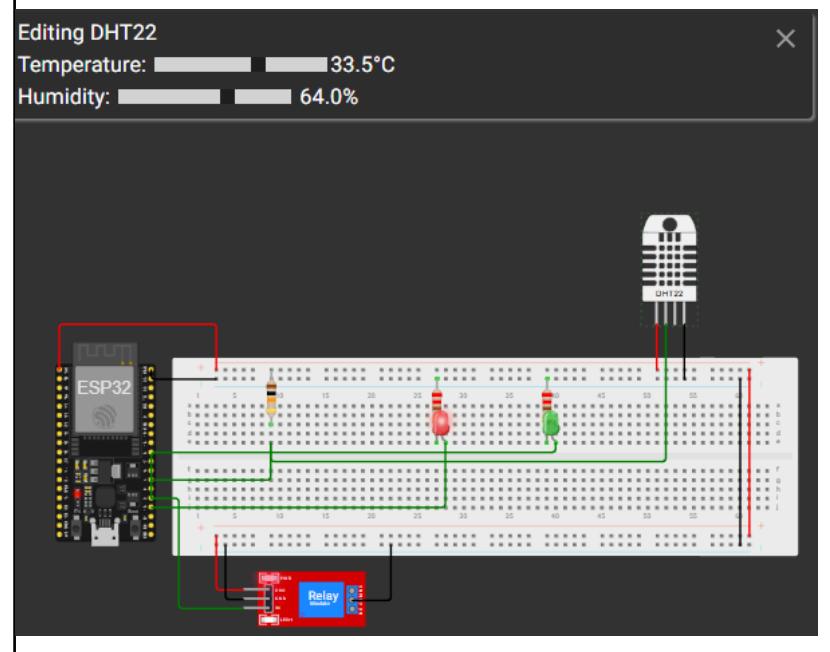
<https://wokwi.com/projects/449672325379650561>

Kondisi	Output
Kondisi Suhu Dingin, secara otomatis nilai Heater PWM = 255	  
Suhu sangat dingin, kelembapan > 42% (MOIST)	

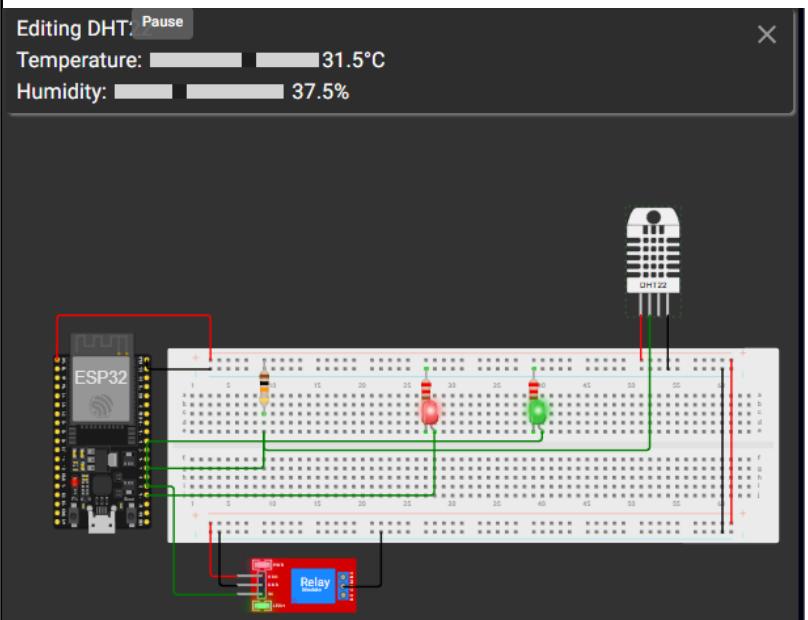
Suhu sangat dingin,
kelembapan < 42%
(DRY)



