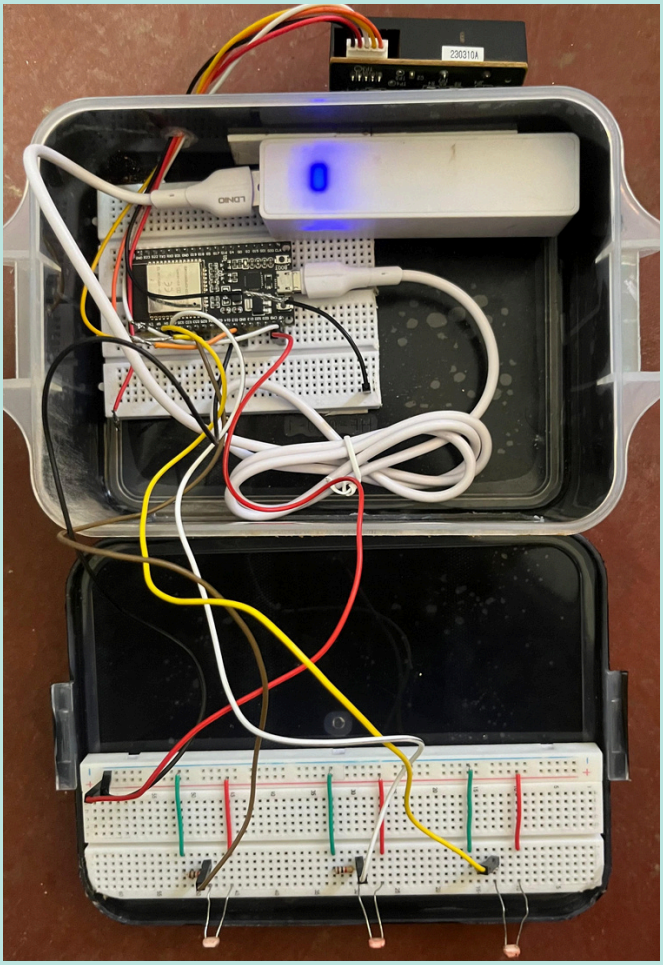
PROJECT REPORT

DUST LEVEL MONITORING DEVICE

TABLE OF CONTENT



REPRESENTATIVES: S.U. DEWASURENDRA | G.LD. ANJANA | L.S.V. JAYATHISSA | K.M.P.M. BANDARA

1. Introduction

This report outlines the development and implementation of a comprehensive IoT-enabled dust monitoring system for solar panels in solar farms. The system integrates various sensors, microcontrollers, and IoT platforms to monitor both panel dust accumulation and atmospheric dust levels. By combining hardware components and software platforms, the system aims to provide real-time insights into dust levels, enabling proactive maintenance and optimization of solar panel performance. The main benefits of the project include its cost-effectiveness and the ability to determine the optimal time for cleaning the solar panels based on accumulated dust levels.

2. Objectives

- Develop a panel dust monitoring system using Light Dependent Resistors (LDRs) to detect dust accumulation on solar panels.
- Implement an atmospheric dust monitoring system utilizing a DSM501A PM 2.5 dust sensor to measure ambient dust levels.
- Integrate an ESP32 microcontroller for data processing and communication with the Node-RED IoT platform.
- Utilize Node-RED to visualize sensor data to clearly communicate dust levels and device status to users.

3. Methodology

1. PANEL DUST MONITORING

To develop the panel dust monitoring system, we utilized three Light Dependent Resistors (LDRs) positioned below a thin transparent sheet. This sheet simulates the surface of solar panels in a solar farm. As dust accumulates on the transparent sheet over time, the amount of sunlight reaching the LDRs decreases, leading to changes in their resistance. By measuring these changes, we can detect when it is necessary to clean the solar panels.

Figure 1: Light Dependent Resistor (LDR)

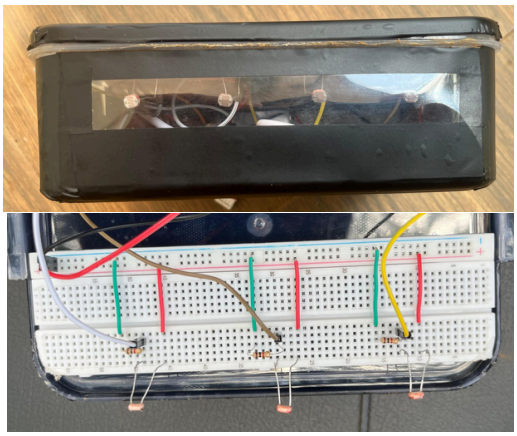


Figure 2: LDR setup of the device

2. ATMOSPHERIC DUST MONITORING

For the atmospheric dust monitoring component of the device, we utilized a DSM501A PM 2.5 dust sensor to measure the level of atmospheric dust at the solar plant location. This allowed us to anticipate the likelihood of dust accumulation on the solar panels based on the ambient dust levels. The pins of the dust sensor were soldered onto jumper wires to connect them to the breadboard inside the device

