Shahjalal University of Science and Technology, Sylhet

Project Report

DEPARTMENT OF ELETRICAL AND ELECTRONIC ENGINEERING

Course Title: Basic communication Engineering Lab

Course code: EEE 340

Group no:04

Project Name : **Development of an Intelligent Weather Reporting System**

SUBMITTED BY

SHARMIN AKTER SWEETY
REG. NO:2020338016
MD. SALEH MUSA
REG. NO:2020338029
SHAHIDA AKTER ESHA
REG. NO:2020338030
ISHRA CHOWDHURY MOUMI
REG. NO: 2020338031
ANIK DEB NATH
REG. NO: 2020338032
TASMIA ISLAM
REG. NO:2020338033
RAYSHA FARJANA
REG. NO:2020338054

SUBMITTED TO

MOHONA DAS GUPTA ,
LECTURER ,
Department of Electrical and Electronic Engineering.
Shahjalal University of Science and Technology, Sylhet.

Development of an Intelligent Weather Reporting System

Abstract

This research proposes an ESP32-based weather reporting system that retrieves real-time weather data from the OpenWeatherMap API and transmits information via Bluetooth. The system enables users to dynamically change the location for weather updates using Bluetooth communication. The project aims to provide a cost-effective, real-time weather monitoring solution for remote or mobile users.

Introduction

Weather monitoring is essential for various applications, including agriculture, transportation, and disaster management. Traditional weather stations are expensive and often inaccessible in remote areas. The proposed system leverages the ESP32 microcontroller to provide an affordable and efficient weather monitoring solution. The device fetches real-time weather data over Wi-Fi and transmits it via Bluetooth, allowing users to receive updates on a Bluetooth-enabled device.

Objectives

- Develop a low-cost, real-time weather monitoring system using ESP32.
- Retrieve weather data from the OpenWeatherMap API.
- Enable dynamic city selection for weather updates via Bluetooth.
- Ensure seamless connectivity and efficient data transmission over Wi-Fi and Bluetooth.

Methodology

Hardware Components

- ESP32 Board
- Bluetooth-enabled device (e.g., smartphone or PC)

Software Requirements

- Arduino IDE
- ESP32 Board Package
- Required Libraries:
    - WiFi.h (for Wi-Fi connectivity)
    - HTTPClient.h (for HTTP requests)
    - BluetoothSerial.h (for Bluetooth communication)

System Architecture

1. **Wi-Fi Connection & Data Retrieval:**
    - The ESP32 connects to a Wi-Fi network and retrieves weather data from the OpenWeatherMap API.
    - API responses are processed in JSON format.
2. **Bluetooth Communication:**
    - The system initializes Bluetooth communication and allows users to send city names to update the weather location.
    - Received weather data is transmitted via Bluetooth to the connected device.
3. **Real-Time Data Processing:**
    - The ESP32 continuously checks for new city inputs and fetches weather updates accordingly.
    - The system updates weather data every 60 seconds.

Code Explanation

Global Variables

```
const char* ssid = "404";              // Replace with your Wi-Fi SSID
const char* password = "123456788";      // Replace with your Wi-Fi Password.
const char* apiKey = "87dee9288951b768602d21ed3f5414a0"; // Replace with your API Key
const char* city = "SYLHET"; // Change to your preferred city
```

These hold the OpenWeatherMap API key and the city for which the weather data is requested.

```
String server = "http://api.openweathermap.org/data/2.5/weather?q=" + String(city) + "&appid="
+ String(apiKey) + "&units=metric";
```

`setup()` Function

```
void setup() {
  Serial.begin(115200);      // Debug Monitor
```

```
  SerialBT.begin("ESP32_Weather"); // Bluetooth Device Nam
  WiFi.begin(ssid, password);
  Serial.print("Connecting to WiFi");
  while (WiFi.status() != WL_CONNECTED) {
    delay(1000);
    Serial.print(".");
  }
  Serial.println("\nWiFi Connected!");
  SerialBT.println("Bluetooth Ready!");
}
```

