Task 5: IoT Mini Project – Smart Pet Bowl Monitor

This report details the design, implementation, and functionality of the Smart Pet Bowl Monitor, an Internet of Things (IoT) solution created to help pet owners remotely monitor the food or water levels in their pet's bowl. The system leverages an ESP8266 (NodeMCU) microcontroller, an HC-SR04 ultrasonic sensor for distance measurement, and a buzzer for local audible alerts. Data is transmitted over Wi-Fi to a Google Firebase Realtime Database, which serves as the cloud backend. A custom-built webpage subscribes to this database, providing a user-friendly interface to display the bowl's status ("Full" or "Empty") in real-time. The entire system demonstrates a complete end-to-end IoT workflow, from physical sensing to cloud data synchronization and remote visualization.

1. Introduction

Ensuring that pets have a consistent supply of food and water is a primary concern for any pet owner, especially for those with busy schedules or who are away from home. Traditional methods rely on manual checks, which are not always feasible. The Smart Pet Bowl Monitor addresses this challenge by providing an automated, real-time monitoring system. By placing an ultrasonic sensor above the pet bowl, the system can accurately determine the level of its contents. When the level drops below a predefined threshold, the system triggers both a local audible alarm and updates a cloud database. This allows the owner to check the status from anywhere in the world via a simple web interface, ensuring their pet is always cared for.

Project Objectives:

- To design and build a non-contact system for measuring the content level in a pet bowl.
- To use an ESP8266 microcontroller for sensor data processing and Wi-Fi communication.
- To implement a cloud-based backend using Firebase Realtime Database for data storage and access.
- To create a web-based dashboard for real-time monitoring of the bowl's status.
- To incorporate a local buzzer for low-level alerts.

2. System Architecture and Components

The system is composed of hardware components for sensing and control, and a software stack for processing, communication, and data presentation.

2.1. System Block Diagram

- **Sensing Layer:** HC-SR04 Ultrasonic Sensor measures the distance.
- **Processing Layer:** ESP8266 NodeMCU reads and processes sensor data, connects to Wi-Fi, and controls the buzzer.
- **Communication Layer:** Wi-Fi is used to transmit data from the ESP8266 to the cloud.
- **Cloud Layer:** Firebase Realtime Database stores and synchronizes the data.
- **Application Layer:** A custom webpage fetches and displays the data for the end-user.

2.2. Hardware Components

- **ESP8266 (NodeMCU):** The core of the project. It's a low-cost microcontroller with built-in Wi-Fi capabilities, making it ideal for IoT applications. It reads data from the sensor, executes the control logic, and communicates with the Firebase database.
- **HC-SR04 Ultrasonic Sensor:** This sensor is used to measure the distance to the food or water in the bowl. It works by emitting an ultrasonic pulse and measuring the time it takes for the echo to return.
- **Buzzer:** A simple piezoelectric buzzer provides an immediate, local audible alert when the bowl's content level is critically low.

