ENVIRONMENTAL MONITORING IN PARKS

The scope of this document is to identify the problem and find solution for park visitors and park management to receive real time Environmental Data.

Objectives:

The project aims to enhance the visitor experience in parks by implementing an IoT-based environmental monitoring system. The objectives include real-time monitoring of temperature, humidity, and light intensity. This data is used to provide insights for park management and enable smart functionalities such as automatic street lights. The project encourages outdoor activities and ensures a comfortable environment for visitors.

IoT Device Setup:

1. **ESP32 Microcontroller:**
    - **Sensors:** DHT22 (Temperature and Humidity), LDR (Light Intensity)
    - **Actuators:** PIR Motion Sensor for Smart Street Lights
    - **Communication:** Wi-Fi for Data Transmission
2. **Arduino Uno:**
    - **Sensor:** Ultrasonic Sensor (For demonstration purposes)
    - **Communication:** Communication with ESP32

Platform Development:

Wokwi Simulator:

- **Purpose:** Used for virtual prototyping and simulation of the IoT devices and Arduino code.
- **Features:** Real-time sensor data simulation, interactive visualization, and debugging capabilities.

Web-based Interface:

- **Components:** HTML, CSS, JavaScript
- **Functionality:** Provides a user-friendly interface to display real-time temperature and humidity data. Allows users to request data from the IoT devices.
- **Communication:** Integrates with ThingSpeak API to fetch and display IoT device data.

Code Implementation:

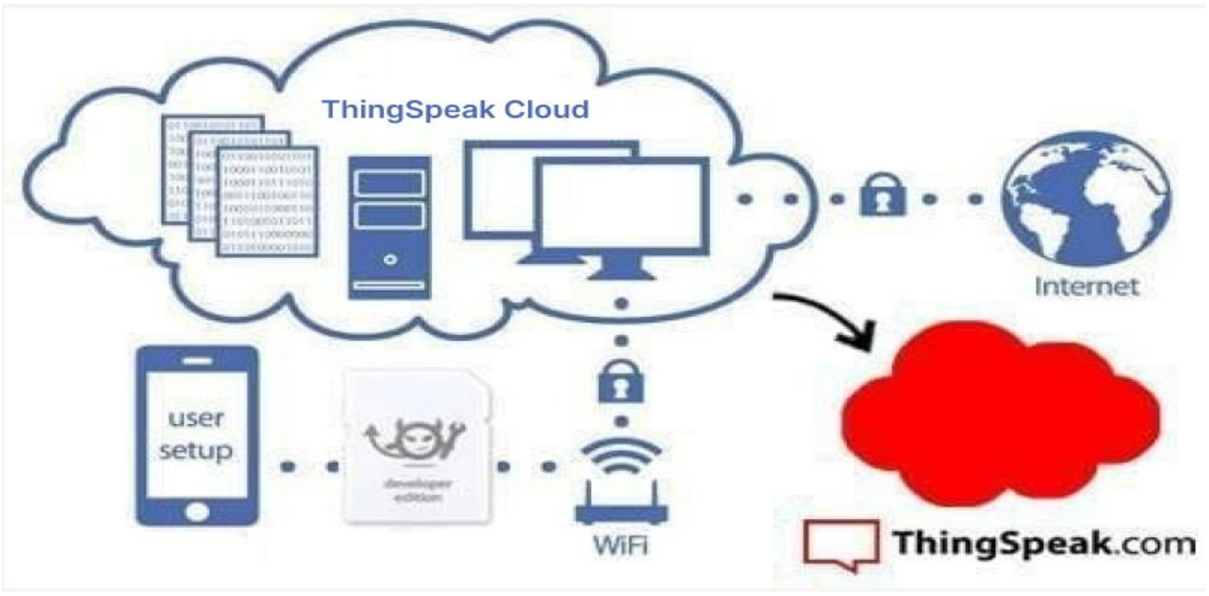
1. **Arduino Code (Sketch.ino):**
    - Initializes sensors and actuators.

- Reads sensor data and processes it.
- Controls smart street lights based on environmental conditions.
- Establishes Wi-Fi connection and sends data to ThingSpeak.