Figure 3: DSM501A PM2.5 Dust Sensor

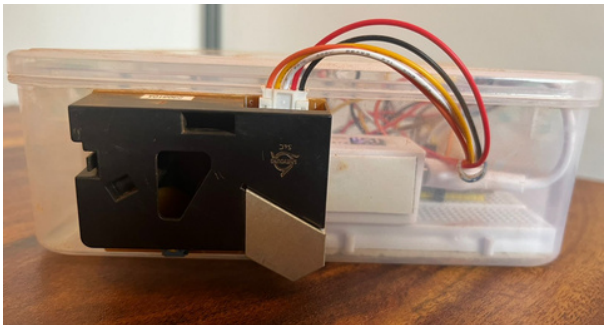


Figure 4: dust sensor setup of the device

3. MICROCONTROLLER - ESP WROOM 32

In our methodology, we integrated the ESP32 WROOM, an IoT microcontroller, into the system for data processing and communication through WiFi with Node-RED platform. The ESP32 WROOM served as the central processing unit responsible for collecting sensor data, implementing calibration algorithms, and transmitting data to the Node-RED dashboard.

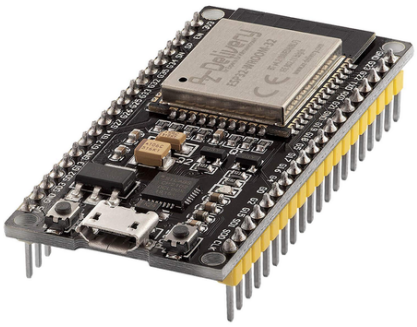


Figure 5: NodeMCU ESP32 (ESP-WROOM-32)

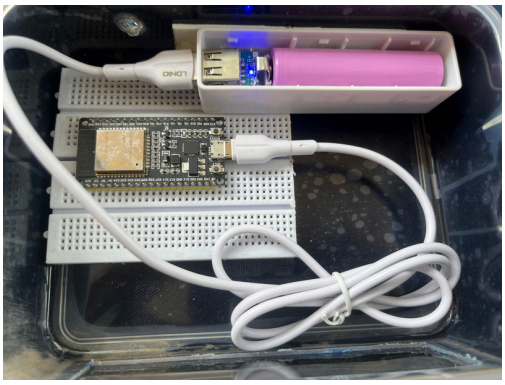


Figure 6: Microcontroller and Power Source Connection in the device

4. POWER SOURCE - 2200 MAH 1.6A RECHARGEABLE LI-ION BATTERY CELL

In our setup, we utilize a 2200 mAh 1.6A Rechargeable Li-Ion Battery Cell as the power source for the ESP32 microcontroller. However, since the battery output voltage is 3.7V and the ESP32 operates at 5V, we employ a Power Bank Case for 1x18650 Batteries to convert the voltage to 5V. This power bank case serves as a voltage regulator, ensuring compatibility between the battery output and the ESP32 input requirements. By connecting the power bank to the ESP32 using an external cable, we establish a reliable power supply for the microcontroller, enabling continuous operation and functionality of the monitoring device.

