NATIONAL TEXTILE UNIVERSITY , FAISALABAD

Project Title	Smart IoT Power Monitor System
Members	Abdullah Tahir Raees Ul Mujtaba Talha Mehmood
Reg. Numbers	23-NTU-CS-1004 23-NTU-CS-1276 23-NTU-CS-1286
Course	Embedded Of IoT Systems

Project Title: Smart IoT Power Monitor System

1. Introduction

This project is an Internet of Things (IoT) system designed to monitor the real-time power consumption and environmental conditions of a DC load (Fan and LED). It measures Voltage, Current, Power, Temperature, and Humidity using an ESP32 microcontroller and sends the data over Wi-Fi to a central dashboard.

2. How It Works (The Logic)

1. **Sense:** The ESP32 reads electrical data from the **INA219 sensor** and weather data from the **DHT11 sensor**.
2. **Send:** Every 2 seconds, the ESP32 sends this data as a JSON packet via Wi-Fi to a Python server (HTTP POST).
3. **Store:** The Python backend saves the data into a **Supabase (PostgreSQL)** cloud database.
4. **Visualize:** A React web dashboard fetches the latest data and displays live charts and gauges.

3. Hardware Components

- **Microcontroller**: ESP32 Development Board (Wi-Fi enabled).

- **Power Sensor**: INA219 (Measures High-Side Voltage and DC Current).
- **Environment Sensor**: DHT11 (Temperature & Humidity).
- **Power Supply:** 9V Battery stepped down to 5V using an **LM2596 Buck Converter**.
- **Load:** 5V DC Fan and LED.
- **Miscellaneous:** Breadboard, Jumper Wires, Resistors (220Ω for LED).

4. Circuit Wiring (Pin Mapping)

A. Power Distribution

- **Input**: 9V Battery connected to **LM2596 Input**.
- **Regulation:** LM2596 Output adjusted to **5.0V**.
- **ESP32 Power**: Powered via **Laptop USB** (for stable Wi-Fi performance).

B. Sensor Connections

Component	**Pin Label**	**Connected To**	**Description**
INA219	VCC	ESP32 3V3	Sensor Power
	GND	ESP32 GND	Common Ground
	SDA	**GPIO 21**	I2C Data Line

Component	**Pin Label**	**Connected To**	**Description**
	SCL	**GPIO 22**	I2C Clock Line
	Vin+	LM2596 OUT+	**Power Entry** (From Battery)
	Vin-	Fan Red Wire	**Power Exit** (To Load)
DHT11	Signal	**GPIO 18**	Data Pin
	VCC	ESP32 3V3	Sensor Power
	GND	ESP32 GND	Ground

C. Load Connections (The Circuit)

- **Fan Positive (+):** Connected to **INA219 Vin-**.
- **Fan Negative (-):** Connected to **Common GND**.
- *Result:* The INA219 sits "in the middle" of the positive wire to count the electrons passing through.

5. Software Architecture

A. Firmware (ESP32)

- **Language:** C++ (Arduino Framework).
- **Libraries Used:** Adafruit_INA219, DHT, WiFi, HTTPClient.
- **Function:**
    - Connects to Wi-Fi.
    - Reads sensors.

    - Formats data into JSON: {"voltage": 4.2, "current": 150, "temperature": 20 ...}.
    - Posts data to http://192.168.1.XX:8000/readings/.

B. Backend (Server)

- **Language:** Python.
- **Framework:** FastAPI (with Uvicorn).
- **Database:** Supabase (Cloud PostgreSQL).
- **Function:**
    - **POST /readings:** Receives data from ESP32 and inserts it into the database.
    - **GET /readings:** Retrieves the latest 10 records (sorted by ID) for the frontend.

C. Frontend (Dashboard)

- **Framework:** React.js (Vite).
- **Libraries:** Axios (API calls), Recharts (Graphs).
- **Function:** Polls the backend every 2 seconds to update charts live.

6. Step-by-Step Running Guide

Step 1: Start the Backend

Open your terminal in the backend folder and run:

Bash

```
uvicorn main:app --host 0.0.0.0 --port 8000 --reload
```

- *Success Check:* You see "Application startup complete."