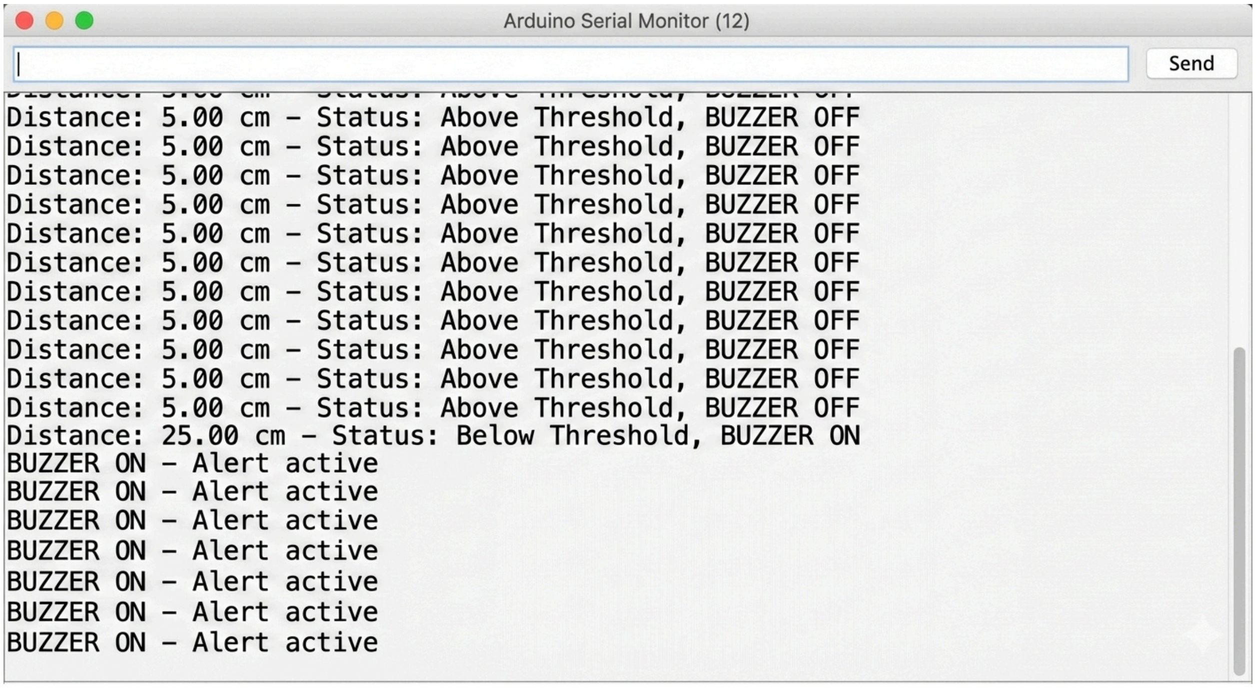
2.3. Software and Cloud Services

- **Arduino IDE:** The Integrated Development Environment used to write, compile, and upload the firmware to the ESP8266 microcontroller. The programming language is C/C++.

```
// =====================================================================
//      SMART PET BOWL MONITOR - ESP8266 FIRMWARE
// =====================================================================
// This code measures the distance using an ultrasonic sensor. If the
// sensor fails, it simulates the distance by toggling between 5cm and 25cm.
// It activates a local buzzer when the level is low and sends the live
// status to a Firebase Realtime Database.
// =====================================================================

// --- LIBRARY INCLUDES ---
#include <ESP8266WiFi.h>
#include <FirebaseESP8266.h> // The Firebase library for ESP8266

// ---------------- CONFIGURATION ----------------

// --- WiFi Credentials ---
#define WIFI_SSID "XYZ"
#define WIFI_PASSWORD "98765432101"

// --- Firebase Credentials ---
// IMPORTANT: The host URL must NOT include "https://"" or a trailing "/"
#define FIREBASE_HOST "pet-food---water-monitor-default-rtdb.asia-southeast1.firebasedatabase.app"
```

Message (Enter to send message to 'NodeMCU 1.0 (ESP-12E Module)' on 'COM3')

```
⚠️ Ultrasonic not responding, using simulated values...
Distance: 25.00 cm
🚨 Buzzer ON (Above Threshold)
📡 Data sent to Firebase!
⚡ Alert updated!
⚠️ Ultrasonic not responding, using simulated values...
Distance: 5.00 cm
✅ Buzzer OFF (Below Threshold)
📡 Data sent to Firebase!
⚡ Alert updated!
⚠️ Ultrasonic not responding, using simulated values...
Distance: 25.00 cm
🚨 Buzzer ON (Above Threshold)
📡 Data sent to Firebase!
⚡ Alert updated!
```



