NATIONAL UNIVERSITY OF SCIENCE AND TECHNOLOGY

School of Electrical Engineering and Computer Science (SEECS)

INTERNET OF THINGS

SEMESTER PROJECT

Name:	Nimra Arif	Minahil Shakoor	Zohair Shakeel	Mariyam Abid	Shahmeer Sheraz
CMS ID:	396778	372373	372011	372835	376281
Class:	BESE 12-B				
Semester	7th				

SUBMITTED TO: **DR. RAFIA MUMTAZ**

DATE OF SUBMISSION: **23RD DECEMBER 2024**

TABLE OF CONTENTS

Table of Figures

1. INTRODUCTION

In recent years, the impact of climate change has become more apparent, with rapidly changing weather patterns affecting daily life in urban and rural areas. Monitoring environmental conditions has therefore become essential to understand the condition of our ecosystems and efficiently adapting to changes. This project presents the development of a smart weather monitoring system that tracks temperature, humidity, and light intensity. The system uses an ESP32 microcontroller, DHT-22 sensor for temperature and humidity, LDR for light intensity measurement, and a passive buzzer for alerting users to abnormal conditions. The data collected is sent to the ThingSpeak Cloud, where it can be accessed remotely, making the system ideal for local weather monitoring in urban or rural environments.

2. MOTIVATION

The motivation for developing this smart weather monitoring system was to address climate change and its consequences on local settings is becoming increasingly urgent. It is essential to have an accessible and affordable system for monitoring environmental factors in places like Lahore, which are severely affected by smog and harsh weather. By offering important insights into regional environmental trends, real-time weather data can help lessen the negative effects of climate change.

By providing a low-cost and simple-to-implement solution, this initiative aims to close that gap and guarantee that populations in both urban and rural areas have access to vital weather data. Furthermore, this system's interaction with the ThingSpeak Cloud platform enables real-time environmental condition monitoring without the need for complicated infrastructure. This initiative is to increase awareness of the significance of climate adaptation and enable local organizations and individuals to take proactive steps based on reliable data.

3. WHY THIS PROJECT

- **Real-Time Data Collection**: The system makes it possible to continuously monitor environmental parameters like light intensity, humidity, and temperature. Making prompt judgements, including when to take precautions against excessive heat or humidity, requires real-time data.
- **Accessible and Inexpensive**: This method provides a cost-effective substitute for costly professional weather stations, guaranteeing that more individuals can keep an eye on the local weather.
- **Climate Awareness**: By offering localized environmental data, this technology supports global efforts to track and comprehend the effects of climate change. This is particularly crucial in places where smog or erratic weather patterns are prevalent, as the system can monitor changes and issue early warnings.
- **Cloud Integration:** The system makes it simple for users to access data from anywhere with an internet connection by transmitting it to the ThingSpeak Cloud. The cloud-based platform is perfect for long-term monitoring since it enables remote access, visualization, and analysis of meteorological data.
- **Environmental Monitoring for Small-Scale Use:** This system offers a cost-effective method for people or small organizations in places without access to costly weather stations to keep an eye on temperature, humidity, and light levels—all of which are important markers of the surrounding environment.

4. BACKGROUND AND EXISTING PROJECTS

Because of issues like natural catastrophes, urbanization, and climate change, weather monitoring is essential. Urban planning, disaster relief, and agriculture are all aided by real-time weather data. Despite their effectiveness, traditional methods lack regional data collecting and are expensive.

By combining sensors, cloud computing, and wireless connection, IoT has revolutionized weather monitoring by providing scalable, affordable, and real-time solutions. In order to create a Smart Weather Monitoring System that gathers information on temperature, humidity, air pressure, and light intensity and stores and visualizes it on a cloud platform for real-time updates, this project makes use of the Internet of Things.

EXISTING PROJECTS

Several projects and systems currently exist in the domain of IoT-based weather monitoring. Notable examples include:

WEATHER UNDERGROUND (WU) PERSONAL WEATHER STATIONS

People may install their own weather stations on Weather Underground to add local data to their network. These systems gather, process, and disseminate environmental data in real time using a mix of proprietary hardware and software.

Figure 1: Weather Underground Personal Weather Station

ARDUINO-BASED WEATHER MONITORING SYSTEMS

Arduino microcontrollers are used in many do-it-yourself Internet of Things weather projects in conjunction with sensors such as the DHT22 (temperature and humidity) and BMP180 (pressure). These initiatives prioritize flexibility and cost for small-scale deployments.