Step 2: Power the Hardware

- Plug the ESP32 into your laptop (USB).
- Plug the 9V Battery into the LM2596 (Power for the Fan).
- *Success Check:* The Fan spins, and the ESP32 Serial Monitor says Server Response: 200.

Step 3: Launch the Dashboard

Open your terminal in the frontend folder and run:

Bash

```
npm run dev
```

- Open your browser to http://localhost:5173.
- *Success Check:* You see the Voltage and Temperature charts updating live!

7. Troubleshooting (Common Issues)

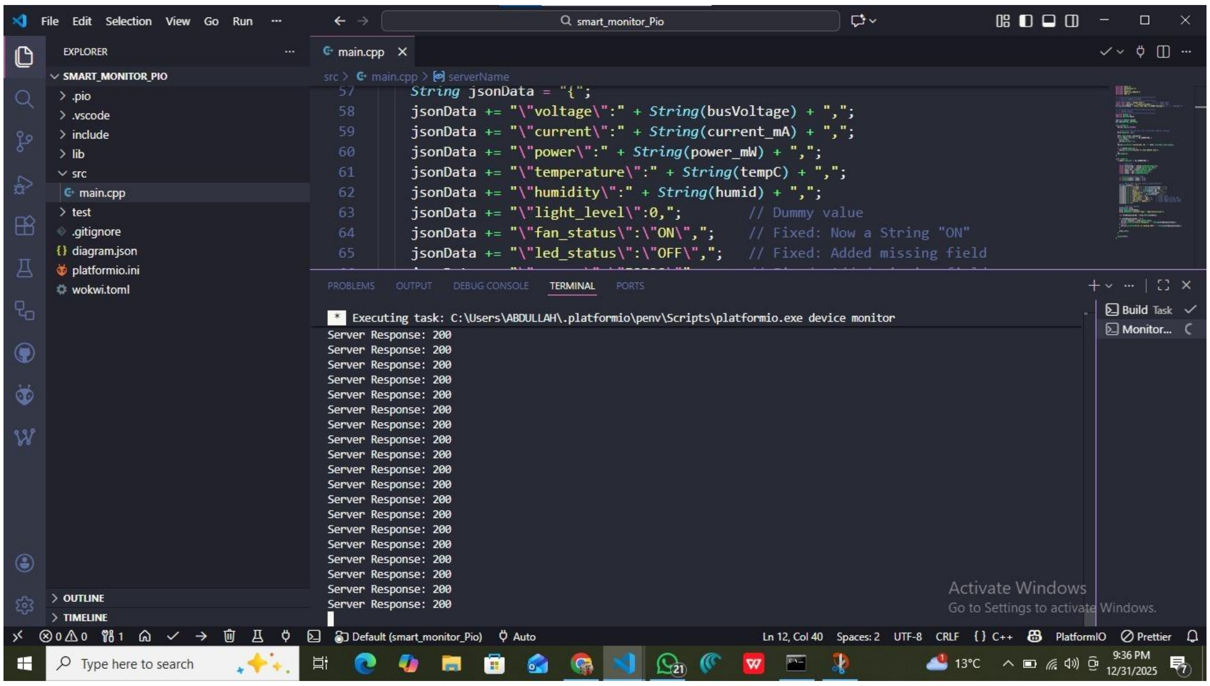
- **Issue**: Dashboard shows old/stuck data (e.g., 12.5V).
  - **Fix**: The database is sending old rows. Sort by ID in the backend (order("id", desc=True)).
- **Issue**: ESP32 Serial says Failed to find INA219.
  - **Fix:** Swap the SDA (21) and SCL (22) wires.
- **Issue**: ESP32 Serial says Error -1 or Connection Refused.
  - **Fix:** Windows Firewall is blocking Python. Turn off Firewall or allow port 8000. Ensure server is running on 0.0.0.0.