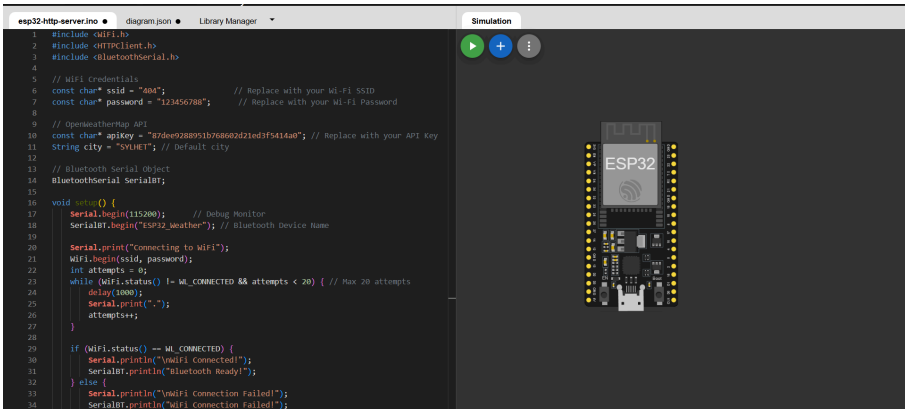
- Attempts to connect to Wi-Fi.
- Displays connection status via the serial monitor.
- Send a Bluetooth message confirming readiness.

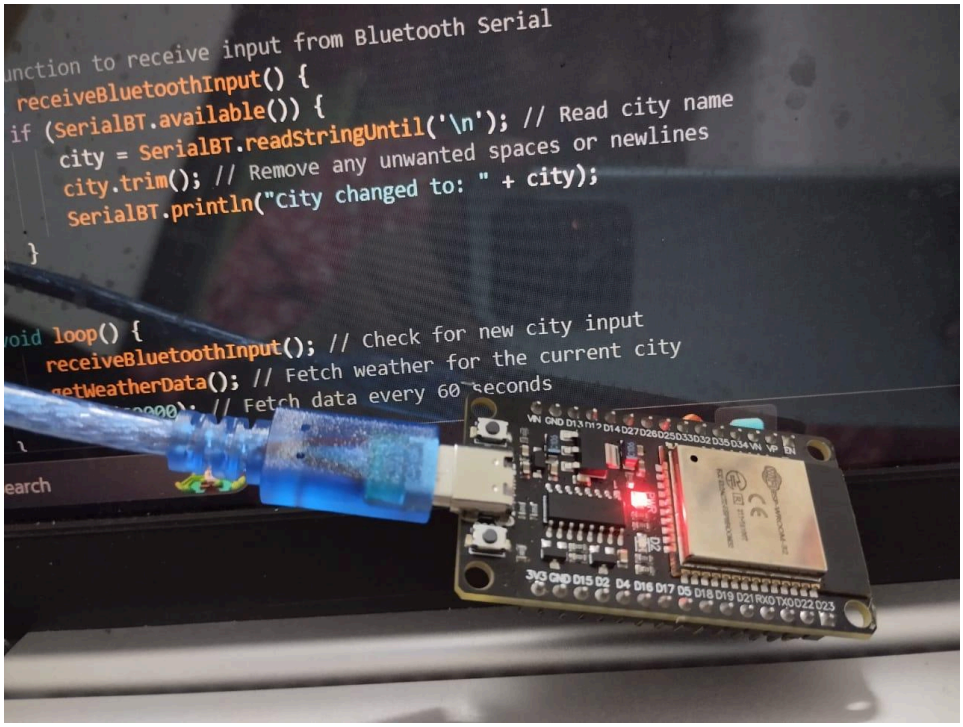
`getWeatherData()` Function

```
 void getWeatherData() {
  if (WiFi.status() == WL_CONNECTED) {
    HTTPClient http;
    http.begin(server);
    int httpResponseCode = http.GET();
      if (httpResponseCode > 0) {
      String payload = http.getString();
      SerialBT.println("Weather Data: ");
      SerialBT.println(payload);
    } else {
      SerialBT.println("API Request Failed");
    }
    http.end();
  } else {
    SerialBT.println("WiFi Lost!");
  }
}
```