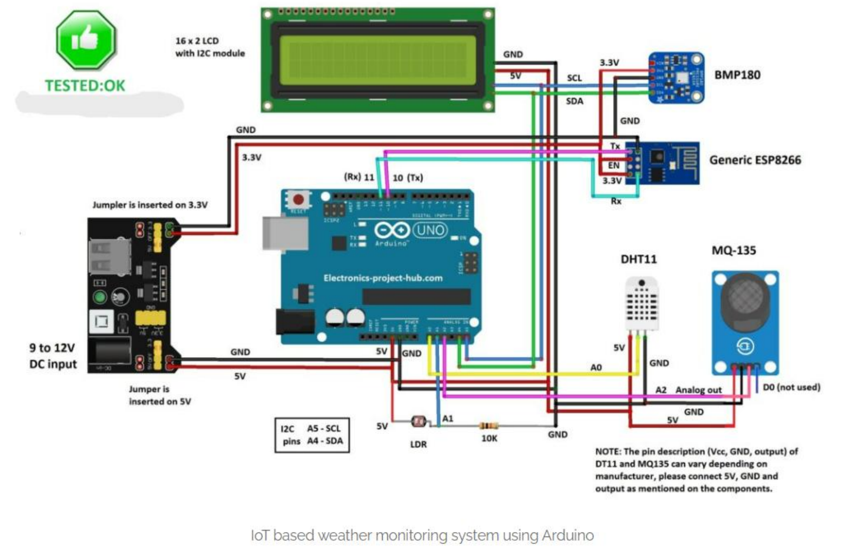


Figure 2: Arduino based weather monitoring system

SMART CITIES WEATHER STATIONS

Governments and private organizations have deployed smart weather stations in urban areas to monitor air quality, temperature, and humidity. These systems are integrated with city infrastructure for predictive analytics and environmental management.

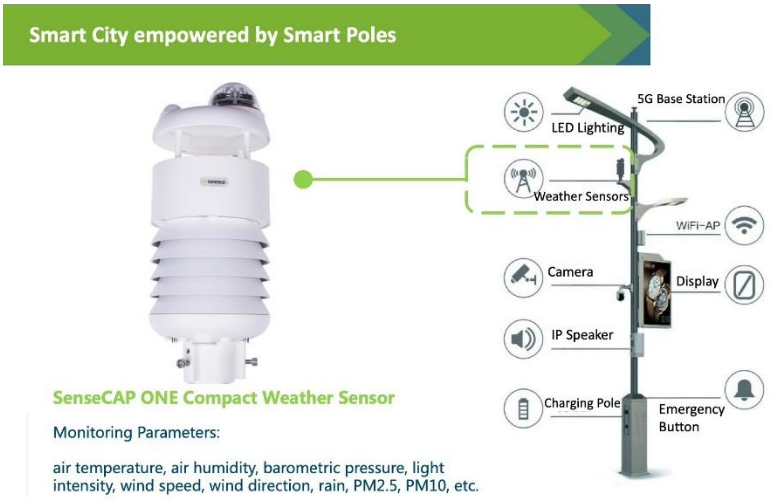


Figure 3:Smart Cities Weather Station

DAVIS VANTAGE PRO2 WEATHER STATIONS:

A premium weather monitoring system with cutting-edge sensors to gather information on temperature, wind speed, and precipitation. It provides both internet and local data access, making it appropriate for both personal and business use.

Figure 4: Davis Vantage Pro2 Weather Stations

NETATMO WEATHER STATION:

A consumer-friendly IoT device that monitors indoor and outdoor environmental conditions, integrating with smartphones for real-time updates and alerts.

Figure 5: Netatmo Weather Station:

5. SYSTEM FEATURES

The smart weather monitoring system developed in this project boasts several key features:

- **Real-Time Environmental Monitoring**: Temperature, humidity, and light intensity are just a few of the vital meteorological characteristics that the system continuously monitors. The ThingSpeak dashboard allows for real-time analysis of these readings, which are recorded at regular intervals.
- **ESP32 microcontroller:** The system's central component, the ESP32 microcontroller, offers Wi-Fi connectivity to facilitate smooth system-to-cloud communication. It manages the gathering, processing, and transmission of data and acts as an interface between the sensors and the ThingSpeak platform.
- **Cloud Integration:** Data collected by the system is sent to the ThingSpeak Cloud, where it is stored and can be accessed remotely. ThingSpeak also provides data visualization tools, enabling users to view historical data, analyze trends, and receive notifications when predefined thresholds are reached.
- **Passive Buzzer Alerts:** A passive buzzer is used to provide audible alerts when certain environmental conditions exceed thresholds, such as high temperature or high humidity feature ensures that users are alerted to potentially harmful conditions in real time.
- **Low Power Consumption**: Due to the system’s low power usage, allowing it to run continuously without excessive energy consumption. This is particularly beneficial for long-term deployment in remote areas or for portable applications.