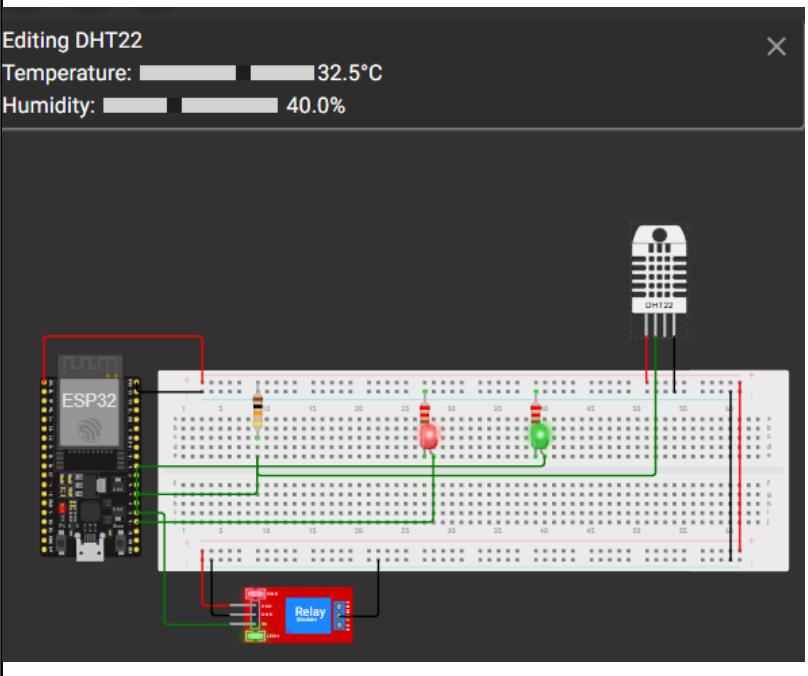
Suhu agak panas,
kelembapan (Tidak
terlalu Terang) >
42% (MOIST)



Suhu agak dingin,
kelembapan < 42%
(DRY)



Suhu optimal
(Lampu Sedang) dan
kelembapan optimal
< 42%



Berdasarkan hasil pengujian tersebut bisa disimpulkan beberapa hal, antara lain:

1. Kondisi Suhu Dingin

Saat suhu lingkungan terbaca **11.8°C** (jauh di bawah target **32°C**), sistem secara otomatis menetapkan nilai **Heater PWM: 255**. Sementara itu, ketika suhu lingkungan mencapai **32°C - 33.5°C**, sistem menurunkan daya **Heater PWM** menjadi **128**.

- LED indikator pemanas menyala dengan kecerahan penuh (sangat terang).

2. Kondisi Suhu Optimal

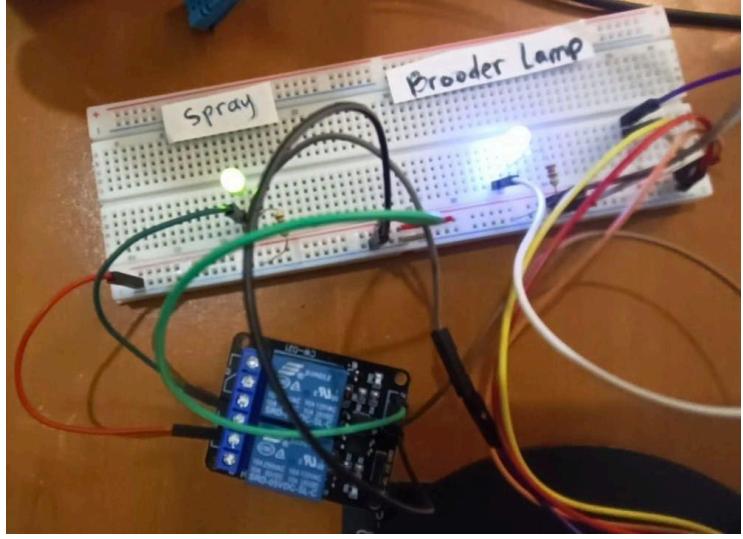
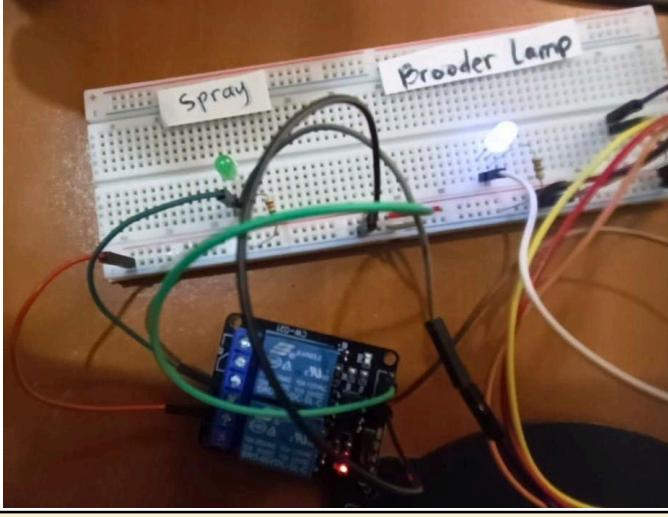
Saat suhu lingkungan mencapai **32°C - 33.5°C**, sistem menurunkan daya **Heater PWM** menjadi **128**.