Figure 7: 2200 mAh 1.6A Rechargeable Li-Ion Battery Cell

Figure 8: Power Bank Case for 1x18650 Batteries

5. SOFTWARES

1. Arduino IDE: In this environment, programming was conducted in C++ to facilitate data acquisition from the LDR sensors and DSM501A dust sensor. The programming involved reading sensor data, categorizing them into clean, medium, and high dust levels, and transmitting the outputs via Wi-Fi to an MQTT broker. The Arduino IDE, being compatible with the ESP32 microcontroller, provided the necessary tools for developing and uploading the code to the microcontroller Additionally, a feature was implemented to monitor the panel dust levels over a specified duration of time. If the output remained consistent within a single state—whether indicating a clean surface, medium dust level, or high dust level—during this period, it would also be published through the MQTT broker.

```
#include <Arduino.h>
#include <WiFi.h>
#include <PubSubClient.h>

// Replace with your WiFi credentials
const char* ssid = "Huawei y9 prime";
const char* password = "12345678";

// Replace with your MQTT broker details
const char* mqtt_server = "test.mosquitto.org";

const char* mqtt_topic = "dust sensor";
const char* mqtt_topic_output = "ldr_sensor_output"; // Topic for output
const char* mqtt_topic_1 = "ldr_sensor1"; //   Topic for output
const char* mqtt_topic_2 = "ldr_sensor2"; // Topic for output
const char* mqtt_topic_3 = "ldr_sensor3"; // Topic for output
const char* mqtt_topic_4 = "ldr_sensor4"; // Topic for output
const char* mqtt_topic_text = "text";

const int LDR_PINS[3] = {33, 34, 35};
const int LDR_1 = analogRead(LDR_PINS[0]); // Assuming initial readings available
const int LDR_2 = analogRead(LDR_PINS[1]);
const int LDR_3 = analogRead(LDR_PINS[2]);

int pin = 25;
unsigned long duration;
```

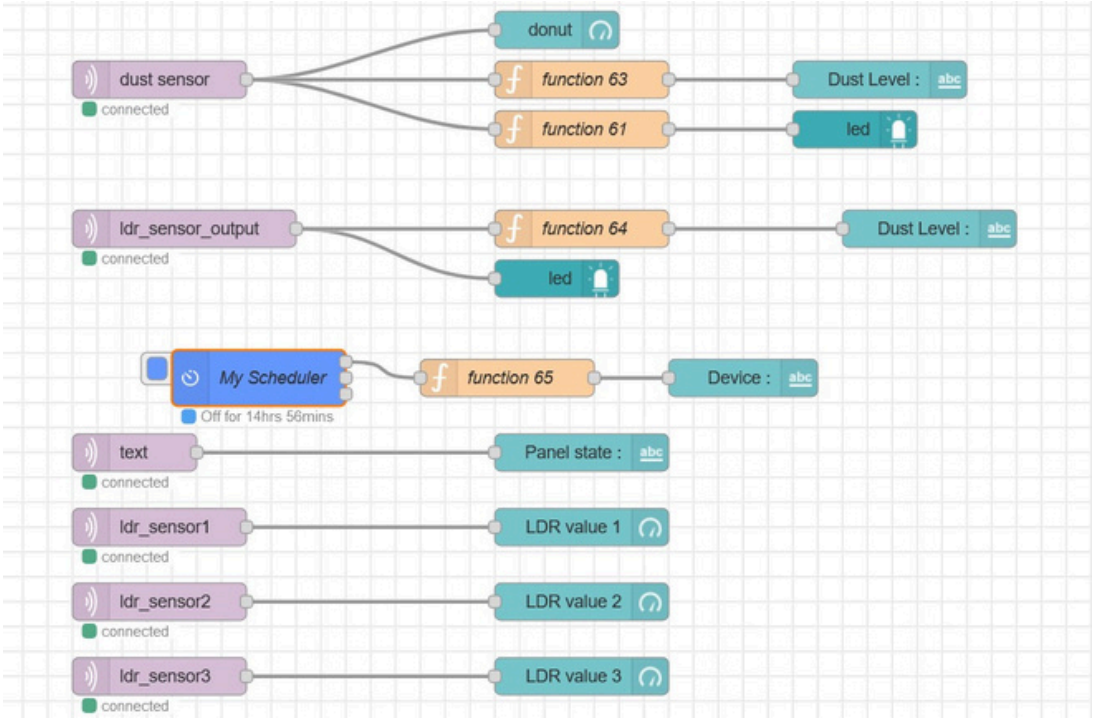
```
const float referenceVoltage = 5.0f; // Assuming 5V power supply for Arduino Uno

int outputHistory[10]; // Array to store last 10 output values (adjust size)
int cleanCount = 0;

String cleanStatusMessage;

WiFiClient espClient;
PubSubClient client(espClient);

void setup() {
  Serial.begin(9600); // Start serial communication for printing data (optional)
  analogReadResolution(12);

  pinMode(pin, INPUT);

  // Connect to WiFi
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("WiFi connected");

  // Connect to MQTT broker
  client.setServer(mqtt_server, 1883);
}

void loop() {
  // Check MQTT connection and reconnect if needed
  if (!client.connected()) {
    reconnect();
  }
  client.loop();

  // Read pulse width
  duration = pulseInLong(pin, LOW);

  int sensorReadings[3];  // Array to store sensor readings