6. PROPOSED METHODOLOGY

This project implements a smart IoT weather monitoring system using an ESP32 microcontroller, various sensors, and the ThingSpeak cloud platform. All development, testing, and analysis were conducted within the Wokwi online simulator environment. The methodology employed can be broken down into the following stages:

1. Virtual Hardware Setup and Interfacing (Wokwi Simulation):

- An ESP32 development board was used as the virtual processing unit within the Wokwi simulator.
- A DHT-22 temperature and humidity sensor was virtually interfaced with the ESP32. The sensor's data pin was connected to GPIO15 of the ESP32 within the simulation, with virtual power and ground connections.
- A virtual photoresistor (LDR) was connected to analog pin GPIO32 to simulate ambient light level measurements. A virtual voltage divider circuit was implemented within the simulator.
- A virtual potentiometer was connected to analog pin GPIO35 to provide a variable input for testing and calibration within the simulated environment.
- A virtual LCD1602 display with an I2C interface was connected to the ESP32's I2C pins (GPIO21 and GPIO22) within the simulator to display real-time simulated sensor readings.
- A virtual LED was connected to GPIO13 to indicate simulated WiFi connection status.
- A virtual buzzer was connected to GPIO14 to provide simulated audible alerts based on predefined temperature and humidity thresholds.
- All components were connected on a virtual breadboard within the Wokwi simulator. The virtual wiring schematic is defined in the project's JSON configuration file (Appendix A), which accurately reflects the simulated connections.

2. Software Development and Programming:

- The Arduino IDE was used for programming the ESP32.
- Libraries compatible with the DHT-22 sensor (DHT.h or similar) were used to interface with the virtual DHT-22 sensor in the simulator. It's important to note that the DHT-22 has lower accuracy and range compared to the DHT22.
- The LiquidCrystal_I2C library was used to control the virtual LCD display.
- The WiFi library was used to simulate a connection to a WiFi network within the simulator. Note that actual network connectivity is not established in the simulation environment.
- The ThingSpeak library was used to simulate sending sensor data to the ThingSpeak cloud platform. While the data is sent using the library calls, the actual transmission occurs within the Wokwi environment, and data is reflected on the ThingSpeak platform as if it were a real transmission.
- The code was structured to perform the following simulated functions:

- Initialize all virtual hardware components.
- Simulate connecting to a WiFi network.
- Read simulated temperature and humidity data from the virtual DHT-22 sensor.
- Read simulated ambient light levels from the virtual LDR.
- Map the analog readings from the virtual potentiometer and LDR.
- Display simulated sensor readings on the virtual LCD.
- Simulate sending sensor data to ThingSpeak at regular intervals.
- Activate the virtual buzzer if simulated temperature or humidity exceeds predefined thresholds.

3. Data Acquisition and Visualization (Simulated):

- Simulated sensor data was sent to the ThingSpeak cloud platform via the simulated network connection within Wokwi.
- ThingSpeak channels were configured to store and visualize the simulated data, demonstrating the intended functionality of the system in a real-world deployment.

4. Testing and Validation (Within Wokwi):

- The system was thoroughly tested within the Wokwi simulator to verify its simulated functionality and data flow.
- Simulated sensor readings were observed and compared with expected values to ensure correct simulated operation.
- The simulated data displayed on the virtual LCD and the simulated data uploaded to ThingSpeak were cross-referenced.

7. CIRCUIT DIAGRAM

The system begins with the initialization of all connected components, including the ESP32 microcontroller, DHT22 sensor for temperature and humidity, potentiometer, LDR for light sensing, LCD for data display, and the buzzer for alerts. The ESP32 establishes a WiFi connection to enable data transmission to the ThingSpeak platform, with the onboard LED blinking during connection attempts and staying solid once connected.

In operation, the DHT22 sensor measures temperature and humidity, while the potentiometer provides an analog value mapped to a percentage. The LDR detects ambient light levels, classifying them into categories like Sunny, Partly Cloudy, Overcast, or Dark. These readings are processed and checked against predefined thresholds for temperature and humidity to determine if warnings are needed.

Real-time data is displayed on the LCD, showing temperature, humidity, and the light condition. If the temperature exceeds or humidity drops below the set thresholds, the buzzer activates to alert users. The ESP32 also sends the collected data, including temperature, humidity, light condition code, and potentiometer value, to ThingSpeak for remote monitoring. Simultaneously, all data is logged on the Serial Monitor for debugging.

The system continuously loops, updating sensor readings, refreshing the LCD, transmitting data to ThingSpeak, and activating alerts when necessary, ensuring seamless environmental monitoring and user notifications.