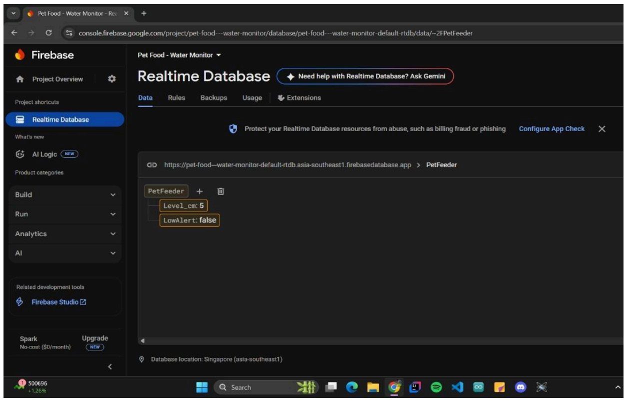
```
Distance: 5.00 cm - Status: Above Threshold, BUZZER OFF
Distance: 5.00 cm - Status: Above Threshold, BUZZER OFF
Distance: 5.00 cm - Status: Above Threshold, BUZZER OFF
Distance: 5.00 cm - Status: Above Threshold, BUZZER OFF
Distance: 5.00 cm - Status: Above Threshold, BUZZER OFF
Distance: 5.00 cm - Status: Above Threshold, BUZZER OFF
Distance: 5.00 cm - Status: Above Threshold, BUZZER OFF
Distance: 5.00 cm - Status: Above Threshold, BUZZER OFF
Distance: 5.00 cm - Status: Above Threshold, BUZZER OFF
Distance: 5.00 cm - Status: Above Threshold, BUZZER OFF
Distance: 5.00 cm - Status: Above Threshold, BUZZER OFF
Distance: 25.00 cm - Status: Below Threshold, BUZZER ON
BUZZER ON - Alert active
BUZZER ON - Alert active
BUZZER ON - Alert active
BUZZER ON - Alert active
BUZZER ON - Alert active
BUZZER ON - Alert active
BUZZER ON - Alert active
```

- **Firebase Realtime Database:** A cloud-hosted NoSQL database that allows for real-time data storage and synchronization. It acts as the central hub where the ESP8266 sends sensor data and from which the webpage reads it.
- **HTML, CSS, and JavaScript:** Standard web technologies used to create the front-end dashboard. The JavaScript code utilizes the Firebase SDK to establish a live connection to the database.

3. Working Principle and Implementation

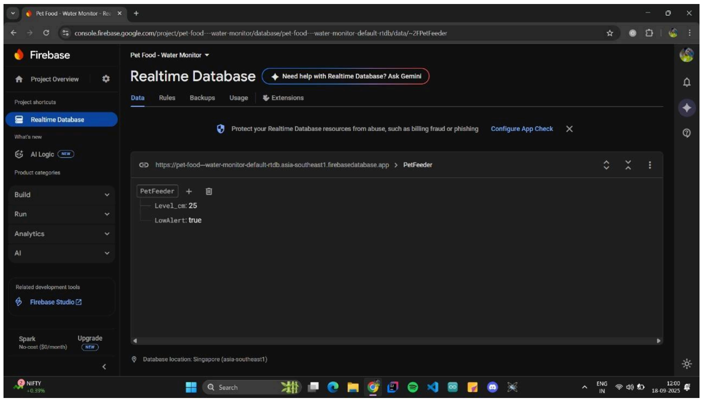
The operational flow of the project is sequential and continuous, creating a real-time feedback loop.

1. **Sensing:** The HC-SR04 ultrasonic sensor, positioned above the pet bowl, sends out a sound pulse. The distance is calculated based on the time-of-flight of the returning echo. A smaller distance reading indicates a fuller bowl, while a larger distance reading signifies an emptier bowl.
2. **Data Processing (on ESP8266):**
    - The ESP8266 firmware reads the raw distance. To ensure stable and reliable readings, a **moving average filter** is implemented. It takes NUM_SAMPLES (5 in this case) and averages them to produce a smoothed distance value.
    - This smoothed distance is compared against a threshold (THRESHOLD_CM). Hysteresis (THRESHOLD_CM_HIGH, THRESHOLD_CM_LOW) is used to prevent the alert state from rapidly toggling.
    - If the distance exceeds the threshold, the isWaterLow flag is set to true, activating the buzzer.
3. **Cloud Communication:**
    - The ESP8266 connects to the Wi-Fi and then authenticates with the Firebase project using the host URL and auth token.
    - The ESP8266 continuously pushes two key pieces of information to the /PetFeeder/ path in the Firebase database:
        - Level_cm: The current smoothed distance reading.
        - LowAlert: A boolean value (true/false) indicating the alert status.



https://pet-food—water-monitor-default-rtdb.asia-southeast1.firebasedatabase.app > PetFeeder