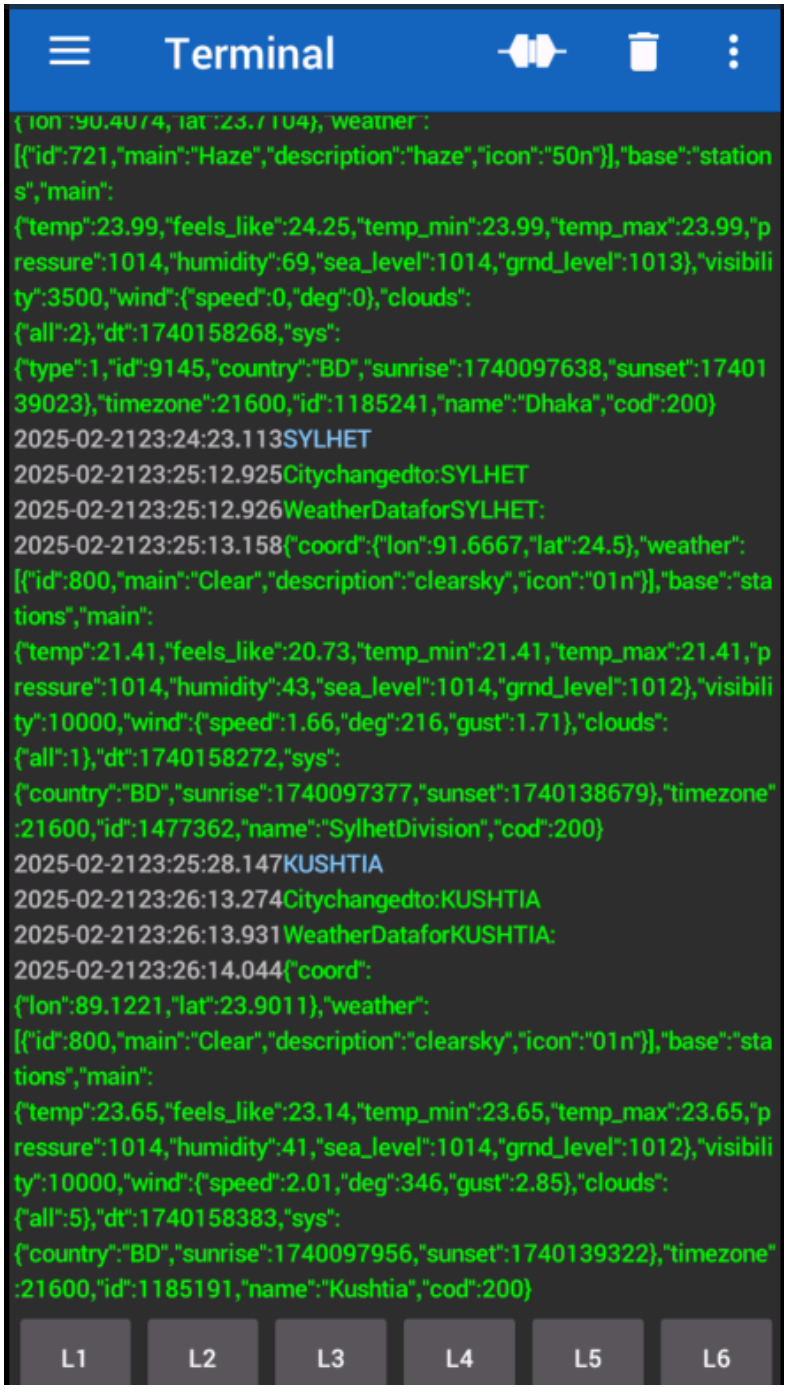
- If the API request is successful, send the data via Bluetooth.
- If the request fails, send an error message.
- If Wi-Fi is disconnected, it alerts the user via Bluetooth.