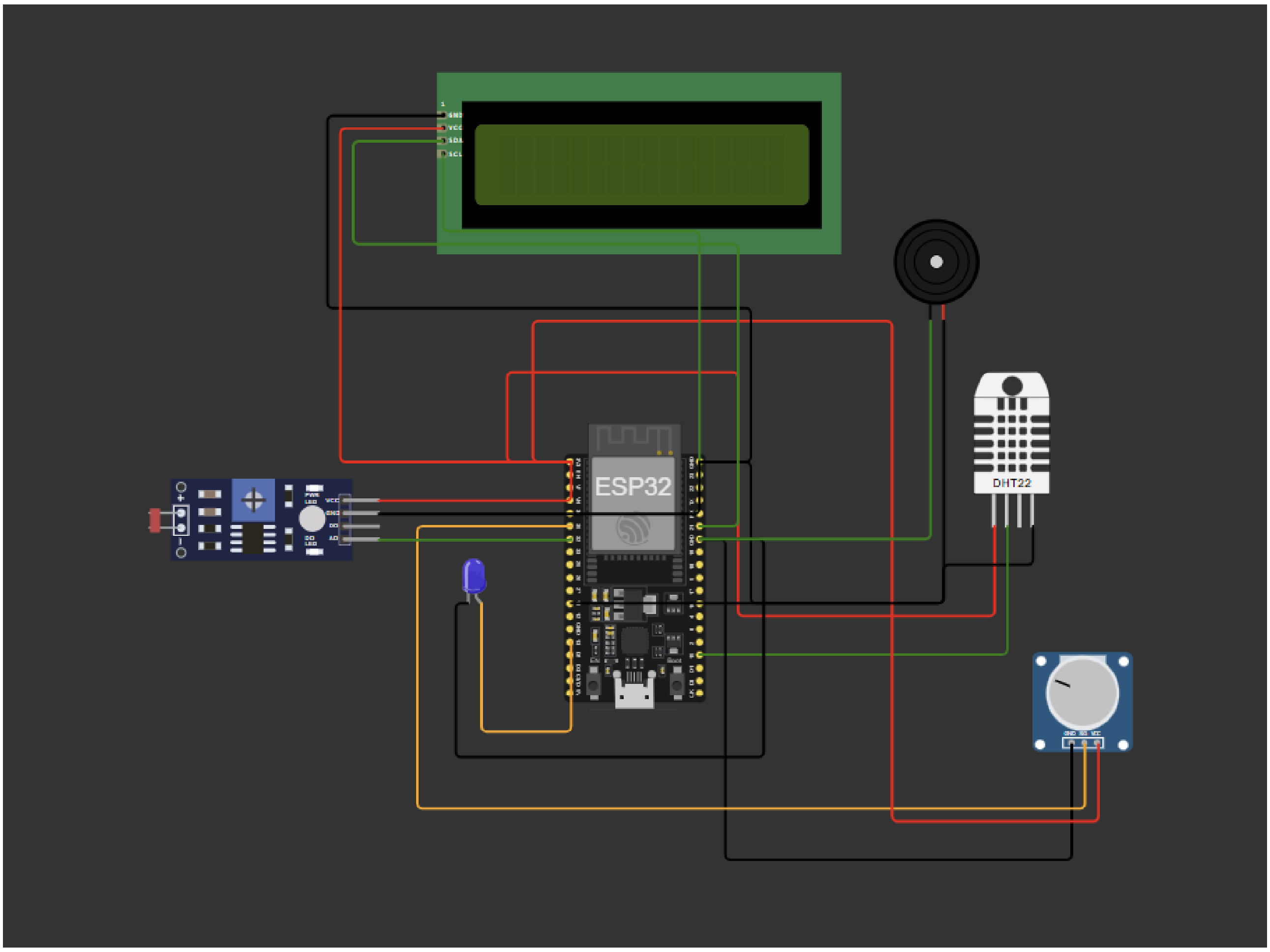


Figure 6: Circuit Diagram

8. DASHBOARD

We have utilized ThingSpeak to create a web-based cloud dashboard for our IoT project, which retrieves data via Wi-Fi through the ESP32. The dashboard displays four graphs: Temperature, Humidity, Light Intensity (derived from LDR readings), and Potential. Each graph's axes are well-labeled and appropriately divided to accommodate the minimum and maximum readings for each sensor output, which are clearly displayed.

Additionally, the dashboard features five unique indicators, each representing a specific weather condition based on the light intensity throughout the day.