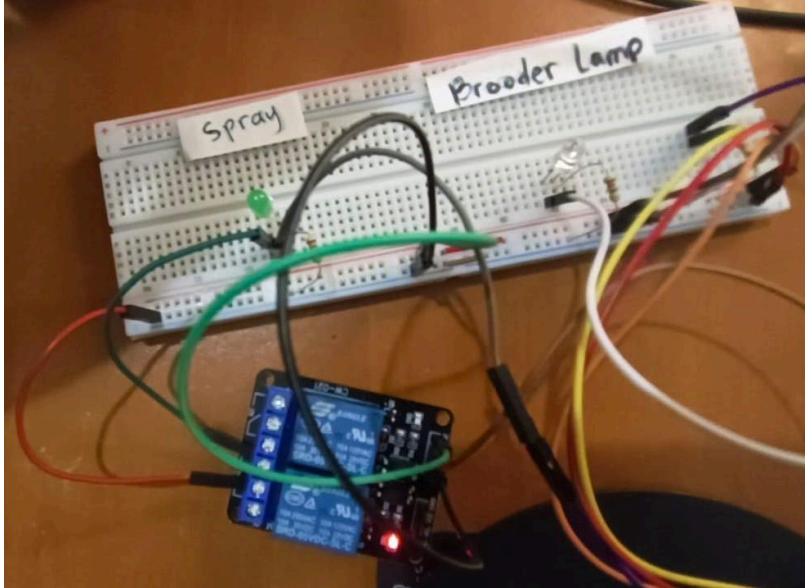
- Cahaya LED pemanas meredup (setengah terang) untuk menjaga stabilitas suhu tanpa *overshoot*.

B. Pengujian Fisik

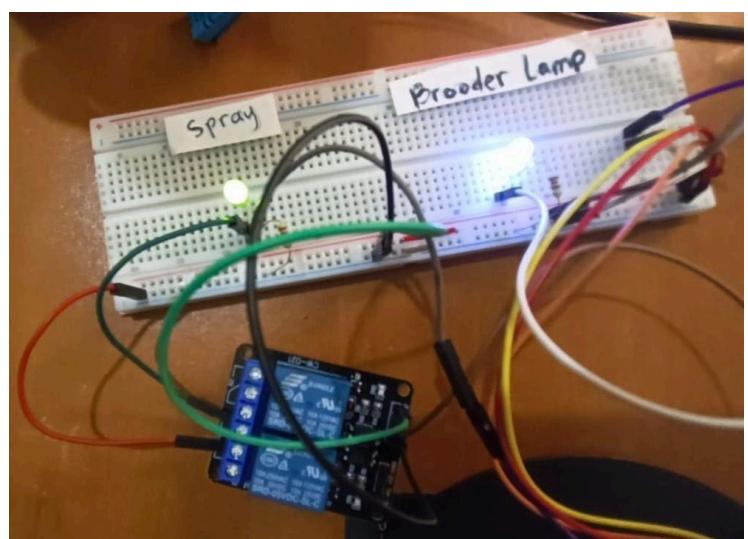
a. Brooder Lamp

Kondisi	Output
Suhu jauh dari optimal (LED putih menyala terang)	

