Name: Omji Verma

ID : 2024peb5066

Abstract

The *Temp Tracker* is an IoT-based weather monitoring station developed as a minor project for a Master's degree in Embedded Systems. This project aims to create an intelligent weather monitoring solution using NodeMCU (ESP8266) and an array of sensors to measure and display real-time environmental data, including temperature, humidity, atmospheric pressure, and battery status.

The system integrates a 0.94-inch OLED display to provide on-site data visualization, while the Blynk platform is used for remote monitoring through a user-friendly dashboard. Key features include real-time data acquisition, Wi-Fi connectivity for cloud integration, and automated sensor functionality checks. The project demonstrates the fusion of IoT technology with environmental monitoring, showcasing its potential for applications in smart homes, agriculture, and industrial environments.

By leveraging efficient sensor modules such as the DHT11, BMP280, and RTC DS3231, along with advanced microcontroller programming, the *Temp Tracker* delivers accurate and accessible weather data. The inclusion of battery monitoring and low-power design enhances its usability for portable and off-grid scenarios. This project represents a comprehensive approach to real-time environmental sensing, contributing to the growing field of smart embedded systems.

Table of Contents

4. Introduction

4.1 Background

Weather monitoring has become increasingly vital for various fields, including agriculture, environmental management, disaster preparedness, and smart home systems. With the advent of IoT, traditional weather monitoring systems have evolved into interconnected, intelligent devices that provide real-time environmental data accessible from anywhere. IoT-enabled weather monitoring systems not only offer convenience but also play a crucial role in data-driven decision-making for individuals and industries. The *Temp Tracker* leverages IoT technology to bring an efficient, user-friendly, and portable solution to real-time weather monitoring.

4.2 Objective

The primary goal of the *Temp Tracker* project is to design and implement an IoT-based weather monitoring system that:

- Measures real-time temperature, humidity, atmospheric pressure, and battery status.
- Displays this data on an OLED screen for local monitoring.
- Transmits the collected data to a Blynk dashboard for remote access and visualization.
  The project aims to enhance user experience through automated sensor checks, reliable performance, and a low-power design suitable for portable use.

4.3 Scope

The *Temp Tracker* system integrates advanced sensor modules and microcontroller programming to deliver accurate weather data. Its scope includes:

- Real-time monitoring of environmental conditions.
- Cloud integration using Blynk for remote data access.
- Applications in smart homes, agriculture, environmental research, and portable weather monitoring solutions.
  The system's modular design allows for future expansions, such as adding new sensors or advanced analytics.