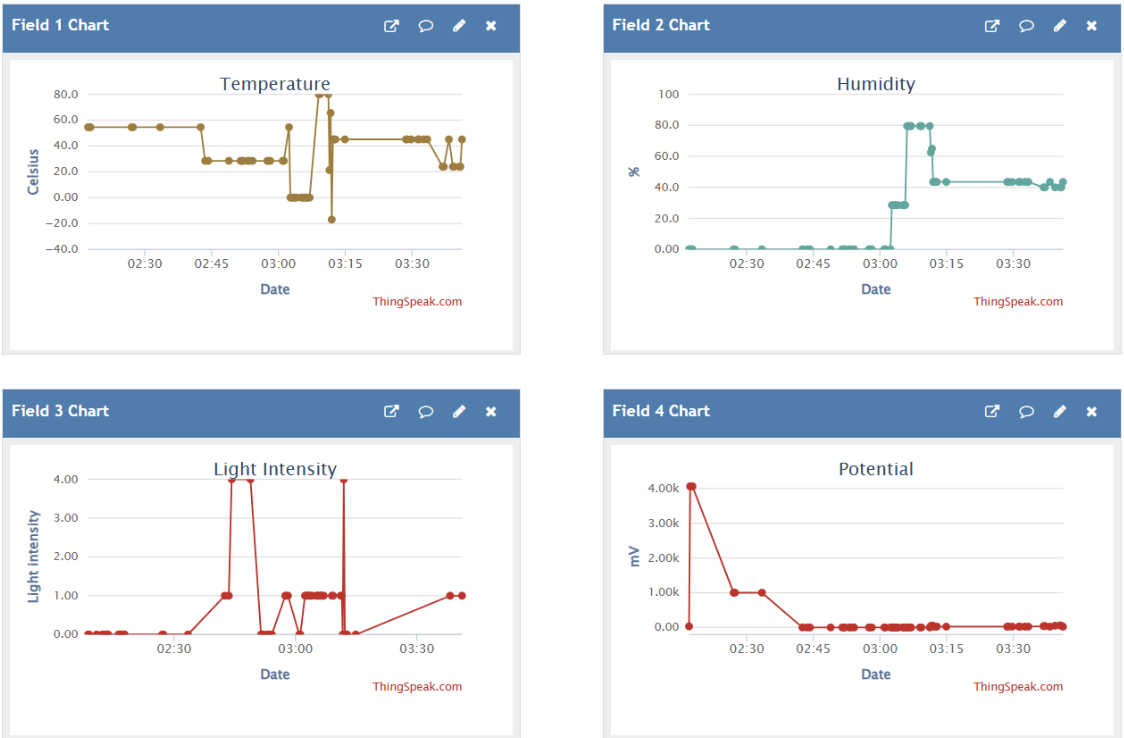


Figure 7: Dashboard Graphs

TEMPERATURE AND HUMIDITY READING

We have used the DHT-22 sensor (available in the Wokwi Simulator) to measure both temperature and humidity. The DHT-22 is a low-cost digital sensor commonly used for basic environmental monitoring.

For temperature, the DHT-22 sensor has a measuring range of -40°C to 80°C, with an accuracy of ±2°C. Its minimum temperature reading is 0°C, and the maximum is 50°C.

For humidity, the DHT-22 sensor has a range of 20% to 90% relative humidity, with an accuracy of ±5% RH. Its minimum humidity reading is 20%, and the maximum is 90%.This sensor is ideal for applications where moderate accuracy and a limited range are sufficient.

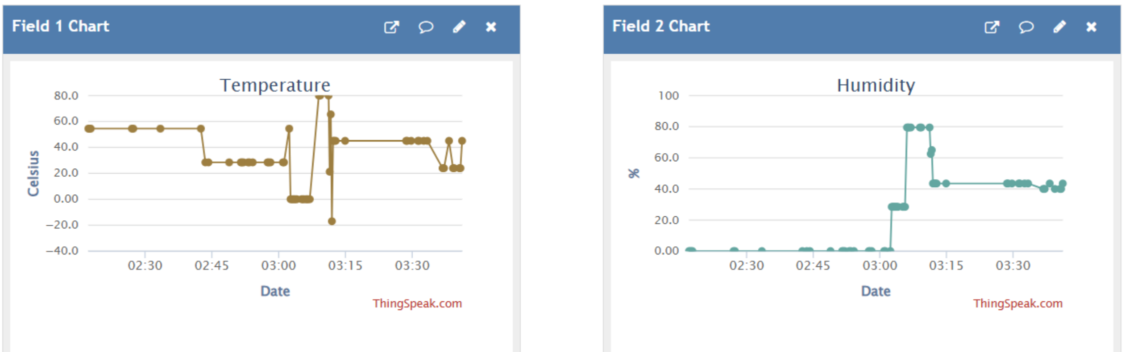


Figure 8: Temperature and Humidity Graph

LIGHT INTENSITY

The light intensity is calculated as a derivative of LDR value. The LDR is exposed to light varying form 0.1 to 100,000 lux. The resistance to this light generated inversely by the LDR is used to divide the light intensity in 5 distinct light conditions of a day: namely sunny, partially cloudy, overcast/cloudy, rain/fog, dark (night)