	
	<p style="text-align: center;">Serial Monitor</p> <pre> Mode: DAY Temp: 25.70°C Humidity: 51.00% Heater PWM: 255 Spray: ON Mode: DAY Temp: 25.00°C Humidity: 52.00% Heater PWM: 255 Spray: ON Mode: DAY Temp: 25.00°C Humidity: 52.00% Heater PWM: 255 Spray: ON Mode: DAY Temp: 25.50°C Humidity: 54.00% Heater PWM: 255 Spray: ON</pre>
Suhu dekat dari optimal (LED putih meredup)	
Suhu sudah optimal (LED putih mati)	<p style="text-align: center;">Serial Monitor</p> <pre> Mode: DAY Temp: 32.30°C Humidity: 91.00% Heater PWM: 128 Spray: OFF Mode: DAY Temp: 32.30°C Humidity: 91.00% Heater PWM: 128 Spray: OFF Mode: DAY Temp: 32.60°C Humidity: 90.00% Heater PWM: 128 Spray: OFF Mode: DAY Temp: 32.60°C Humidity: 90.00% Heater PWM: 128 Spray: OFF Mode: DAY Temp: 32.30°C Humidity: 91.00% Heater PWM: 128 Spray: OFF Mode: DAY Temp: 32.30°C Humidity: 91.00% Heater PWM: 128 Spray: OFF</pre>

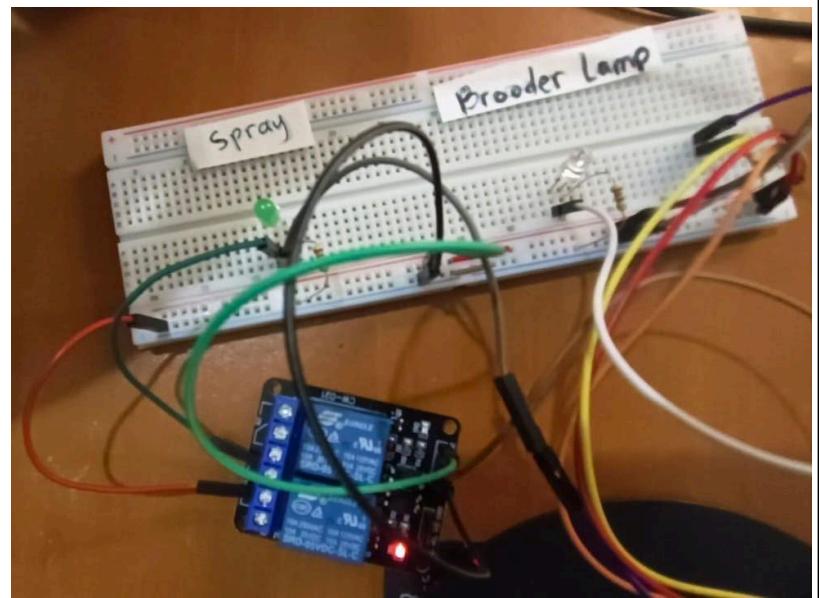
	
	<p style="background-color: #ffffcc; padding: 5px; text-align: center;">Serial Monitor</p> <pre> Mode: DAY Temp: 33.30°C Humidity: 91.00% Heater PWM: 50 Spray: OFF Mode: DAY Temp: 33.30°C Humidity: 91.00% Heater PWM: 50 Spray: OFF Mode: DAY Temp: 33.80°C Humidity: 92.00% Heater PWM: 0 Spray: OFF Mode: DAY Temp: 33.80°C Humidity: 92.00% Heater PWM: 0 Spray: OFF</pre>

b. Dehumidifier Spray

Kondisi	Output
Kelembapan < 60% (LED hijau menyala selama 3 second)	