---

5. System Design and Architecture

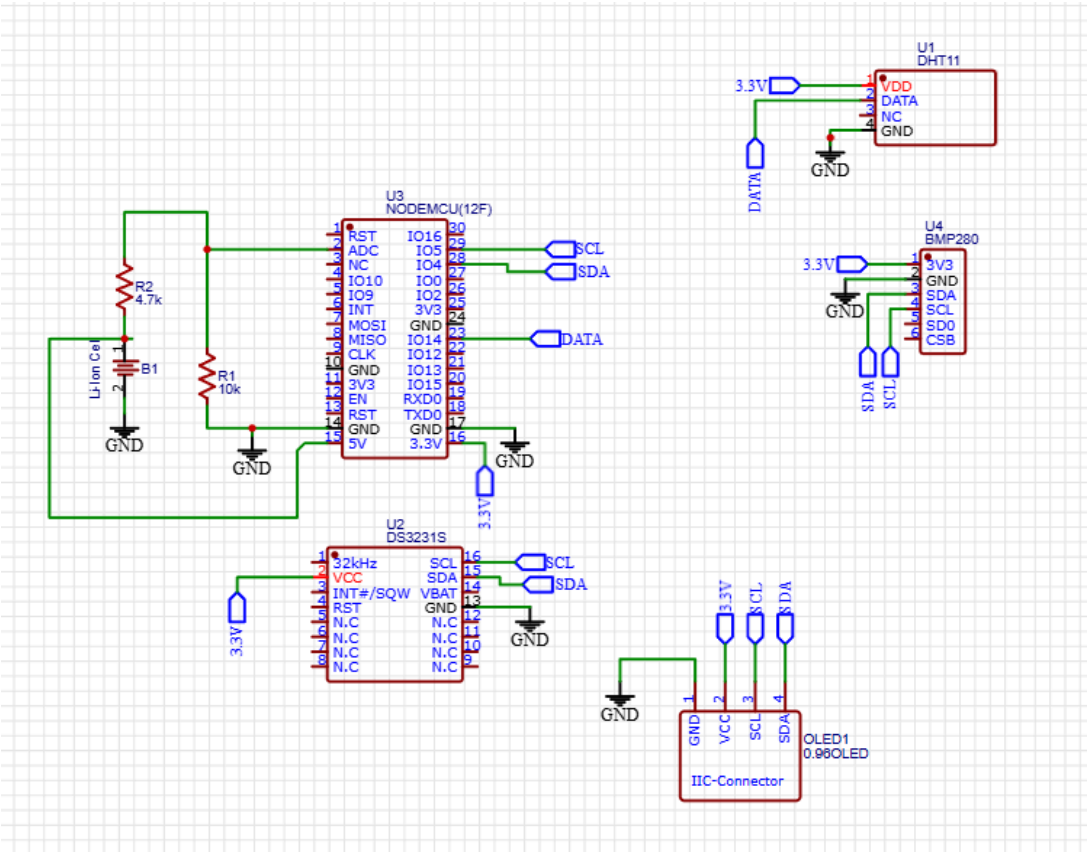
5.1 Component Overview

1. **NodeMCU (ESP8266)**
    - Features: Wi-Fi-enabled microcontroller with GPIO pins, low power consumption.
    - Role: Acts as the central processing unit, managing sensor data, display output, and cloud communication.
2. **DHT11 Sensor**
    - Specifications: Measures temperature (0–50°C) and humidity (20–90% RH) with moderate accuracy.
    - Role: Captures environmental temperature and humidity.
3. **BMP280 Sensor**
    - Specifications: Measures atmospheric pressure (300–1100 hPa) with high accuracy; supports altitude calculation.
    - Role: Provides barometric pressure readings for weather analysis.
4. **RTC DS3231**
    - Specifications: High-precision real-time clock module with battery backup.
    - Role: Keeps track of current time, enabling timestamped data logging.
5. **OLED Display (SSD1306)**
    - Specifications: 0.94-inch, 128x64 pixel resolution, I2C communication.
    - Role: Displays real-time weather data, Wi-Fi status, and battery percentage.
6. **Battery Monitoring Module**
    - **TP4056**: Manages Li-ion battery charging with overcharge protection.
    - **Voltage Divider**: Reduces battery voltage for ADC measurement on the NodeMCU.
    - Role: Enables portable operation and provides battery status updates.

5.2 Circuit Diagram

The circuit diagram illustrates the hardware connections between the components:

- **NodeMCU**
    - DHT11 connected to GPIO14 (D5).
    - BMP280 connected via I2C (SCL to GPIO5, SDA to GPIO4).
    - RTC DS3231 connected via I2C (SCL to GPIO5, SDA to GPIO4).
    - OLED Display connected via I2C (SCL to GPIO5, SDA to GPIO4).
    - Battery voltage divider connected to ADC (A0).


U1
DHT11
3.3V
VDD
DATA
NC
GND
GND
DATA
U4
BMP280
3.3V
3V3
GND
SDA
SCL
SD0
CSB
GND
SDA
SCL
U3
NODEMCU(12F)
RST
ADC
NC
IO10
IO9
INT
MOSI
MISO
CLK
GND
3V3
EN
RST
GND
5V
IO16
IO5
IO4
IO0
IO2
3V3
GND
IO14
IO12
IO13
IO15
RXD0
TXD0
GND
3.3V
SCL
SDA
DATA
R2
4.7k
R1
10k
B1
Li-Ion Cell
GND
GND
GND
3.3V
U2
DS3231S
32kHz
VCC
INT#/SQW
RST
N.C
N.C
N.C
N.C
SCL
SDA
VBAT
GND
N.C
N.C
N.C
N.C
SCL
SDA
3.3V
GND
3.3V
SCL
SDA
GND
VCC
SCL
SDA
IIC-Connector
OLED1
0.96OLED