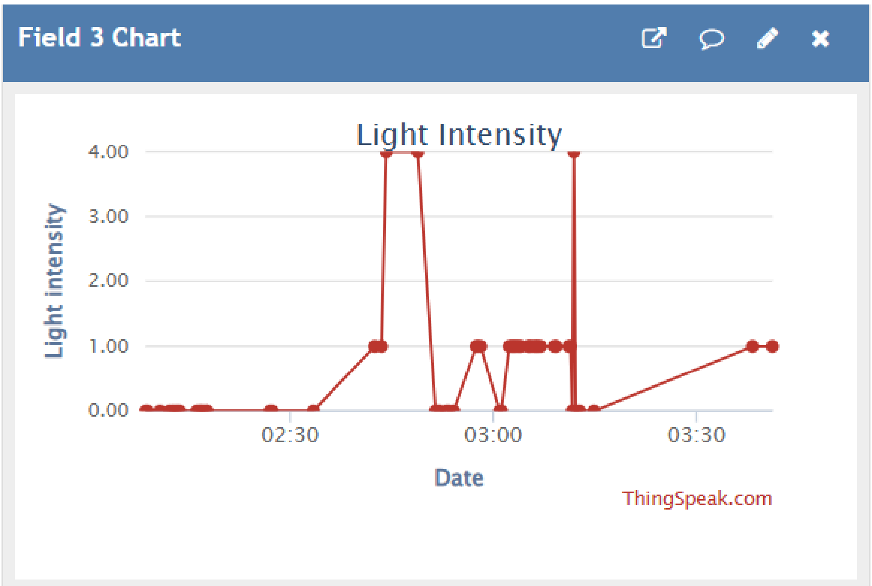


Figure 9: Light Intensity Graph

The following indicators are associated with the LDR and are lit up indicating that the weather is sunny, Partly cloudy, cloudy, foggy, or dark. There is a room to add more indicators too, like breaking the sunny indicator into further bright sunny or normal sunny but for now we have used just these indicators

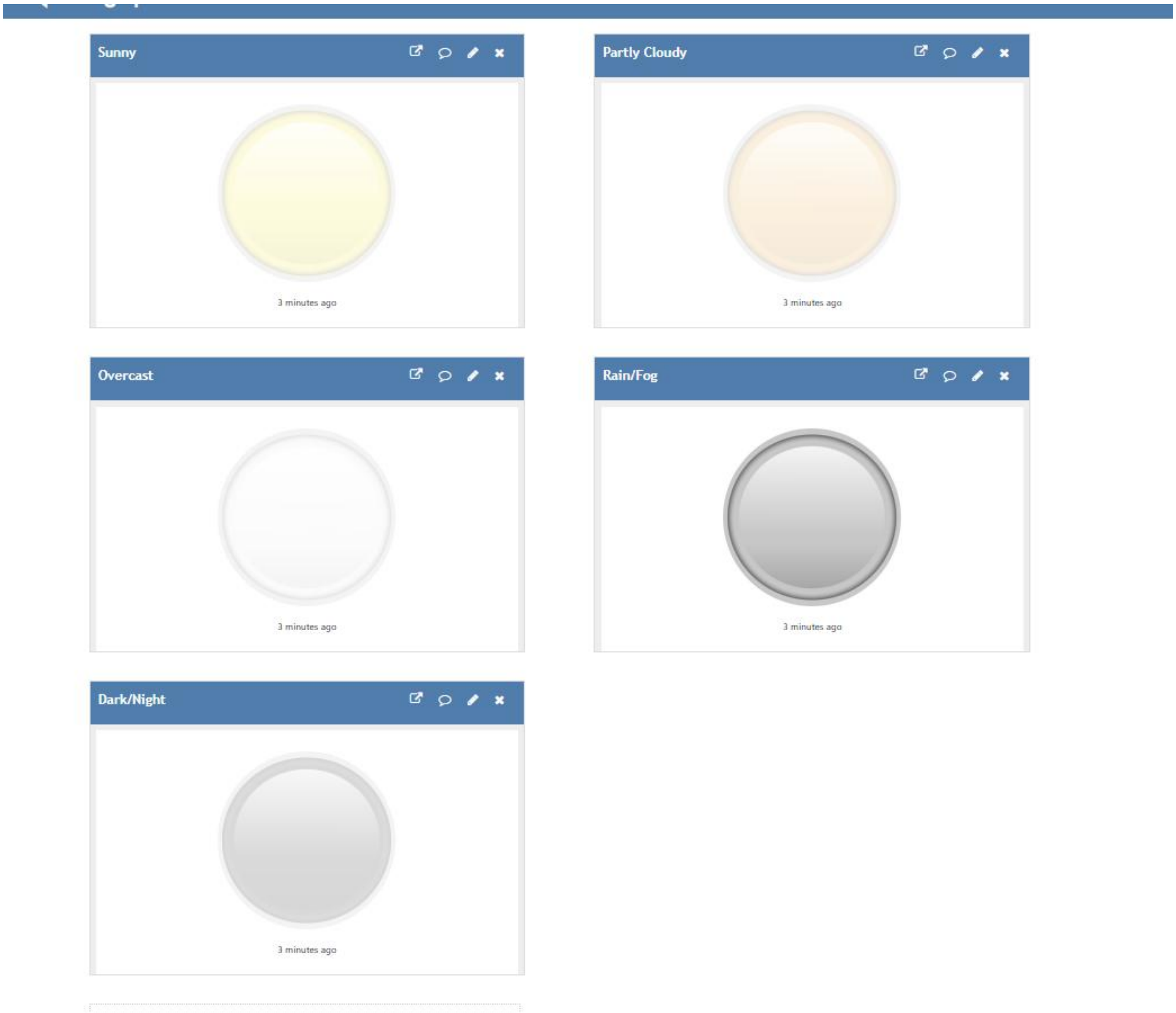


Figure 10: Indicators for the light intensity

9. COMPONENTS USED

There are multiple components used to make this simple yet useful weather monitoring system. We have divided the components into 4 categories, namely microcontroller, sensors, actuator, and miscellaneous.