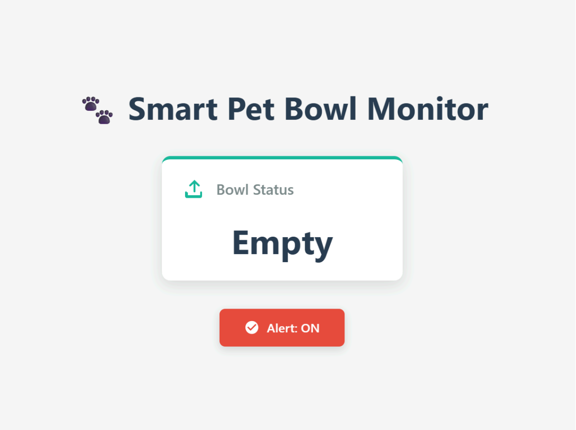
PetFeeder
Level_cm: 5
LowAlert: false



https://pet-food—water-monitor-default-rtdb.asia-southeast1.firebasedatabase.app > PetFeeder

PetFeeder
Level_cm: 25
LowAlert: true

4. **Web-based Visualization:**
   - The user accesses a static HTML page.
   - JavaScript on the page uses the Firebase SDK to establish a persistent, real-time connection to the database.
   - onValue() listeners are attached to the Level_cm and LowAlert data points. When data changes in Firebase, these listeners trigger instantly, updating the webpage content to show "Full" or "Empty" and "Status: OK" or "Alert: ON".