6. Software Design

6.1 Development Environment

The software for the *Temp Tracker* system was developed using the **Arduino IDE**. The following tools and libraries were employed:

- **Arduino IDE Version**: 1.8.19 (or newer)
- **Required Libraries**:
    - Blynk (v1.0.1)
    - ESP8266WiFi (v2.7.4)
    - Adafruit SSD1306 (v2.5.7)
    - Adafruit GFX (v1.11.3)
    - DHT Sensor Library (v1.4.2)
    - Adafruit BMP280 (v2.1.0)
    - RTClib (v1.13.0)
    - Battery Sense

6.2 Software Architecture

The software is modular and follows a structured flow:

- **Setup Function**:
    - Initializes hardware components (sensors, display, RTC).
    - Connects to Wi-Fi and the Blynk cloud platform.
    - Runs a startup animation and sensor test.
- **Loop Function**:
    - Continuously reads sensor data (temperature, humidity, pressure).
    - Processes data for display and transmission.
    - Displays information on the OLED.
    - Sends data to the Blynk dashboard.
- **Core Modules**:
    - **Sensor Initialization**: Ensures all sensors are functioning properly.
    - **Data Processing**: Applies averaging and error-handling techniques for reliable readings.
    - **Output**: Updates OLED and sends data packets to Blynk.

6.3 Library Details

1. **Blynk**:
    - Used for IoT integration to send and receive data between the device and the Blynk dashboard.

2. **ESP8266WiFi**:
    - Manages Wi-Fi connectivity and ensures reliable data transmission.
3. **Adafruit SSD1306**:
    - Controls the OLED display for rendering text and graphical data.
4. **Adafruit GFX**:
    - Provides graphical rendering functions, such as drawing shapes and text formatting.
5. **DHT Sensor Library**:
    - Interfaces with the DHT11 sensor to fetch temperature and humidity values.
6. **Adafruit BMP280**:
    - Handles communication with the BMP280 sensor to acquire barometric pressure readings.
7. **RTClib**:
    - Manages the RTC DS3231 module for timekeeping functionalities.
8. **Battery Sense**:
    - Processes voltage readings from the battery monitoring module to calculate the charge percentage.

6.4 Algorithm

1. **Setup Phase**:
    - Initialize OLED display and sensors.
    - Connect to Wi-Fi and Blynk.
    - Run the `sensorTest()` function to verify sensor operations.
2. **Data Acquisition**:
    - Read temperature and humidity from DHT11.
    - Fetch pressure data from BMP280.
    - Get time from RTC DS3231.
    - Measure battery voltage via ADC and calculate percentage.
3. **Data Processing**:
    - Average readings over five samples to minimize noise.
    - Format data for display and cloud transmission.
4. **Display and Transmission**:
    - Update OLED screen with weather data, time, and Wi-Fi status.
    - Send data to Blynk for remote monitoring.

6.5 Key Functions

1. **Sensor Initialization**:
    - Initializes and verifies the functionality of DHT11, BMP280, RTC, and Battery Sense modules.

2. **Data Display on OLED**:
    - Formats and displays temperature, humidity, pressure, battery percentage, and time on the OLED.
3. **Wi-Fi and Blynk Integration**:
    - Connects to Wi-Fi and syncs with the Blynk server to transmit sensor data.
4. **Sensor Testing**:
    - Executes self-check routines for all connected sensors and displays results during startup.