	<p style="background-color: #ffffcc; padding: 5px; text-align: center;">Serial Monitor</p> <pre>>>> Dehumidifier Spray ON <<< Mode: DAY Temp: 29.50°C Humidity: 61.00% Heater PWM: 255 Mode: DAY Temp: 29.50°C Humidity: 61.00% Heater PWM: 255 Mode: DAY Temp: 29.50°C Humidity: 61.00% Heater PWM: 255 >>> Dehumidifier Spray OFF <<< >>> Dehumidifier Spray ON <<< Mode: DAY Temp: 29.50°C Humidity: 60.00% Heater PWM: 255 Mode: DAY Temp: 29.50°C Humidity: 60.00% Heater PWM: 255 Mode: DAY Temp: 29.30°C Humidity: 59.00% Heater PWM: 255 >>> Dehumidifier Spray OFF <<<</pre>
--	---

Kelembapan > 62%
(LED hijau mati)



Serial Monitor

```
>>> Dehumidifier Spray ON <<<
| Mode: DAY | Temp: 25.50°C | Humidity: 54.00% | Heater PWM: 255 | Spray: ON
>>> Dehumidifier Spray OFF <<<
| Mode: DAY | Temp: 25.80°C | Humidity: 67.00% | Heater PWM: 255 | Spray: OFF
Mode changed to: NIGHT
| Mode: NIGHT | Temp: 25.80°C | Humidity: 67.00% | Heater PWM: 0 | Spray: OFF
| Mode: NIGHT | Temp: 25.30°C | Humidity: 74.00% | Heater PWM: 0 | Spray: OFF
| Mode: NIGHT | Temp: 25.30°C | Humidity: 74.00% | Heater PWM: 0 | Spray: OFF
| Mode: NIGHT | Temp: 25.60°C | Humidity: 77.00% | Heater PWM: 0 | Spray: OFF
| Mode: NIGHT | Temp: 25.60°C | Humidity: 77.00% | Heater PWM: 0 | Spray: OFF
```

3.3 EVALUATION

Pengembangan logika kontrol dibuat lebih dinamis agar target suhu berubah otomatis mengikuti umur ayam (minggu ke-1 vs minggu ke-2) tanpa perlu *flashing* ulang kode program. Hal ini dapat dicapai dengan mengirimkan nilai *Set Point* baru melalui protokol MQTT dari Dashboard Node-RED, atau memprogram algoritma kurva penurunan suhu (*temperature decay curve*) pada ESP32 yang menyesuaikan target suhu berdasarkan variabel "hari ke-n" (misal: 35°C pada hari ke-1, menurun menjadi 30°C pada hari ke-7).

Implementasi modul ESP32-CAM sangat direkomendasikan untuk mendeteksi pola kerumunan anak ayam, yang berfungsi sebagai metode validasi biologis untuk melengkapi data suhu numerik. Sejalan dengan itu, penggantian sensor DHT11 dengan tipe BME280 atau SHT31 diperlukan guna mencapai akurasi pembacaan suhu dan kelembapan setara tingkat industri. Terakhir, ketahanan sistem terhadap gangguan koneksi dapat diperkuat melalui fitur *Local Data Logging* menggunakan modul SD Card, yang memastikan data tetap

tersimpan secara lokal saat jaringan terputus dan diunggah kembali ke Node-RED secara otomatis ketika koneksi pulih.

Sistem RAG saat ini memiliki keterbatasan karena hanya memberikan *fixed context* (100 data terakhir) secara pasif kepada AI. Untuk pengembangan masa depan, disarankan mengimplementasikan konsep **Agentic AI** yang lebih canggih. Menggunakan implementasi tool calling, di mana chatbot tidak hanya menerima data yang disodorkan, tetapi dapat secara aktif "meminta" data spesifik sesuai pertanyaan pengguna. Contoh: Jika pengguna bertanya "*Bagaimana kondisi suhu kemarin siang?*", Agentic AI dapat menjalankan fungsi `getHistory(date='yesterday', time='noon')` untuk mengambil data relevan secara dinamis, sehingga analisis tidak terbatas pada jendela waktu terkini saja.