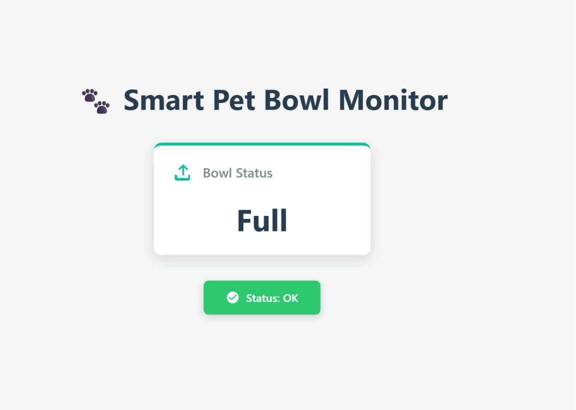


Smart Pet Bowl Monitor

Bowl Status

Empty

Alert: ON

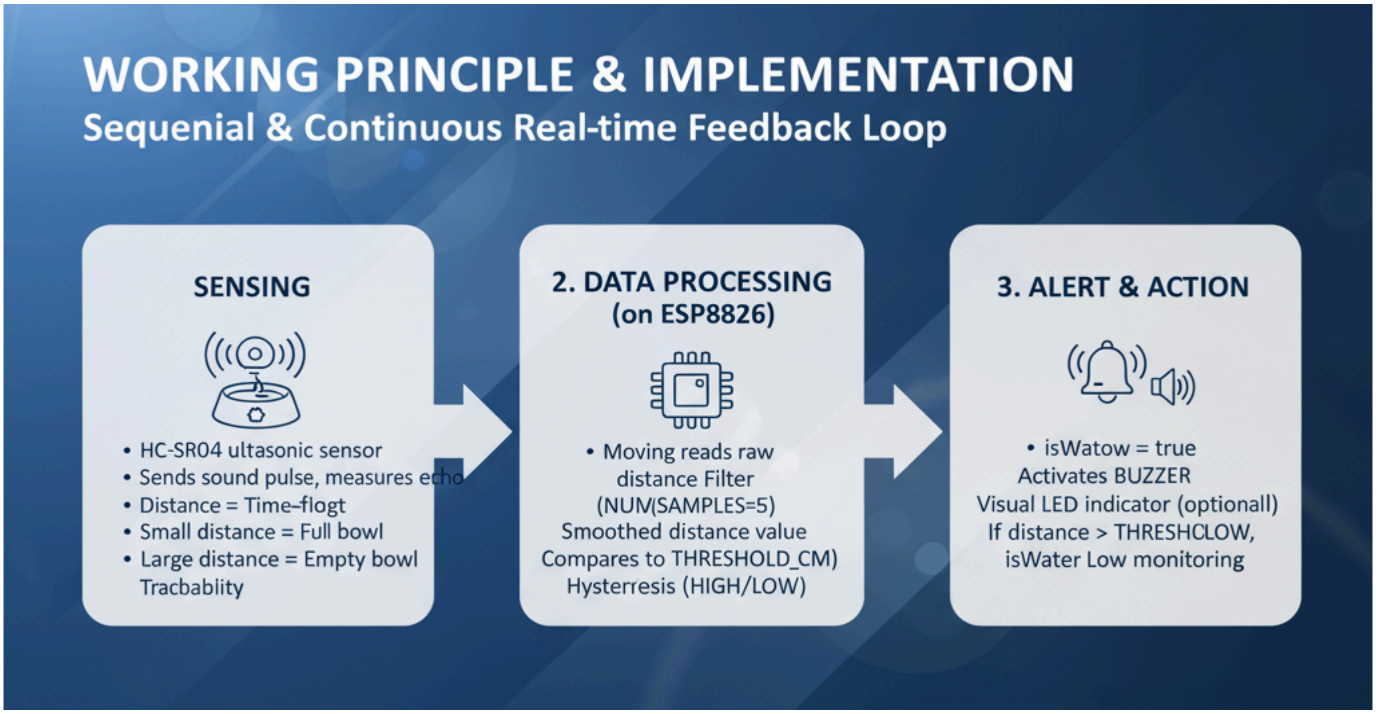


Smart Pet Bowl Monitor

Bowl Status

Full

Status: OK



WORKING PRINCIPLE & IMPLEMENTATION

Sequenial & Continuous Real-time Feedback Loop

SENSING

- HC-SR04 ultasonic sensor
- Sends sound pulse, measures echo
- Distance = Time–flogt
- Small distance = Full bowl
- Large distance = Empty bowl
Tracbablity

2. DATA PROCESSING (on ESP8826)

- Moving reads raw distance Filter (NUM(SAMPLES=5)
Smoothed distance value
Compares to THRESHOLD_CM)
Hysterresis (HIGH/LOW)

3. ALERT & ACTION

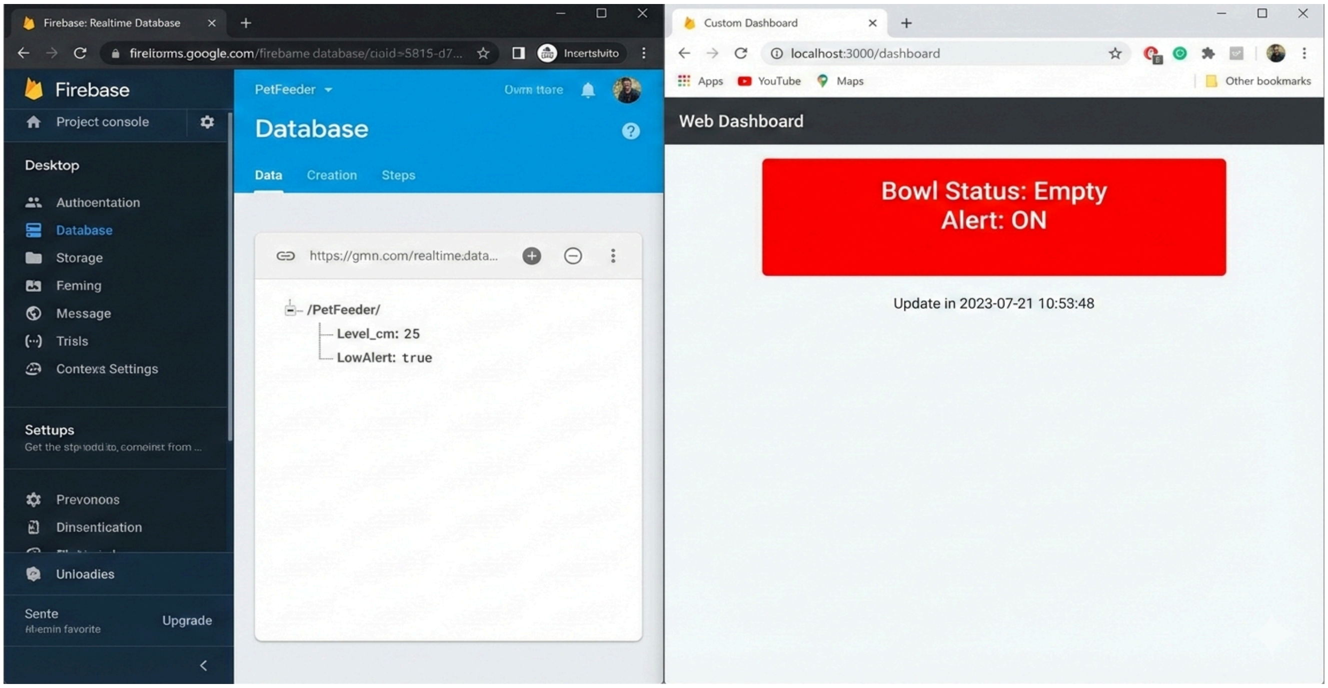
- isWatow = true
Activates BUZZER
Visual LED indicator (optionall)
If distance > THRESHCLOW,
isWater Low monitoring