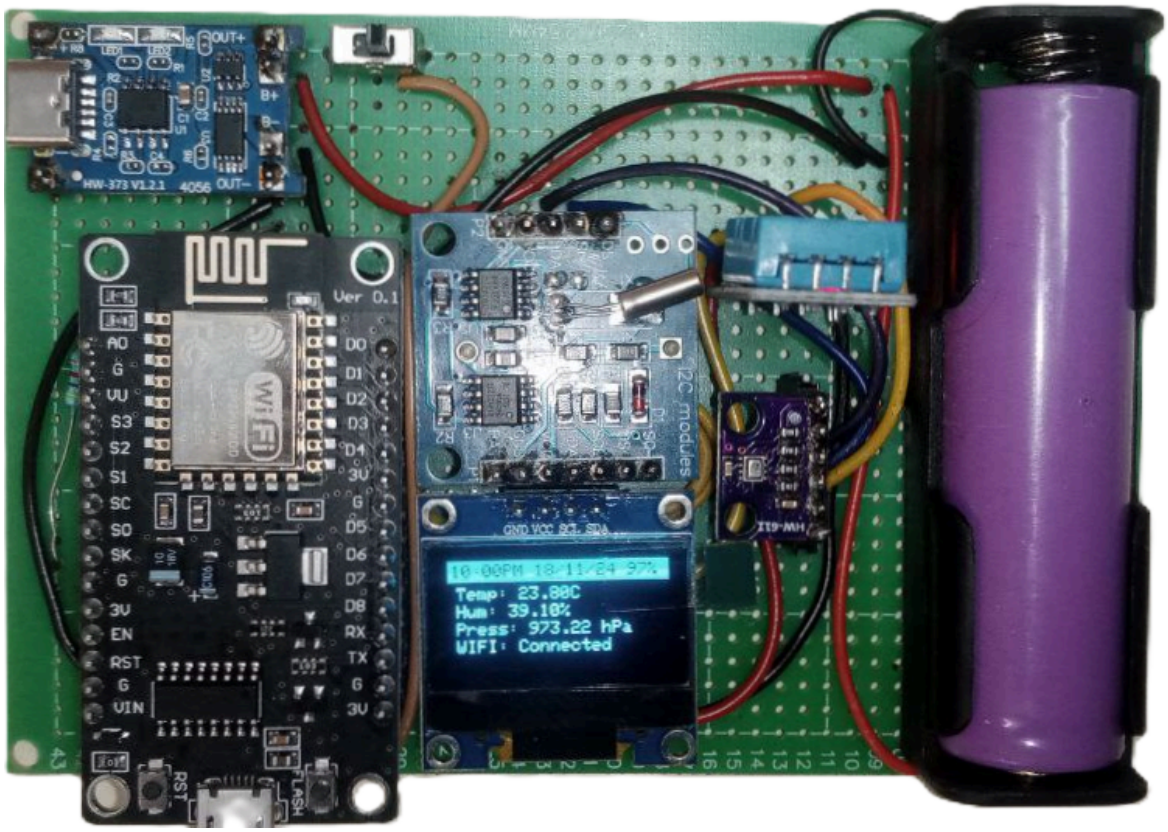
7. Hardware Implementation

7.1 Hardware Requirements

Component	Specifications
NodeMCU (ESP8266)	Wi-Fi-enabled microcontroller
DHT11	Temperature and humidity sensor
BMP280	Barometric pressure sensor
RTC DS3231	Real-time clock module
OLED Display (SSD1306)	0.94-inch, 128x64 pixels, I2C
Li-ion Battery	3.7V, 18650 type
TP4056	Battery charging module
Resistors	10kΩ, 4.7kΩ
Wires and Breadboard	For connections

7.2 Assembly and Connections

1. **NodeMCU Wiring**:
    - Connect sensors, OLED, and battery module to corresponding GPIO pins as per the circuit diagram.
2. **Sensor Connections**:
    - DHT11: VCC to 3V3, GND to GND, Data to D5 (GPIO14).
    - BMP280: VCC to 3V3, GND to GND, SCL to D1 (GPIO5), SDA to D2 (GPIO4).
    - RTC DS3231: VCC to 3V3, GND to GND, SCL to D1 (GPIO5), SDA to D2 (GPIO4).
3. **OLED Connection**:
    - VCC to 3V3, GND to GND, SCL to D1 (GPIO5), SDA to D2 (GPIO4).
4. **Battery Module**:
    - Connect the TP4056 module to the battery and NodeMCU. Use a voltage divider to scale the battery voltage for ADC input.
5. **Assembly**:
    - Used a PCB for structured assembly. Ensure connections are firm and solder if required.
6. **Testing**:
    - Verify power delivery to all components.
    - Check functionality of each sensor before running the complete system.



