**Project Report: Smart Water Level Indicator using IoT and ESP32**

**1. Title**

**IoT-based Smart Water Level Indicator and Controller using ESP32**

**📄 2. Abstract**

This project presents a smart water level monitoring system using an ESP32 microcontroller integrated with an ultrasonic sensor. It provides real-time level indication, automatic water pump control, and cloud-based notifications using IFTTT. The system aims to reduce water wastage, prevent tank overflows, and enable remote monitoring via mobile or web dashboards.

**3. Objectives**

* Measure real-time water level without physical contact.
* Display water level locally and remotely.
* Alert users during low and overflow conditions.
* Automate pump control using threshold values.
* Log data to cloud platform (Firebase/IFTTT/Blynk).

**4. Hardware Components**

| **Component** | **Quantity** | **Purpose** |
| --- | --- | --- |
| ESP32 | 1 | Microcontroller with Wi-Fi |
| HC-SR04 Ultrasonic Sensor | 1 | Water level measurement |
| 5V Relay Module | 1 | Water pump control |
| DC Water Pump (12V) | 1 | Water transfer (optional) |
| OLED Display (I2C) | 1 | Local water level display |
| Buzzer + LED | 1 each | Audio-visual alerts |
| Jumper Wires + Breadboard | As needed | Circuit assembly |
| Power Supply (5V/12V) | 1 | ESP32 and pump power |
| Voltage Divider Resistors | 2 | Safe voltage input from sensor to ESP32 |

**5. Circuit Diagram**

[Insert your schematic using tools like Fritzing or draw.io]

Main Connections:

* **Ultrasonic TRIG** → GPIO 5
* **Ultrasonic ECHO** → GPIO 18 *(with voltage divider)*
* **Relay IN** → GPIO 4
* **OLED SDA/SCL** → GPIO 21/22
* **Buzzer** → GPIO 15
* **LED** → GPIO 16

**6. Software Components**

* **Arduino IDE** – Writing and uploading ESP32 code
* **Adafruit\_SSD1306** – OLED library
* **WiFi library** – ESP32 cloud connectivity
* **HTTPClient** – Triggering IFTTT alerts
* **IFTTT / Firebase / Blynk** – Cloud services for notifications and dashboards

**7. Working Principle**

1. Ultrasonic sensor measures the distance from sensor to water surface.
2. ESP32 calculates tank fill percentage using:

mathematica

CopyEdit

Water Level % = 100 - (distance \* 100 / tank depth)

1. OLED displays current level in %.
2. If level < threshold (e.g., 25%), ESP32:
   * Turns ON pump via relay
   * Activates buzzer/LED
   * Sends alert using IFTTT
3. If level > threshold (e.g., 90%), it:
   * Turns OFF pump
   * Sends overflow alert

**8. Cloud Integration Options**

* **IFTTT Webhooks**: Sends email/Telegram alerts on level breach
* **Firebase**: Stores water level data with timestamp for future analysis
* **Blynk App**: Live monitoring + manual pump control (optional)

**9. Features**

1.Real-time level display on OLED  
2. Local audio-visual alerts  
3. Automatic pump control  
4.Remote notifications (via email/Telegram)  
5.Expandable cloud dashboard  
6. Fully wireless system

**💡 10. Applications**

* Domestic water tanks
* Industrial water storage
* Overhead/underground tank monitoring
* Smart agriculture irrigation tanks

**⚙️ 11. Future Enhancements**

* **Water quality sensing** (TDS, turbidity)
* **Voice control** via Google Assistant or Alexa
* **LoRa module** for remote rural tanks
* **Battery backup** for power outages
* **Mobile app** with full control and analytics

**12. Results**

The system successfully measured water level and automated the pump. Alerts were received correctly at threshold levels, and level data was displayed in real time.

**13. Testing Table**

| **Scenario** | **Input Water Level** | **Pump State** | **Alert Sent** | **OLED Display** |
| --- | --- | --- | --- | --- |
| Normal | 50% | OFF | No | 50% |
| Low Level | <25% | ON | Yes | 20% |
| Overflow | >90% | OFF | Yes | 95% |

**14. Conclusion**

This project demonstrates a reliable, smart water monitoring system using IoT and embedded technology. It minimizes human effort, prevents water wastage, and allows remote access—making it ideal for homes, buildings, and farms.

**📎 15. References**

* ESP32 Docs
* [Arduino Libraries](https://www.arduino.cc/)
* IFTTT Webhooks
* [Firebase Realtime Database](https://firebase.google.com/)
* [Blynk IoT Platform](https://blynk.io/)