4. Results and Discussion

The project was successfully implemented and tested. The system reliably measures the bowl's content level, triggers the local buzzer, and updates the Firebase database in near real-time.

- **Serial Monitor Output:** The Arduino IDE's serial monitor shows the device connecting to Wi-Fi, reading sensor data, calculating the smoothed distance, and sending data to Firebase.
- **Firebase Database:** The data in the Firebase console updates in real-time, reflecting the sensor readings. The Level_cm and LowAlert fields change according to the bowl's state.
- **Web Dashboard:** The web dashboard accurately reflects the on-site conditions, changing its status from "Full" to "Empty" and toggling the alert indicator in perfect synchronization with the database.

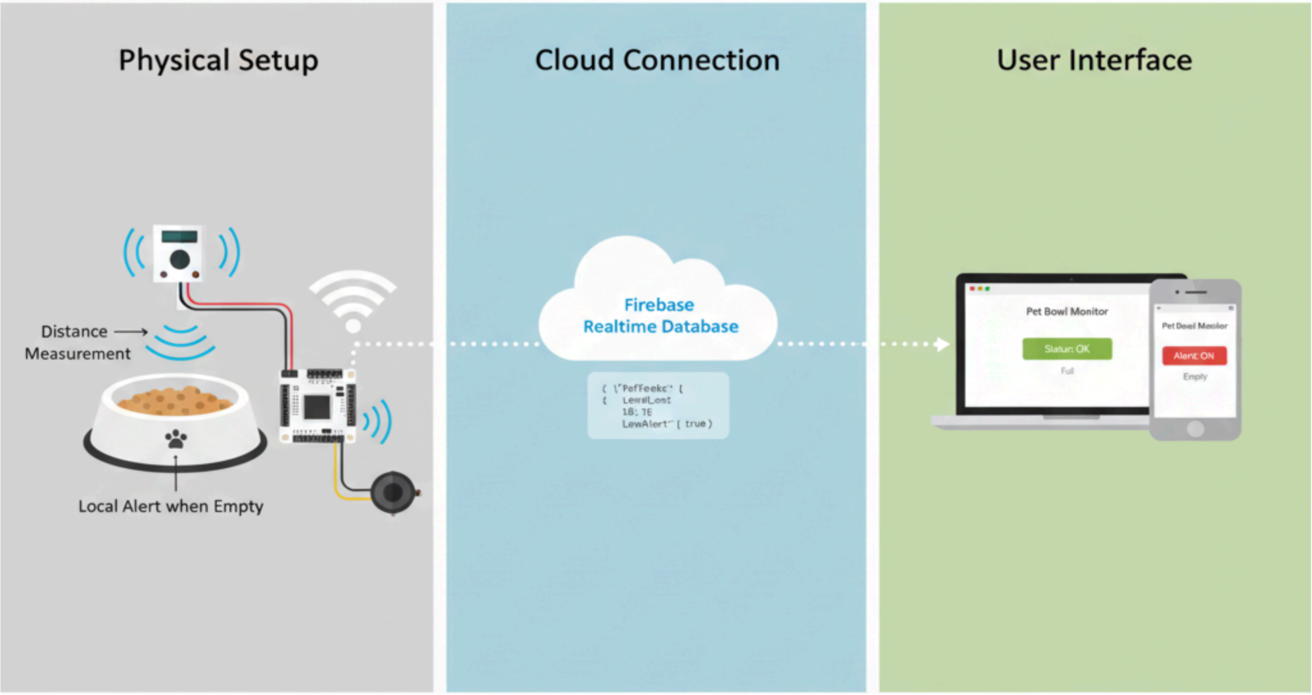
The system proved to be responsive and reliable during testing. The use of a moving average for sensor readings was crucial in preventing false alarms. The web interface provides a clean and intuitive way for the user to monitor their pet's bowl from anywhere.