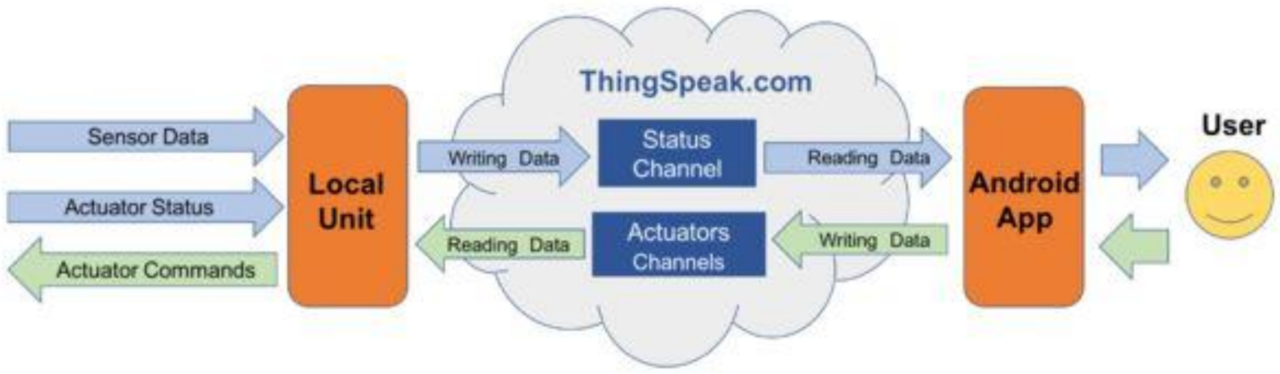
2. **Web Interface Code (Index.html, Styles.css, Script.js):**
    - Defines the structure and style of the web interface.
    - Handles user interactions, fetches real-time data from ThingSpeak using API.
    - Displays fetched data dynamically and updates the UI.

Diagrams and Schematics:

1. **Circuit Diagram:**

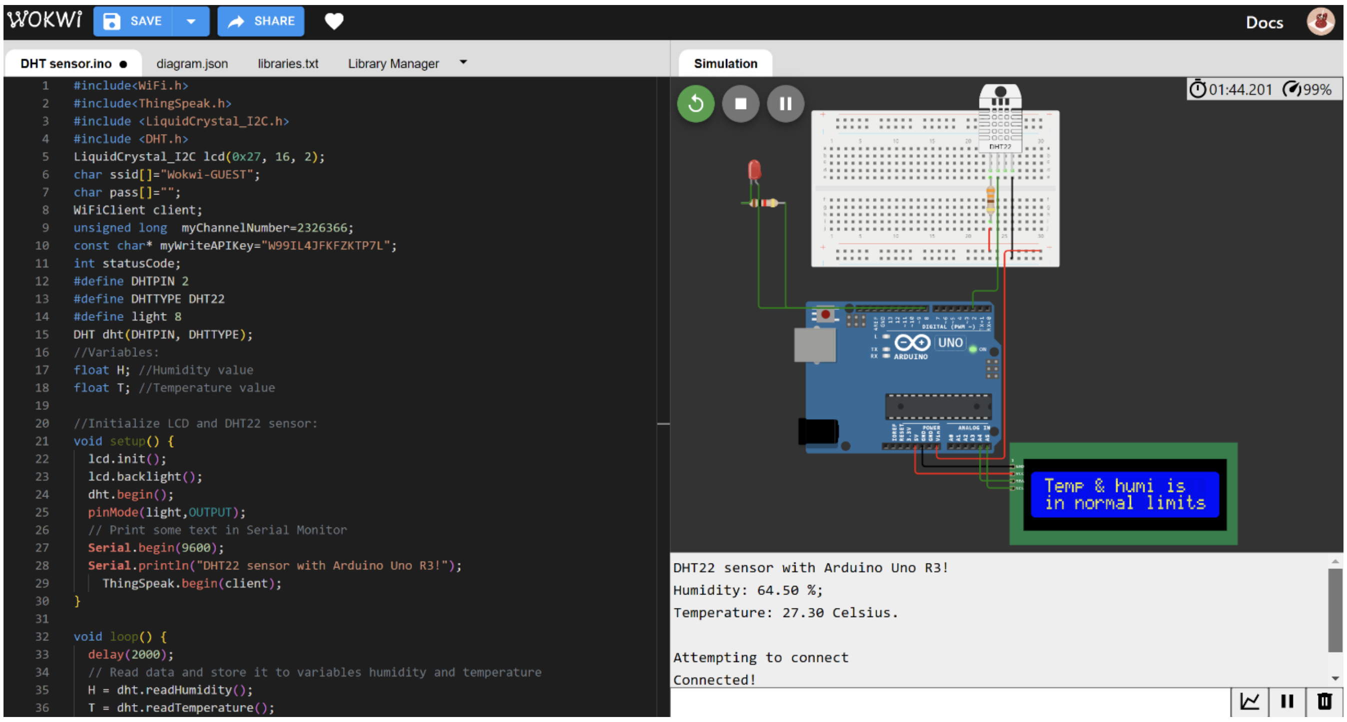


2. **System Architecture:**



Screenshots:

1. **IoT Device Simulation:**



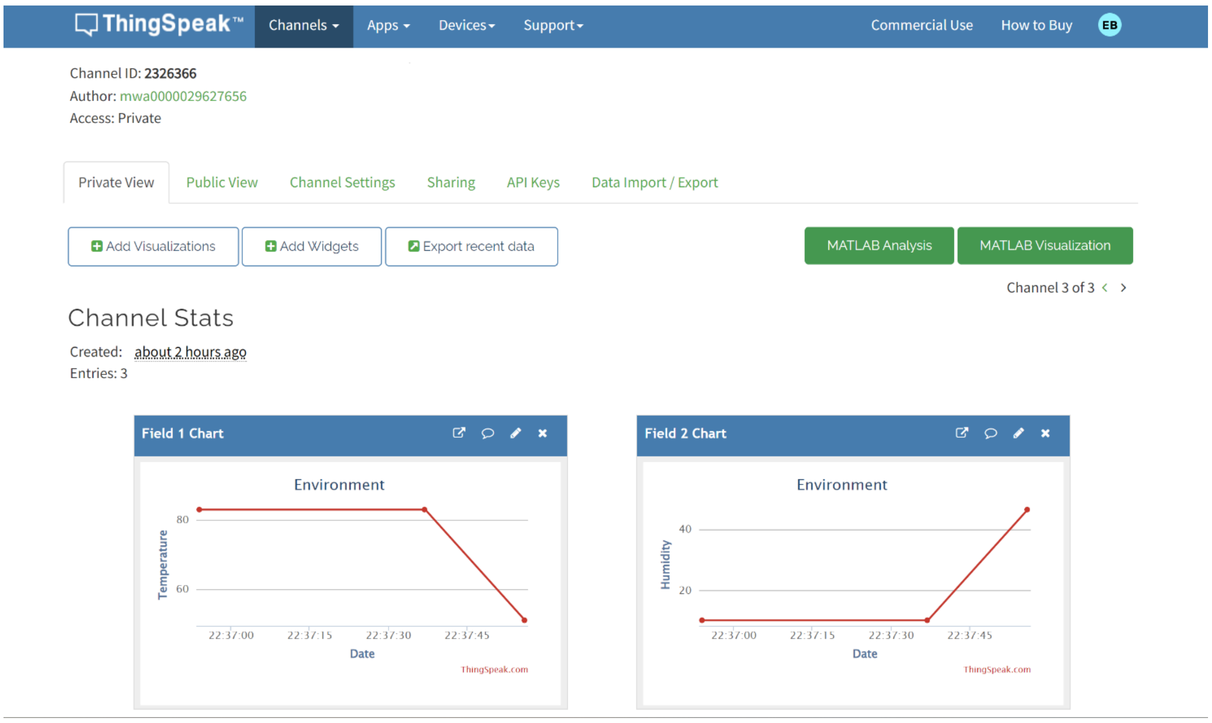
```
#include<WiFi.h>
#include<ThingSpeak.h>
#include <LiquidCrystal_I2C.h>
#include <DHT.h>
LiquidCrystal_I2C lcd(0x27, 16, 2);
char ssid[]="Wokwi-GUEST";
char pass[]="";
WiFiClient client;
unsigned long  myChannelNumber=2326366;
const char* myWriteAPIKey="W99IL4JFKFZKTP7L";
int statusCode;
#define DHTPIN 2
#define DHTTYPE DHT22
#define light 8
DHT dht(DHTPIN, DHTTYPE);
//Variables:
float H; //Humidity value
float T; //Temperature value

//Initialize LCD and DHT22 sensor:
void setup() {
  lcd.init();
  lcd.backlight();
  dht.begin();
  pinMode(light,OUTPUT);
  // Print some text in Serial Monitor
  Serial.begin(9600);
  Serial.println("DHT22 sensor with Arduino Uno R3!");
    ThingSpeak.begin(client);
}

void loop() {
  delay(2000);
  // Read data and store it to variables humidity and temperature
  H = dht.readHumidity();
  T = dht.readTemperature();
```

```
DHT22 sensor with Arduino Uno R3!
Humidity: 64.50 %;
Temperature: 27.30 Celsius.

Attempting to connect
Connected!
```