  for (int i = 0; i < 3; i++) {
    sensorReadings[i] = analogRead(LDR_PINS[i]);
  }
```

```
Serial.println("Sensor Readings:");
for (int i = 0; i < 3; i++) {
  Serial.print("LDR ");
  Serial.print(i + 1);
  Serial.print(": ");
  Serial.println(sensorReadings[i]);
}

int output;

// Check for all LDR values between 3500 and 4095
bool allWithinRange = true;
for (int i = 0; i < 3; i++) {
  if (sensorReadings[i] < 4000 || sensorReadings[i] > 4095) {
    allWithinRange = false;
    break;
  }
}

// Check for all within 3000 and 4095 (original logic)
int countWithinRanges = 0;
for (int i = 0; i < 3; i++) {
  if (sensorReadings[i] >= 3000 && sensorReadings[i] <= 4095) {
    countWithinRanges++;
  }
}

if (allWithinRange) {
  output = 0;  // All LDR values between 3500 and 4095
} else if (countWithinRanges == 3) {  // All within 3000 and 4095 (unchanged)
  output = 1;
} else {
  output = 2;  // None between 3500-4095 or not all within 3000-4095
}

// Update output history
outputHistory[(cleanCount % 10)] = output; // Circular buffer approach

// Check for consistent output (2)
cleanCount = 0;
for (int i = 0; i < 10; i++) {
  if (outputHistory[i] == output) {
    cleanCount++;
  } else {
    break;
```

```
    }
  }

  // Print message based on clean count and output value
  if (cleanCount == 10) {
    if (output == 0) {
      cleanStatusMessage = "Solar panel is clean";
    } else if (output == 1) {
      cleanStatusMessage = "Solar panel is not clean";
    } else if(output == 2){
      cleanStatusMessage = "Urgently clean the solar panel";
    }
    cleanCount = 0; // Reset clean count (optional)
  }

  // Publish output to MQTT topic
  client.publish(mqtt_topic, String(duration).c_str());
  client.publish(mqtt_topic_output, String(output).c_str());
  client.publish(mqtt_topic_1, String(sensorReadings[0]).c_str());
  client.publish(mqtt_topic_2, String(sensorReadings[1]).c_str());
  client.publish(mqtt_topic_3, String(sensorReadings[2]).c_str());
  client.publish(mqtt_topic_text, cleanStatusMessage.c_str());

  // Print sensor values and output (optional)
  Serial.print("Dust sensor value: ");
  Serial.println(duration);

  Serial.print("Output: ");
  Serial.println(output); // Print the output value (0, 1, or 2)

  delay(1000); // Delay between readings (adjust as needed)
}

void reconnect() {
  while (!client.connected()) {
    if (client.connect("ESP32_Dust_Sensor")) {  // Give a unique client ID
      Serial.println("Connected to MQTT broker");
    } else {
      Serial.print("Failed to connect, rc=");
      Serial.print(client.state());
      Serial.println(" try again in 5 seconds");
      delay(5000);
    }
  }
}
```