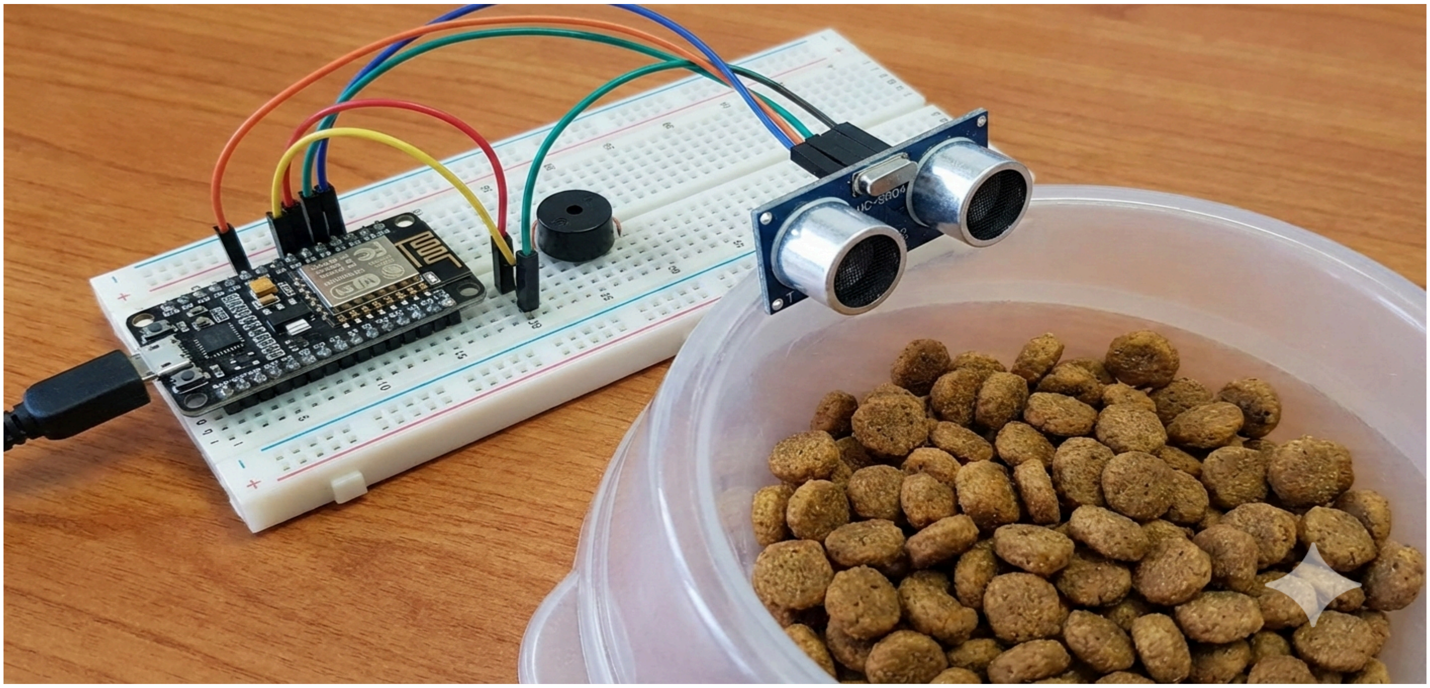


/PetFeeder/
Level_cm: 25
LowAlert: `true`

Bowl Status: Empty
Alert: ON

Update in 2023-07-21 10:53:48





5. Conclusion and Future Scope

The Smart Pet Bowl Monitor is a practical and effective IoT solution that successfully integrates hardware sensing with a cloud backend and a web-based user interface. It provides pet owners with peace of mind through a convenient and reliable remote monitoring system.

Future Enhancements:

- **Automatic Refilling:** Integrate a servo motor and a food/water reservoir to automatically refill the bowl when a low level is detected.
- **Mobile Application:** Develop a dedicated mobile app for Android/iOS for a more integrated user experience, including native push notifications.
- **Data Analytics:** Log historical data to analyze a pet's feeding/drinking patterns over time, which could provide insights into their health.
- **Camera Integration:** Add an ESP32-CAM to provide a live video feed of the pet bowl.