8. Blynk Dashboard Setup

8.1 Account Creation and Configuration

1. Download the **Blynk IoT** app from the Play Store or App Store.
2. Create a Blynk account or log in if you already have one.
3. Access the **Blynk Web Dashboard** via blynk.cloud.

8.2 Creating the Device

1. **Add New Device**:
    - Navigate to **Devices** > **Add New Device**.
    - Select **ESP8266** as the hardware type.
    - Choose **Wi-Fi** as the connection method.
2. **Blynk Template**:
    - Generate a template with a unique **Template ID**, **Template Name**, and **Auth Token**.
    - Copy these details and update them in your Arduino code.

8.3 Data Streams

1. Go to the **Data Streams** section in the Blynk dashboard.
2. Add the following virtual data streams:

Data	Type	Virtual Pin	Units
Temperature	Virtual	V0	°C
Humidity	Virtual	V1	%
Atmospheric Pressure	Virtual	V2	hPa
Battery Percentage	Virtual	V3	%

3. Configure each data stream with the required range, units, and update interval.

8.4 Dashboard Design

1. Navigate to the **Web Dashboard** tab.
2. Add widgets for the data streams:

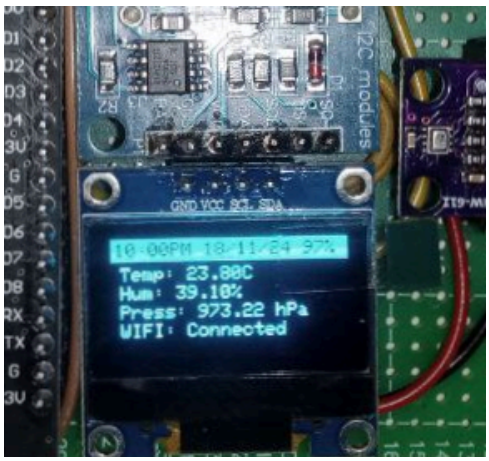
Widget	Data Source	Settings
Gauge	Temperature (V0)	Range: 0–50°C, Units: °C
Label	Humidity (V1)	Units: %
Chart	Atmospheric Pressure (V2)	Range: 900–1100 hPa
Battery Icon	Battery Percentage (V3)	Units: %

3. Arrange the widgets for easy visualization and save the layout.

9. Results and Discussion

9.1 Real-Time Weather Monitoring

- The OLED display shows real-time data for:
    - Temperature, Humidity, Pressure.
    - Battery percentage and Wi-Fi status.
    - Current time in AM/PM format.



9.2 Blynk Dashboard Monitoring

- The Blynk dashboard provides remote visualization of the collected data.
- Examples include:
    - Temperature variation over time (line graph).
    - Real-time battery percentage updates.

9.3 Sensor Accuracy and Performance

- Discuss the accuracy of DHT11 and BMP280 readings compared to standard values.
- Provide a summary of any deviations and the possible reasons for them.
- Highlight the reliability of the DS3231 RTC module in maintaining time.

9.4 Power Consumption Analysis

- Analyze the battery discharge rate over a 24-hour period.
- Discuss the effectiveness of the TP4056 module in recharging the Li-ion battery.

10. Challenges and Troubleshooting

10.1 Wi-Fi Connection Issues

- Common Issue: Inability to connect to Wi-Fi.
- **Solution**: Verify SSID and password in the code. Check the signal strength and ensure no MAC filtering is enabled on the router.

10.2 Sensor Malfunctions

- Common Issue: Sensors not responding or providing incorrect values.
- **Solution**:
    - Verify wiring and connections.
    - Check for sensor-specific library compatibility.

10.3 Debugging Techniques

- Use the **Serial Monitor** in Arduino IDE for real-time debugging.
- Examples of common errors and their solutions:
    - Error in sensor initialization: Ensure correct pin mappings in the code.
    - Incorrect Blynk Auth Token: Recheck and update in the Arduino code.