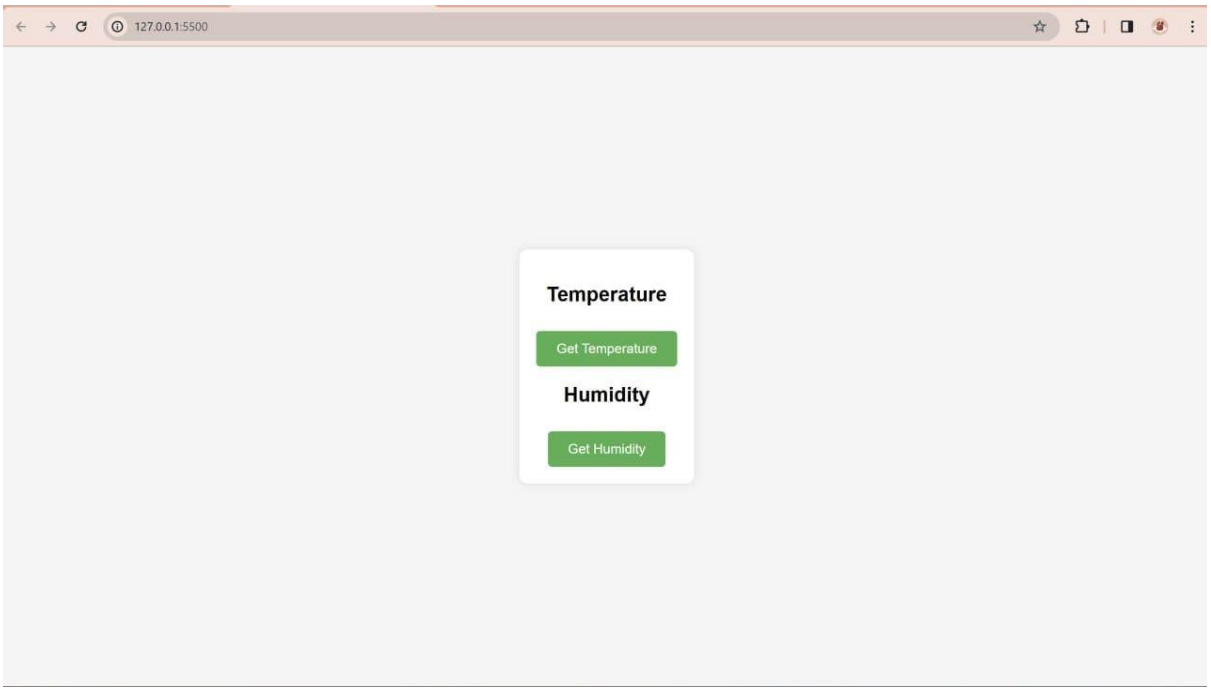


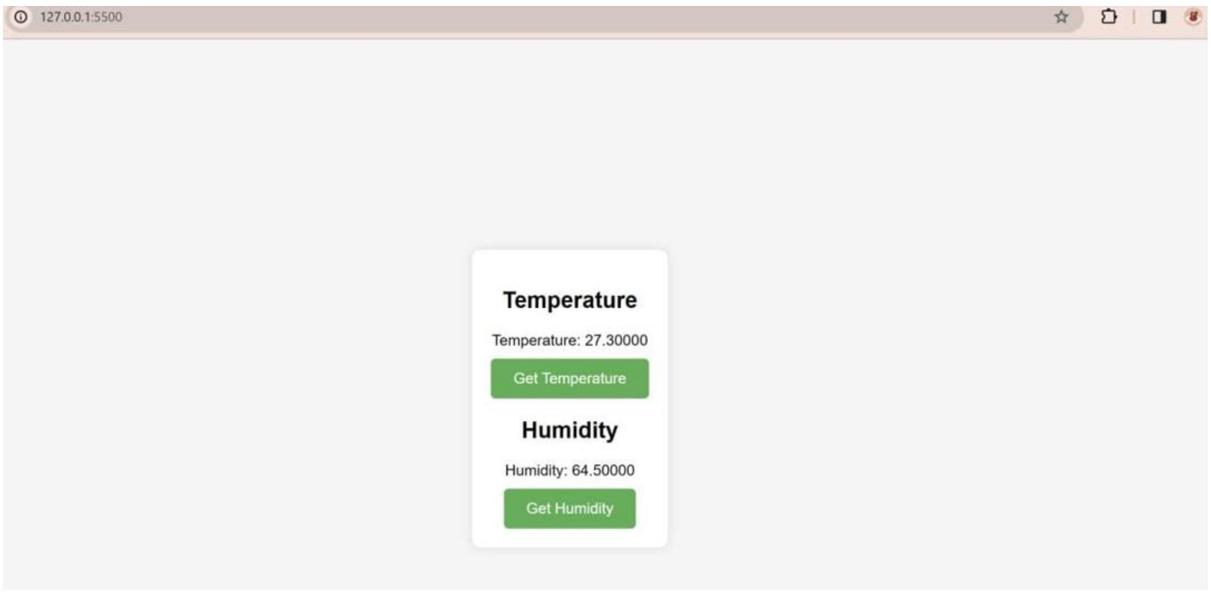
Channel ID: 2326366
Author: mwa0000029627656
Access: Private