Github link:

ScreenShots

Project Diagram:

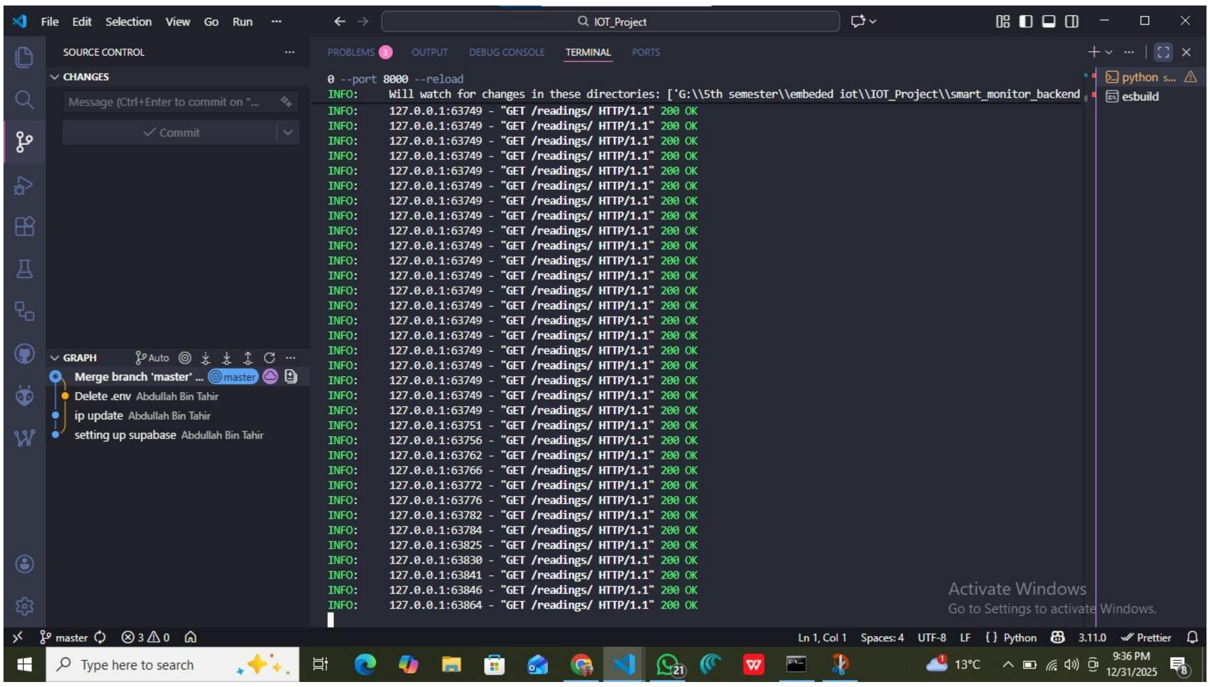
➢ Serial Monitor



```
    String jsonData = "{";
    jsonData += "\"voltage\":" + String(busVoltage) + ",";
    jsonData += "\"current\":" + String(current_mA) + ",";
    jsonData += "\"power\":" + String(power_mW) + ",";
    jsonData += "\"temperature\":" + String(tempC) + ",";
    jsonData += "\"humidity\":" + String(humid) + ",";
    jsonData += "\"light_level\":0,";         // Dummy value
    jsonData += "\"fan_status\":\"ON\",";     // Fixed: Now a String "ON"
    jsonData += "\"led_status\":\"OFF\",";    // Fixed: Added missing field
```