Figure 9: Arduino IDE full code

2. Node-RED Platform: Node-RED served as the foundational IoT platform for the project. Here, a Node-RED flow was created to receive the published sensor data and visualize it in a dashboard format. This platform enabled seamless integration and visualization of the sensor data, providing real-time insights into atmospheric and panel dust levels for monitoring and maintenance purposes.



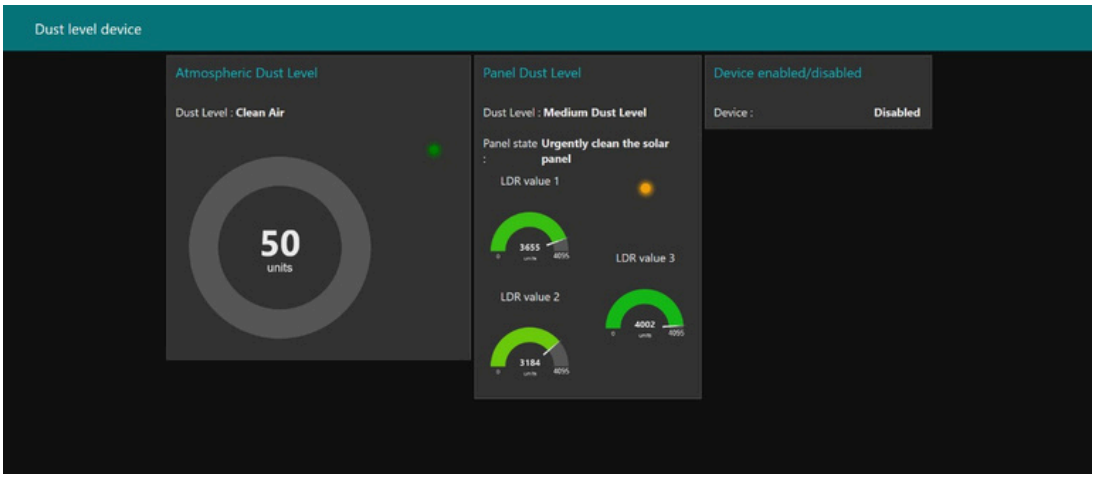


Figure 10: Node-RED flow and the complete dashboard

6. FINAL SETUP

In the final device, all the components, including the panel dust monitoring setup, DSM501A dust sensor, power source, and ESP32, have been integrated inside. Additionally, jumper wires and a breadboard were utilized for connections. Moreover, to shield the LDRs from excessive sunlight exposure, the final box is coated in black. Only the thin transparent sheet is left exposed to sunlight for accurate LDR readings.