MICROCONTROLLER

ESP32 (Microcontroller):

- **Usage**: Acts as the central unit of the system, managing data collection, processing, and communication with ThingSpeak.
- **Connections**: It interfaces with all sensors and actuators, including the DHT22 sensor, LDR, Buzzer, LED, Potentiometer, and LCD.

ESP32 Simulation

The ESP32 is a popular WiFi and Bluetooth-enabled microcontroller, widely used for IoT Projects. ESP32-C3, ESP32-S2, ESP32-S3, ESP32-C6, ESP32-H2, and ESP32-P4 (beta).

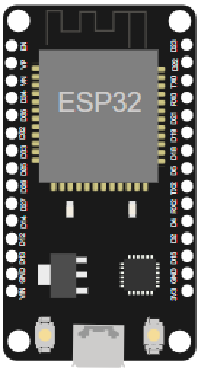


Figure 11: ESP32 From Wokwi

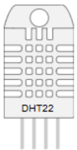
SENSORS

DHT22 (Temperature and Humidity Sensor):

- **Usage**: Measures the temperature and humidity of the environment.
- **Connections**:
    - VCC to 3.3V on ESP32
    - GND to GND on ESP32
    - Data pin (SDA) to pin 15 on ESP32

wokwi-dht22 Reference

Digital Humidity and Temperature sensor.



Pin names

Name	Description
VCC	Positive voltage
SDA	Digital data pin (input/output)
NC	Not connected
GND	Ground

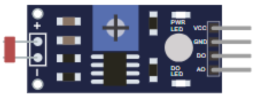
Figure 12: dht22 From Wokwi

LDR (Light Dependent Resistor):

- **Usage**: Measures the ambient light levels.
- **Connections**:
    - VCC to 3.3V on ESP32
    - GND to GND on ESP32
    - Analog output (AO) to pin 32 on ESP32

wokwi-photoresistor-sensor

Photoresistor (LDR) sensor module



Pin names

Name	Description
VCC	Positive power supply
GND	Ground
DO	Digital output
AO	Analog output

Figure 13: Photoresistor LDR from Wokwi

ACTUATOR

Buzzer:

- **Usage**: Emits sound for notifications when temperature and humidity exceed threshold.
- **Connections**:
  - One terminal to GND on ESP32
  - Other terminal to pin 14 on ESP32

wokwi-buzzer Reference

A piezoelectric buzzer

Pin names

Name	Description
1	Negative(Black) pin
2	Positive(Red) pin

Figure 14: Buzzer form Wokwi

MISCELLANEOUS

LED:

- **Usage**: Provides visual feedback for status of Wi-Fi connectivity
- **Connections**:
  - Anode (A) to pin 13 on ESP32
  - Cathode (C) to GND on ESP32

wokwi-led Reference

Standard 5mm LED.

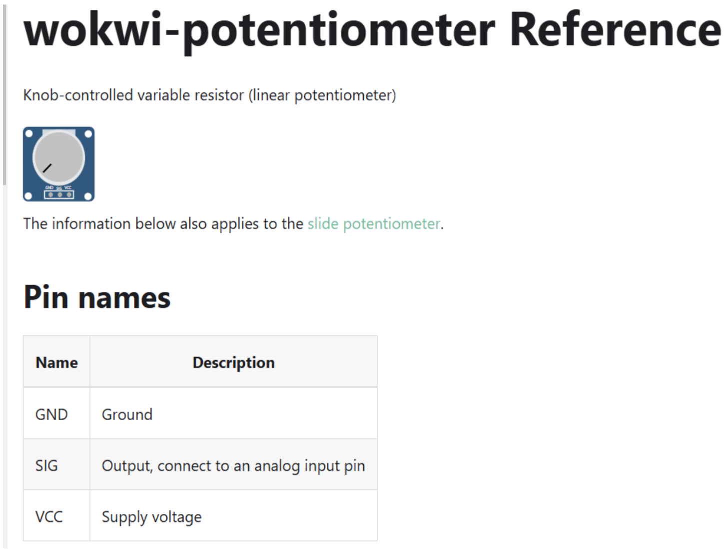
Pin names

Name	Description
A	Anode (positive pin)
C	Cathode (negative pin)

Figure 15: Led from Wokwi

Potentiometer:

- **Usage**: Allows manual adjustment and input, such as calibrating sensor thresholds or controlling other parameters.
- **Connections**:
  - VCC to 3.3V on ESP32
  - GND to GND on ESP32
  - Signal pin (SIG) to pin 35 on ESP32



wokwi-potentiometer Reference

Knob-controlled variable resistor (linear potentiometer)

The information below also applies to the slide potentiometer.