CHAPTER 4

CONCLUSION

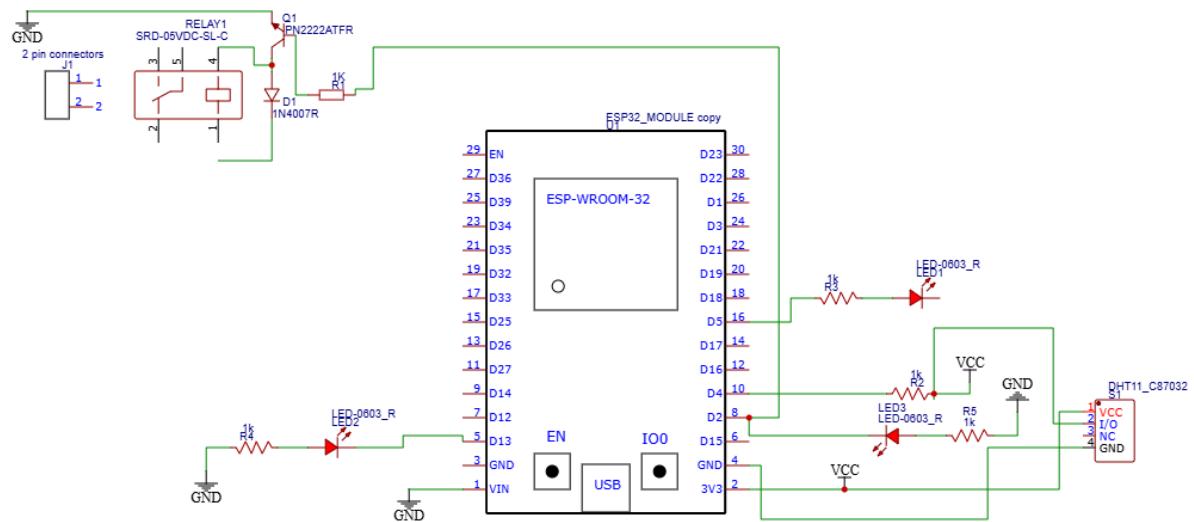
Berdasarkan perancangan dan pengujian yang telah dilakukan, sistem "ChickCare" berhasil diimplementasikan sebagai solusi IoT terintegrasi untuk pemantauan dan pengendalian lingkungan inkubator ayam. Mekanisme kontrol berbasis *Pulse Width Modulation* (PWM) pada mikrokontroler ESP32 terbukti efektif dalam menjaga stabilitas suhu secara presisi, meminimalkan fluktuasi ekstrem yang berisiko bagi kesehatan unggas pada fase *brooding*. Nilai tambah utama dari sistem ini terletak pada integrasi kecerdasan buatan Google Gemini melalui arsitektur *Retrieval-Augmented Generation* (RAG) di Node-RED, yang mampu mengubah data log sensor mentah menjadi wawasan naratif dan rekomendasi tindakan yang mudah dipahami. Secara keseluruhan, proyek ini mendemonstrasikan bahwa konvergensi antara teknologi *embedded system* yang andal dengan kemampuan analisis *Generative AI* memiliki potensi besar untuk meningkatkan efisiensi operasional dan standar keselamatan dalam industri peternakan modern.

REFERENCES

- [1] Nutrena, “Heat Lamps For Chicks,” *Nutrena Animal Feeds*, Jan. 18, 2024.
<https://nutrenaworld.com/heat-lamps-for-chicks/> (accessed Dec. 02, 2025).
- [2] Czarick, Michael, dkk. “Poultry Housing Tips Do Chicks Benefit From 24 Hours of Light?” 2022. Department of Poultry Science of University of Georgia [Online]. Available: <https://www.poultryventilation.com/wp-content/uploads/vol34n7.pdf> (accessed Dec. 03, 2025)

APPENDICES

Appendix A: Project Schematic



Appendix B: Documentation