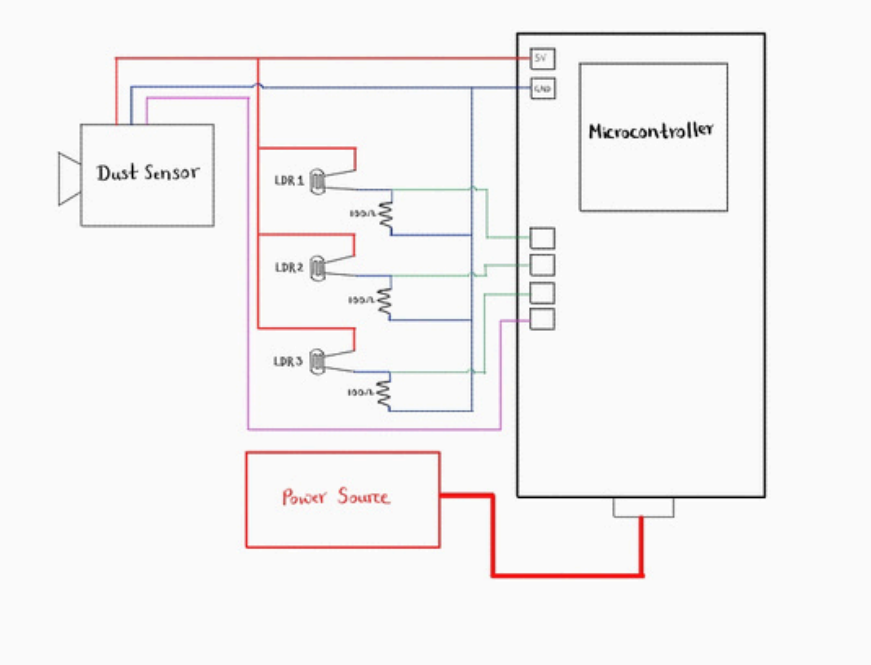


Figure 11: Sketch of the full setup

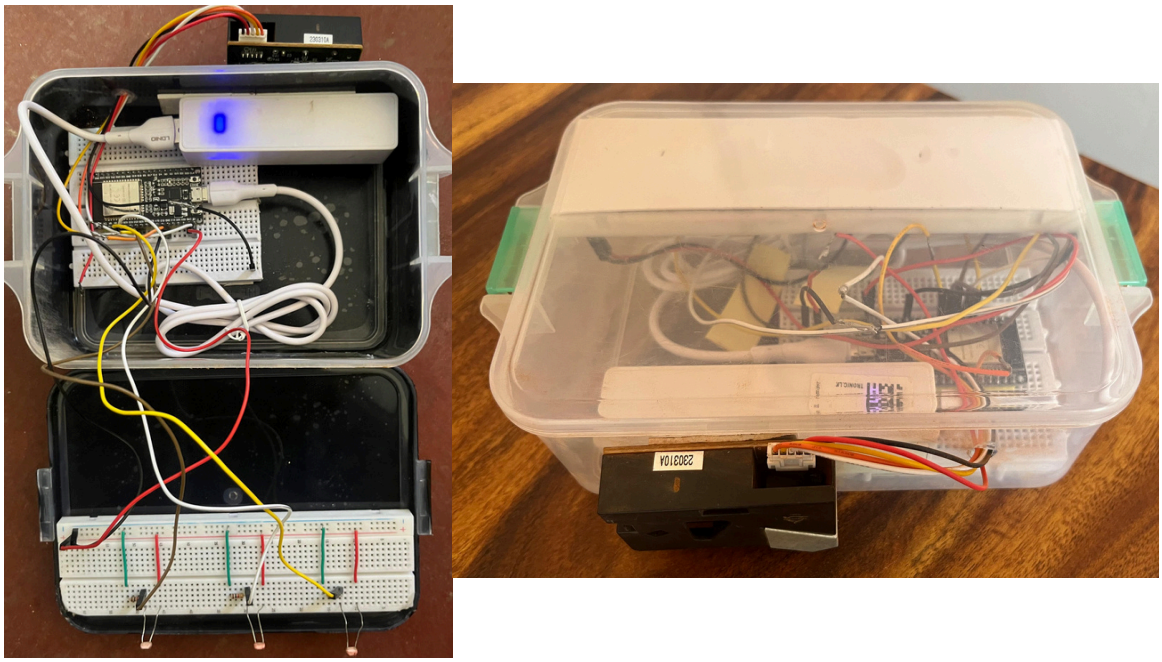


Figure 12: Finalized and completed device

4. Calibrations

1. CALIBRATION PROCESS - PANEL DUST MONITORING

The calibration process involved establishing baseline readings for the LDRs under different levels of dust accumulation. This was achieved through the following steps:

- **Clean surface calibration:** The transparent sheet was thoroughly cleaned to obtain baseline LDR readings for a dust-free surface. These readings served as the reference point for subsequent measurements.

- **Medium dust level calibration:** A small amount of dust was intentionally deposited on the transparent sheet, simulating a moderate level of dust accumulation. The LDR readings under this condition were recorded to establish the corresponding resistance values.

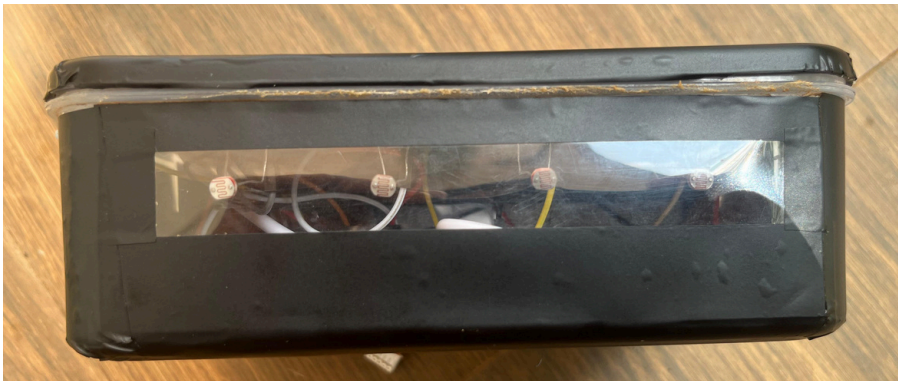
- **High dust level calibration:** Additional dust was added to the transparent sheet to create a more significant dust layer, representing a higher level of dust accumulation. The corresponding LDR readings were recorded for this condition.

- **Multiple attempts and average calibration:** To ensure accuracy, the calibration process was repeated multiple times, and the average LDR readings for each dust level were calculated. This provided more reliable calibration values for detecting different levels of dust accumulation on the solar panels.

The following presents the calibrated values for different dust levels obtained through the calibration process.

Dust Level	LDR Reading Range
Clean Surface	4000 - 4095
Medium Dust	3000 - 3500
High Dust	0 - 3000

Clean surface :



Medium dust layer :

High dust layer :

Figure 13: Calibration process of the panel dust monitoring : clean surface, medium dust layer, larger dust layer on the transparent sheet surface

2. CALIBRATION PROCESS - ATMOSPHERIC DUST MONITORING

To calibrate the dust sensor, we followed a simple procedure:

- **Baseline reading**: We obtained a baseline reading for a clean, dust-free atmosphere by taking readings inside an enclosed area, where the dust levels are typically minimal. The reading we consistently obtained in this environment was 0, indicating negligible atmospheric dust.

- **Medium dust Level calibration**: We then took the device outside of the site to an area with moderate dust levels. This allowed us to obtain readings corresponding to a medium atmospheric dust level.

- **High dust level calibration**: To simulate higher atmospheric dust levels, we created an environment with increased dust concentration. In this setting, we obtained readings indicative of a higher medium dust level.

The calibration process yielded the following calibrated values for different atmospheric dust levels:

Dust Level	LDR Reading Range
Clean Atmosphere	0-22500
Medium Dust	22500- 100800
High Dust	100800 - 400000

By averaging multiple attempts, we ensured the accuracy of the calibrated values, enabling the device to effectively monitor atmospheric dust levels and anticipate potential dust accumulation on the solar panels.