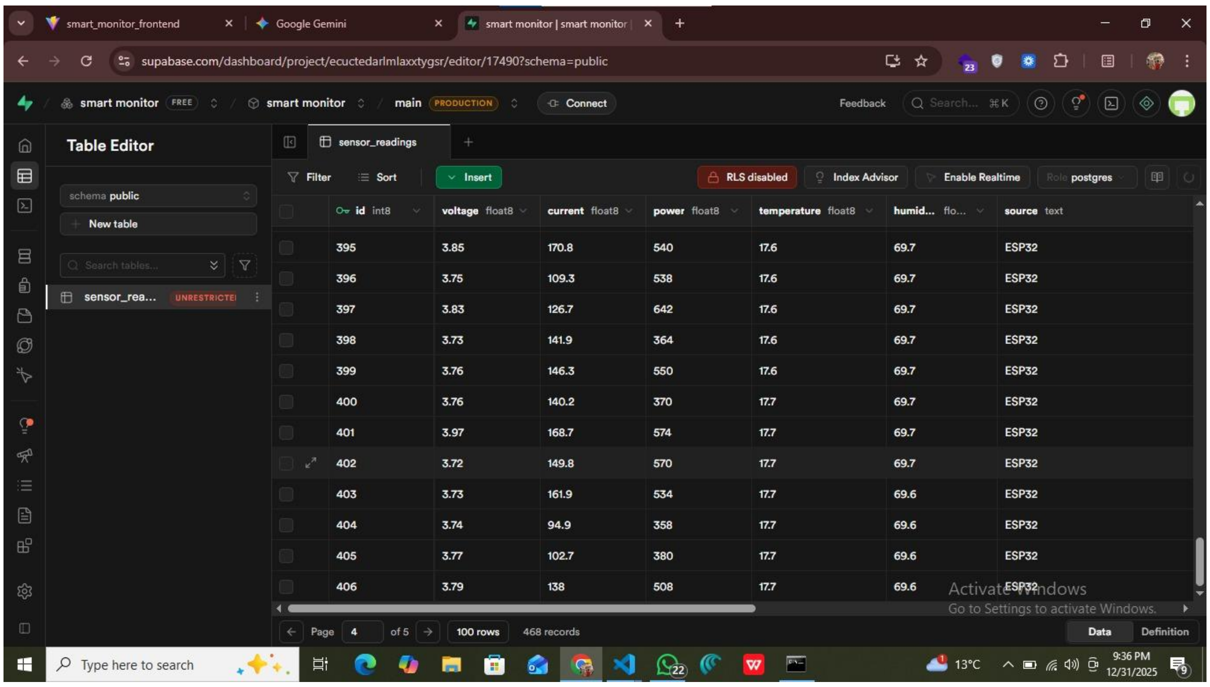
```
 * Executing task: C:\Users\ABDULLAH\.platformio\penv\Scripts\platformio.exe device monitor
Server Response: 200
Server Response: 200
Server Response: 200
Server Response: 200
Server Response: 200
Server Response: 200
Server Response: 200
Server Response: 200
Server Response: 200
Server Response: 200
Server Response: 200
Server Response: 200
Server Response: 200
Server Response: 200
Server Response: 200
Server Response: 200
Server Response: 200
Server Response: 200
Server Response: 200
```