Channel Stats

Created: about 2 hours ago
Entries: 3

2. **Web Interface:**





Temperature: 27.30000

Humidity: 64.50000

Project Explanation:

The IoT devices, including ESP32 and Arduino Uno, are equipped with sensors to measure temperature, humidity, light intensity, and proximity. These devices are simulated using Wokwi Simulator, ensuring real-time data generation. The data is then sent to ThingSpeak cloud platform through Wi-Fi.

The web interface allows users to view current environmental data and trigger data requests. When a user clicks the "Get Temperature" or "Get Humidity" buttons, the web interface communicates with ThingSpeak, fetches the corresponding data, and displays it dynamically. The interface provides a seamless experience for users to monitor park conditions remotely.

Submission:

- **GitHub Repository:**

  https://github.com/eee-lain/IOT_phase1.git

- **Replication Instructions:**

1. Set up ESP32 and Arduino Uno boards with specified sensors and actuators.

2. Upload the provided Arduino code (Sketch.ino) to the respective boards using Arduino IDE.

3. Deploy the web interface files (Index.html, Styles.css, Script.js) on a web server.

4. Update ThingSpeak channel ID and API key in the Arduino code.

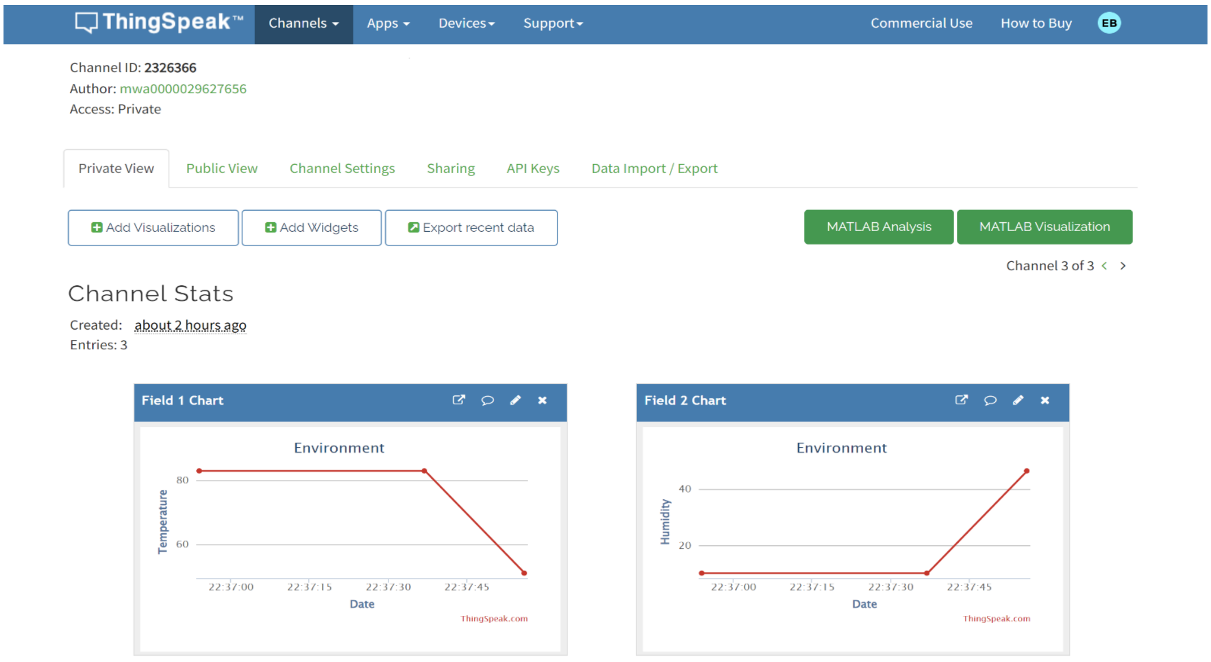
5. Access the web interface through a web browser to view real-time data and trigger requests.