5. Results

In the Node-RED dashboard, there are two distinct sections dedicated to panel dust monitoring and atmospheric dust monitoring, each equipped with LED indicators to convey the respective results.

Additionally, the system schedules a time with minimal sunlight changes and indicates when the device is enabled for that period, remaining disabled for the rest.

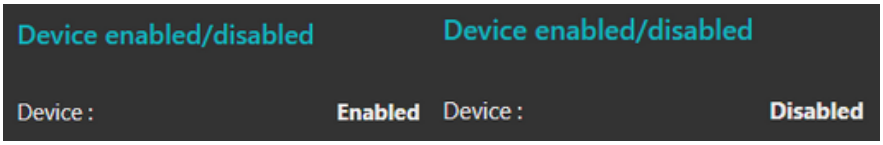


Figure 14 : The Node-RED dashboard displays a "Enabled" message from 11:00 a.m. to 1:00 p.m., and for the rest of the time, it shows a "Disabled" message.

1. ATMOSPHERIC DUST MONITORING LED INDICATORS:

- **Green LED:** Indicates a clean atmosphere based on the calibrated ranges. When the dust sensor reading falls within the clean range, the green LED will light up. Additionally, on the Node-RED dashboard, the state is displayed as "Clean Atmosphere."

- **Yellow LED:** Illuminates when the dust sensor reading corresponds to a medium dust level, as per the calibrated ranges. Simultaneously, on the Node-RED dashboard, the state is visually represented as "Medium Dust Level."

- **Red LED:** Lights up to signify a high dust level, indicating a critical level of atmospheric dust concentration. Correspondingly, the Node-RED dashboard displays the state as "High Dust Level."

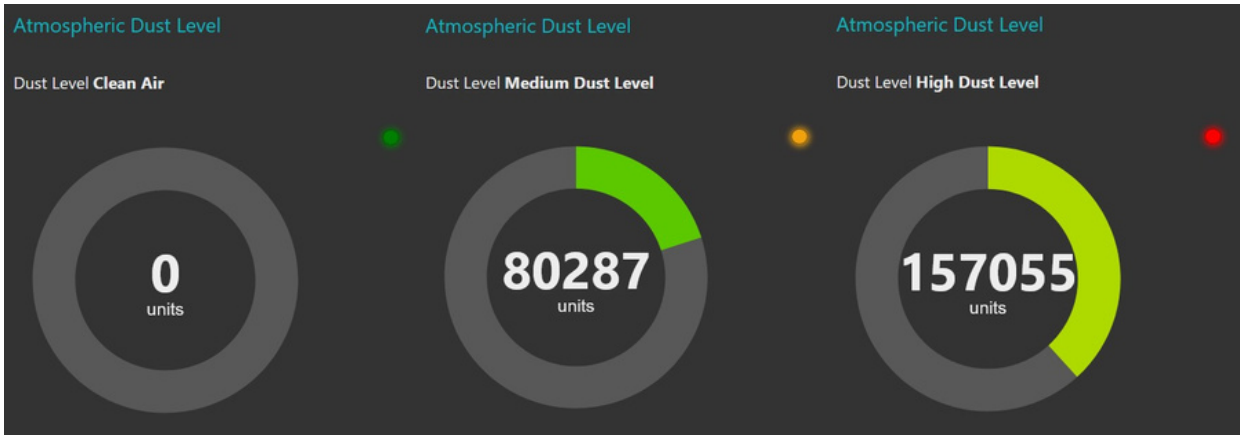


Figure 15 : LED indicators on the Node-RED dashboard display green, yellow, and red for indicating a clean atmosphere, medium dust level, and high dust level, respectively, for atmospheric dust monitoring.

2. PANEL DUST MONITORING LED INDICATORS:

- **Green LED:** Signals a clean panel surface, aligning with the calibrated ranges for panel dust monitoring when all 3 LDRs are in the clean surface range. Additionally, the dashboard will display "Clean Surface" to represent this state.

- **Yellow LED:** Indicates a moderate level of dust accumulation on the solar panels. It is recommended to clean the panels when the LED is yellow. Correspondingly, the dashboard will show "Medium Dust Level" to indicate this state.

- **Red LED:** Lights up to indicate a significant accumulation of dust on the solar panels, prompting immediate cleaning action. The dashboard will also display "High Dust Level" to reflect this critical state.