➢ Backend



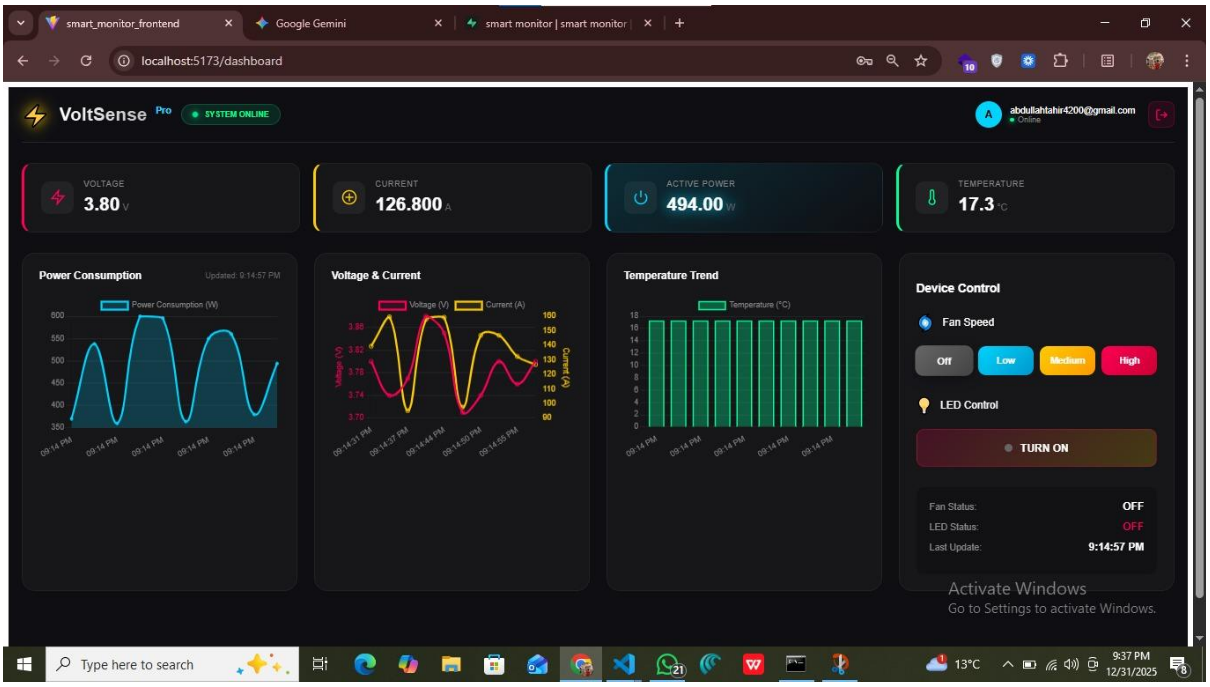
```
0 --port 8000 --reload
INFO:     Will watch for changes in these directories: ['G:\\5th semester\\embeded iot\\IOT_Project\\smart_monitor_backend
INFO:     127.0.0.1:63749 - "GET /readings/ HTTP/1.1" 200 OK
INFO:     127.0.0.1:63749 - "GET /readings/ HTTP/1.1" 200 OK
INFO:     127.0.0.1:63749 - "GET /readings/ HTTP/1.1" 200 OK
INFO:     127.0.0.1:63749 - "GET /readings/ HTTP/1.1" 200 OK
INFO:     127.0.0.1:63749 - "GET /readings/ HTTP/1.1" 200 OK
INFO:     127.0.0.1:63749 - "GET /readings/ HTTP/1.1" 200 OK
INFO:     127.0.0.1:63749 - "GET /readings/ HTTP/1.1" 200 OK
INFO:     127.0.0.1:63749 - "GET /readings/ HTTP/1.1" 200 OK
INFO:     127.0.0.1:63749 - "GET /readings/ HTTP/1.1" 200 OK
INFO:     127.0.0.1:63749 - "GET /readings/ HTTP/1.1" 200 OK
INFO:     127.0.0.1:63749 - "GET /readings/ HTTP/1.1" 200 OK
INFO:     127.0.0.1:63749 - "GET /readings/ HTTP/1.1" 200 OK
INFO:     127.0.0.1:63749 - "GET /readings/ HTTP/1.1" 200 OK
INFO:     127.0.0.1:63749 - "GET /readings/ HTTP/1.1" 200 OK
INFO:     127.0.0.1:63749 - "GET /readings/ HTTP/1.1" 200 OK
INFO:     127.0.0.1:63749 - "GET /readings/ HTTP/1.1" 200 OK
INFO:     127.0.0.1:63749 - "GET /readings/ HTTP/1.1" 200 OK
INFO:     127.0.0.1:63749 - "GET /readings/ HTTP/1.1" 200 OK
INFO:     127.0.0.1:63749 - "GET /readings/ HTTP/1.1" 200 OK
INFO:     127.0.0.1:63749 - "GET /readings/ HTTP/1.1" 200 OK
INFO:     127.0.0.1:63749 - "GET /readings/ HTTP/1.1" 200 OK
INFO:     127.0.0.1:63751 - "GET /readings/ HTTP/1.1" 200 OK
INFO:     127.0.0.1:63756 - "GET /readings/ HTTP/1.1" 200 OK
INFO:     127.0.0.1:63762 - "GET /readings/ HTTP/1.1" 200 OK
INFO:     127.0.0.1:63766 - "GET /readings/ HTTP/1.1" 200 OK
INFO:     127.0.0.1:63772 - "GET /readings/ HTTP/1.1" 200 OK
INFO:     127.0.0.1:63776 - "GET /readings/ HTTP/1.1" 200 OK
INFO:     127.0.0.1:63782 - "GET /readings/ HTTP/1.1" 200 OK
INFO:     127.0.0.1:63784 - "GET /readings/ HTTP/1.1" 200 OK
INFO:     127.0.0.1:63825 - "GET /readings/ HTTP/1.1" 200 OK
INFO:     127.0.0.1:63830 - "GET /readings/ HTTP/1.1" 200 OK
INFO:     127.0.0.1:63841 - "GET /readings/ HTTP/1.1" 200 OK
INFO:     127.0.0.1:63846 - "GET /readings/ HTTP/1.1" 200 OK
INFO:     127.0.0.1:63864 - "GET /readings/ HTTP/1.1" 200 OK
```