Hardware setup





RESULT


Terminal
{"lon":90.4074,"lat":23.7104},"weather":
[{"id":721,"main":"Haze","description":"haze","icon":"50n"}],"base":"station
s","main":
{"temp":23.99,"feels_like":24.25,"temp_min":23.99,"temp_max":23.99,"p
ressure":1014,"humidity":69,"sea_level":1014,"grnd_level":1013},"visibili
ty":3500,"wind":{"speed":0,"deg":0},"clouds":
{"all":2},"dt":1740158268,"sys":
{"type":1,"id":9145,"country":"BD","sunrise":1740097638,"sunset":17401
39023},"timezone":21600,"id":1185241,"name":"Dhaka","cod":200}
2025-02-2123:24:23.113SYLHET
2025-02-2123:25:12.925Citychangedto:SYLHET
2025-02-2123:25:12.926WeatherDataforSYLHET:
2025-02-2123:25:13.158{"coord":{"lon":91.6667,"lat":24.5},"weather":
[{"id":800,"main":"Clear","description":"clearsky","icon":"01n"}],"base":"sta
tions","main":
{"temp":21.41,"feels_like":20.73,"temp_min":21.41,"temp_max":21.41,"p
ressure":1014,"humidity":43,"sea_level":1014,"grnd_level":1012},"visibili
ty":10000,"wind":{"speed":1.66,"deg":216,"gust":1.71},"clouds":
{"all":1},"dt":1740158272,"sys":
{"country":"BD","sunrise":1740097377,"sunset":1740138679},"timezone"
:21600,"id":1477362,"name":"SylhetDivision","cod":200}
2025-02-2123:25:28.147KUSHTIA
2025-02-2123:26:13.274Citychangedto:KUSHTIA
2025-02-2123:26:13.931WeatherDataforKUSHTIA:
2025-02-2123:26:14.044{"coord":
{"lon":89.1221,"lat":23.9011},"weather":
[{"id":800,"main":"Clear","description":"clearsky","icon":"01n"}],"base":"sta
tions","main":
{"temp":23.65,"feels_like":23.14,"temp_min":23.65,"temp_max":23.65,"p
ressure":1014,"humidity":41,"sea_level":1014,"grnd_level":1012},"visibili
ty":10000,"wind":{"speed":2.01,"deg":346,"gust":2.85},"clouds":
{"all":5},"dt":1740158383,"sys":
{"country":"BD","sunrise":1740097956,"sunset":1740139322},"timezone"
:21600,"id":1185191,"name":"Kushtia","cod":200}
L1
L2
L3
L4
L5
L6

Expected Outcomes

- [ ] A functional prototype of an ESP32-based weather monitoring system.
- [ ] Real-time weather updates available on Bluetooth-connected devices.
- [ ] Seamless Wi-Fi and Bluetooth integration for dynamic city selection.
- [ ] Improved accessibility to weather data in remote locations.

Limitations

- [ ] Dependence on a stable Wi-Fi connection for data retrieval.
- [ ] Potential API request limits imposed by OpenWeatherMap.
- [ ] Bluetooth range constraints limiting data accessibility.

Future Work

- [ ] Implement a graphical display for Earthquake monitoring.
- [ ] Improve power efficiency for battery-operated applications.
- [ ] Expand functionality to include any natural Disaster.

Conclusion

This research aims to develop an ESP32-based weather reporting system that leverages Wi-Fi and Bluetooth for real-time weather monitoring. By enabling users to dynamically update the monitored location via Bluetooth, the system enhances accessibility and usability, making weather data more readily available in various applications.

References

[1] OpenWeatherMap API Documentation: https://openweathermap.org/api

[2] Espressif ESP32 Documentation: https://docs.espressif.com