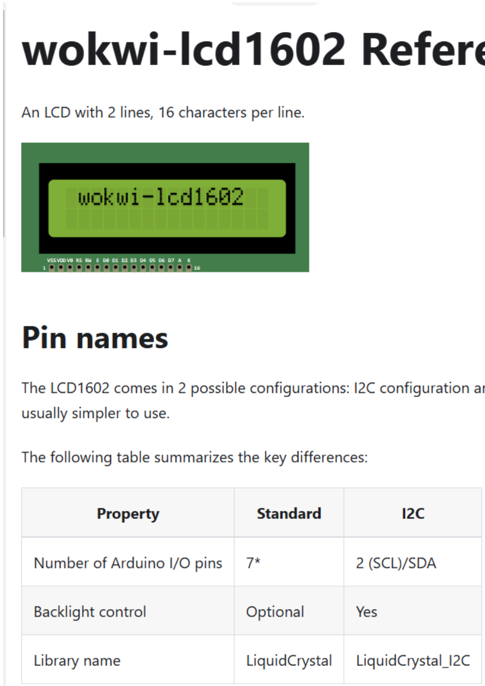
Pin names

Name	Description
GND	Ground
SIG	Output, connect to an analog input pin
VCC	Supply voltage

Figure 16: Potentiometer from Wokwi

LCD **(Liquid Crystal Display)**:

- **Usage**: Displays real-time sensor data and system status messages.
- **Connections**:
  - VCC to 3.3V on ESP32
  - GND to GND on ESP32
  - SCL to pin 22 on ESP32
  - SDA to pin 21 on ESP32



wokwi-lcd1602 Reference

An LCD with 2 lines, 16 characters per line.

Pin names

The LCD1602 comes in 2 possible configurations: I2C configuration and standard configuration. The I2C configuration is usually simpler to use.

The following table summarizes the key differences:

Property	Standard	I2C
Number of Arduino I/O pins	7*	2 (SCL)/SDA
Backlight control	Optional	Yes
Library name	LiquidCrystal	LiquidCrystal_I2C

Figure 17: LCD from Wokwi

10. SYSTEM LIMITATIONS

Despite its functionality and benefits, the smart weather monitoring system faces several challenges:

- **Power Management:**
  Continuous operation of the ESP32 in active mode with Wi-Fi connectivity can lead to high power consumption, limiting its deployment in remote areas without access to mains power.
- **Data Accuracy**:
  The DHT-22 sensor has limited accuracy and resolution. For critical applications, more precise sensors like the DHT22 might be needed.
- **Wi-Fi Dependency**:
  The system relies on stable Wi-Fi connectivity for real-time data transmission to the cloud. In areas with poor internet infrastructure, this can affect its performance.
- **Limited Parameter Monitoring**:
  The system monitors only temperature, humidity, and light intensity. It does not track other critical weather parameters, such as air pressure, wind speed, or air quality, limiting its applicability for comprehensive weather analysis.
- **Scalability**:
  Expanding the system to include more sensors or multiple deployments in different locations requires additional hardware and increased energy demands.
- **LDR**: We currently do not have many sensor options in Wiki. At the moment, we are using an LDR to determine if the day is sunny, cloudy, etc. However, this method is limited to daytime use, as the LDR is not effective for checking weather conditions, such as cloud cover, during the night.

11. ESTIMATE OF POWER CONSUMPTION

- ESP32: Up to 240mA (Wi-Fi active transmission)
- DHT22: 2.5mA
- LED: 20mA
- LCD: Up to 60mA (assuming backlight is on)
- Buzzer: 30mA (when active)

Total Current Consumption:

240mA(ESP32)+2.5mA(DHT22)+20mA(LED)+60mA(LCD)+30mA(Buzzer)=352.5mA240mA (ESP32) + 2.5mA (DHT22) + 20mA (LED) + 60mA (LCD) + 30mA (Buzzer) = 352.5mA240mA(ESP32)+2.5mA(DHT22)+20mA(LED)+60mA(LCD)+30mA(Buzzer)=352.5mA

Estimation of Total Power Consumption

The total power consumption can be calculated using the formula:

Power (W)=Voltage (V)×Current (A

Given the voltage is 3.3V:

Power=3.3V×0.3525A≈1.16W\

Power Supply Recommendations

Based on the estimated total power consumption, you should use a power supply that can provide at least 1.16W of power. To ensure reliable operation and some margin for safety, it's recommended to use a power supply with a higher capacity:

- **Voltage:** 3.3V (this is critical as the ESP32 and other components operate at this voltage)
- **Current:** At least 500mA to 1A

12. REFERENCES

1. Lux Levels for Weather https://www.electricalcounter.co.uk/articles/what-are-lux-levels
2. Wokwi. *Virtual Parts Documentation*. Available at: https://docs.wokwi.com/parts.
3. *ThingSpeak Documentation*. Available at: https://www.mathworks.com/help/thingspeak/

THE END