➢ Database



id int8	voltage float8	current float8	power float8	temperature float8	humid... flo...	source text
395	3.85	170.8	540	17.6	69.7	ESP32
396	3.75	109.3	538	17.6	69.7	ESP32
397	3.83	126.7	642	17.6	69.7	ESP32
398	3.73	141.9	364	17.6	69.7	ESP32
399	3.76	146.3	550	17.6	69.7	ESP32
400	3.76	140.2	370	17.7	69.7	ESP32
401	3.97	168.7	574	17.7	69.7	ESP32
402	3.72	149.8	570	17.7	69.7	ESP32
403	3.73	161.9	534	17.7	69.6	ESP32
404	3.74	94.9	358	17.7	69.6	ESP32
405	3.77	102.7	380	17.7	69.6	ESP32
406	3.79	138	508	17.7	69.6	ESP32

➢ Frontend



smart_monitor_frontend | Google Gemini | smart monitor | smart monitor

localhost:5173/dashboard

VoltSense Pro · SYSTEM ONLINE

abdullahtahir4200@gmail.com · Online

VOLTAGE 3.80 V

CURRENT 126.800 A

ACTIVE POWER 494.00 W

TEMPERATURE 17.3 °C

Power Consumption · Updated: 9:14:57 PM

Voltage & Current

Temperature Trend

Device Control

Fan Speed: Off · Low · Medium · High

LED Control: TURN ON

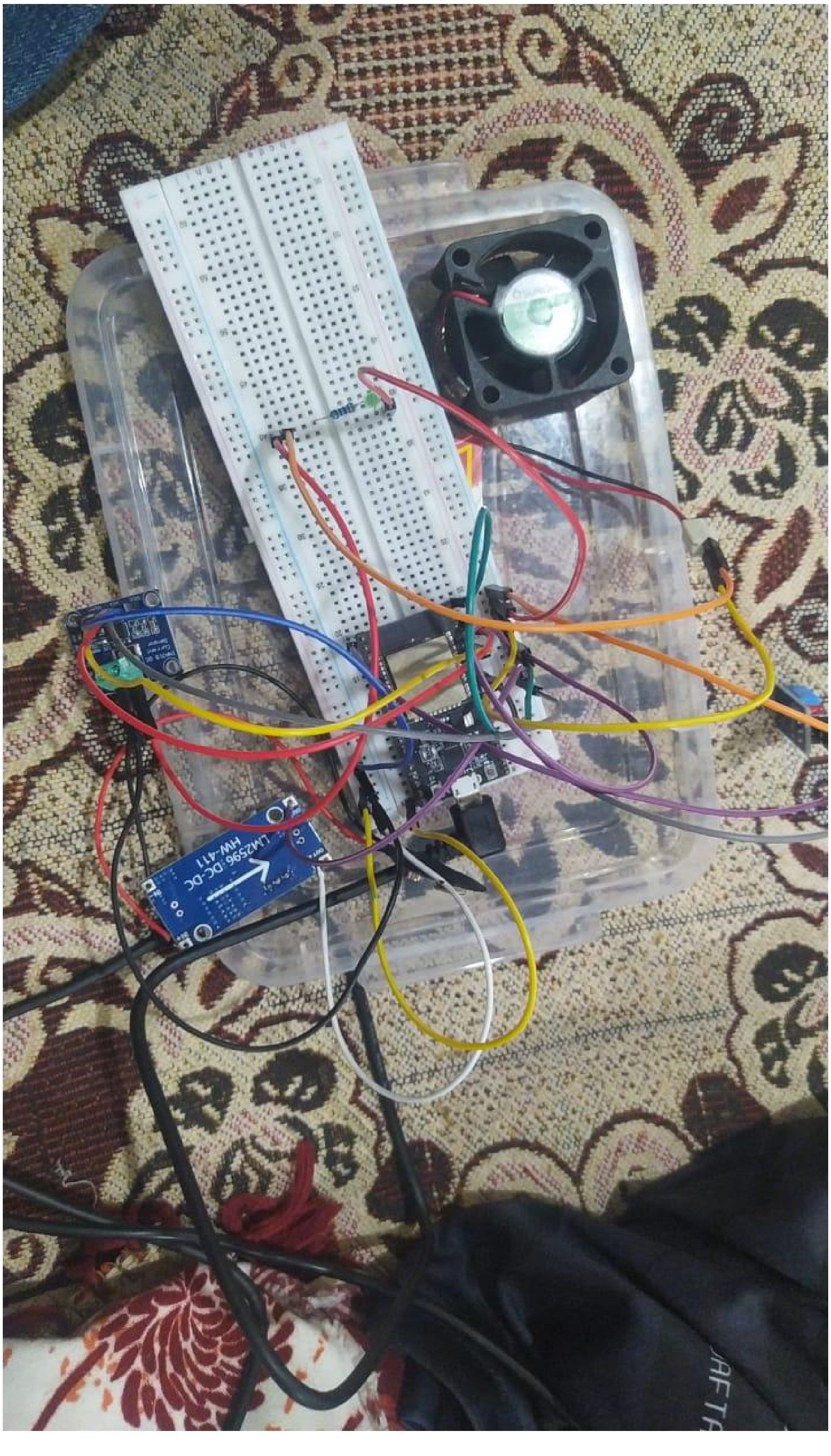
Fan Status: OFF

LED Status: OFF

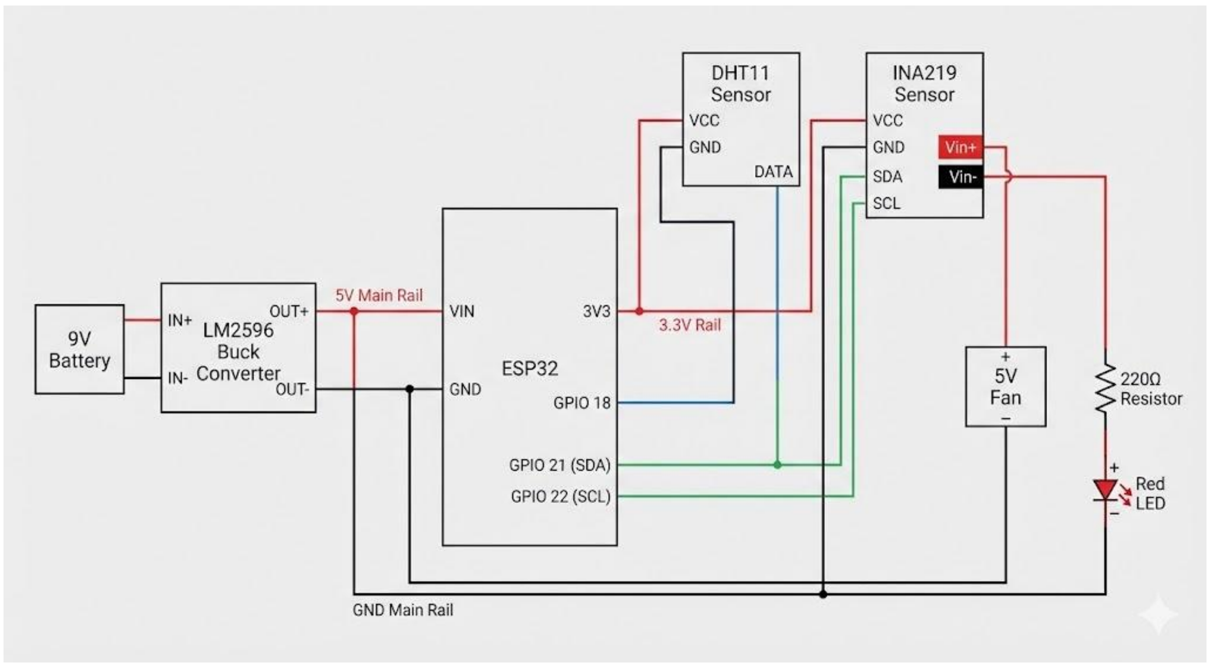
Last Update: 9:14:57 PM

Activate Windows
Go to Settings to activate Windows.

➢ Hardware


LM2596 DC-DC
HW-411

➢ *Circuit Diagram*


DHT11 Sensor
VCC
GND
DATA
INA219 Sensor
VCC
GND
SDA
SCL
Vin+
Vin-
9V Battery
IN+
IN-
LM2596 Buck Converter
OUT+
OUT-
5V Main Rail
VIN
GND
ESP32
3V3
3.3V Rail
GPIO 18
GPIO 21 (SDA)
GPIO 22 (SCL)
+
5V Fan
−
220Ω Resistor
+
Red LED
−
GND Main Rail