Furthermore, to address the issue of frequent sensor reading fluctuations and ensure accurate monitoring, the dashboard will display persistent states if the device maintains a specific condition for a set period. This means that if the transparent sheet (solar panel) remains clean, shows medium dust levels, or indicates high dust levels consistently over time, the dashboard will clearly indicate this prolonged state. This feature helps reduce errors caused by rapid changes in sensor readings due to sun light changes, allowing for a more reliable understanding of the current environmental conditions.

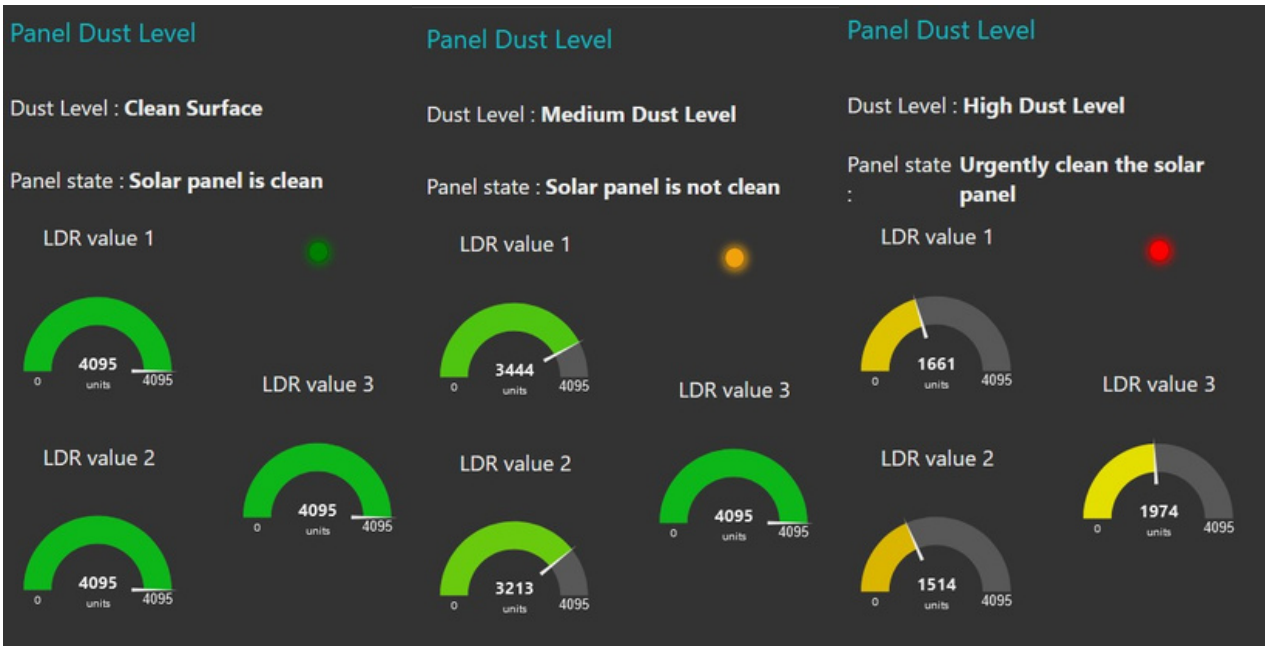


Figure 16 : LED indicators on the Node-RED dashboard display green, yellow, and red for indicating a clean surface, medium dust level, and high dust level, respectively, for panel dust monitoring. Additionally, the system monitored the state over a specified time period and displayed corresponding messages.

6. Constraints

- The project operates only during daytime due to its reliance on sunlight for LDR readings, limiting its effectiveness during nighttime or low-light conditions.
- The LDR values are calibrated for a specific location; therefore, the ranges may vary with different locations.
- LDR readings can be affected by changes in sunlight, potentially leading to inaccurate dust level assessments.
- The LDR sensors used in the project may not be highly sensitive, leading to potential inaccuracies in detecting dust accumulation.
- The DSM501A dust sensor may not be entirely reliable and can detect other movements besides dust particles, potentially leading to false readings.
- Data communication is limited by the range of Wi-Fi, necessitating additional sensors or alternative communication methods for longer-range transmission.
- The power source, a rechargeable Li-Ion battery, requires recharging after a certain period, leading to potential downtime. This can be addressed by using higher capacity batteries or integrating solar-powered circuits for continuous operation.

7. Conclusion

In summary, our dust monitoring system for solar panels integrates sensors, microcontrollers, and IoT platforms to provide real-time insights into dust accumulation. Despite limitations like daylight dependency and sensor sensitivity issues, the system offers a cost-effective solution for proactive maintenance. Future enhancements may include better sensors and improved communication methods for increased reliability.