- **Integration with Python:**

  - Python scripts can be used to analyze the fetched data from ThingSpeak, generate reports, and trigger automated actions based on environmental conditions. Example Python scripts for data analysis and integration are provided in the GitHub repository.

- **Example Outputs:**

2. **IoT Device Data Transmission:**



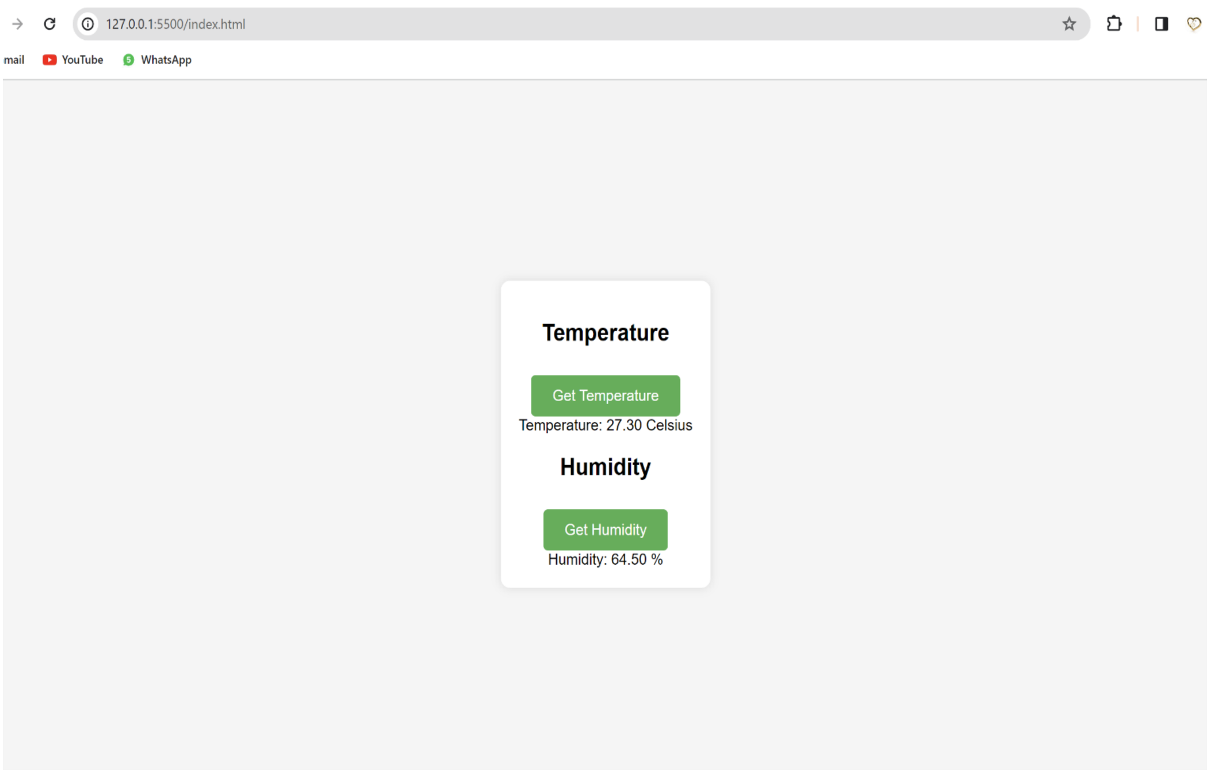
Channel ID: 2326366
Author: mwa0000029627656
Access: Private

Channel Stats

Created: about 2 hours ago
Entries: 3

---

3. **Platform UI:**



This documentation provides a comprehensive overview of the IoT environmental monitoring project, including its objectives, device setup, platform development, code implementation, diagrams, and instructions for replication. For detailed code and further exploration, please refer to the provided GitHub repository.