**Internet of Things Fundamentals**

*Subject Project*

BS AI 6th Smester SP-25 (AIE-3079)

Date: 24-06-2025

**Project Title:**

IoT Based water level monitoring system

**Group Name/no.:**

**4/AutoBots**

**Team Members:**

|  |  |  |  |
| --- | --- | --- | --- |
| Members | Registration no | Name | Signature |
| **Member-1 (Leader)** | **22-NTU-CS-1354** | **Maryam Sameen** |  |
| **Member-2** | **22-NTU-CS-1364** | **Muhammad Kaif** |  |
| **Member-3** | **22-NTU-CS-1358** | **Muhammad Abdullah** |  |
| **Member-4** |  |  |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Contributions in % of each Team Members for each component | | | | | |
|  | | Member-1 | Member-2 | Member-3 | Member-4 |
| Distribution Components | | Maryam Sameen | Muhammad Kaif | Muhammad Abdullah | Name |
| Coding | ESP32-coding | 33.3% | 33.3% | 33.3% |  |
| Python Coding | 33.3% | 33.3% | 33.3% |  |
| UI Design | | 33.3% | 33.3% | 33.3% |  |
| Database | | 33.3% | 33.3% | 33.3% |  |
| Cloud Integration | | 33.3% | 33.3% | 33.3% |  |
| IoT Gateway | |  |  |  |  |
| Edge Processing | | 33.3% | 33.3% | 33.3% |  |
| Documentation | | 33.3% | 33.3% | 33.3% |  |
| Presentation  Design | | 33.3% | 33.3% | 33.3% |  |
| Replace for other contribution | |  |  |  |  |
| Replace for other contribution | |  |  |  |  |
| Replace for other contribution | |  |  |  |  |
| Replace for other contribution | |  |  |  |  |

*To be filled by the evaluator*

# Team-Based Evaluation (60 Marks)

|  |  |  |
| --- | --- | --- |
| Criteria | Obtained Marks | Out of |
| System Design & Architecture |  | 10 |
| Hardware Integration & Circuit Setup |  | 10 |
| IoT Gateway and Cloud Communication |  | 10 |
| Working Prototype Demonstration |  | 10 |
| Performance & Reliability Testing |  | 10 |
| Presentation |  | 10 |
| Total (Team-Based) |  | 60 |

# Individual-Based Evaluation (40 Marks per Member)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Member 1 | Member 2 | Member 3 | Member 4 |
| Criteria |  |  |  |  |
| Understanding of the Project & Role | /10 | /10 | /10 | /10 |
| Code Contribution and Explanation | /10 | /10 | /10 | /10 |
| Q/A VIVA | /10 | /10 | /10 | /10 |
| Documentation/Reporting & Communication | /10 | /10 | /10 | /10 |
| Total (Individual-Based) | /40 | /40 | /40 | /40 |
| Total Overall (60+40) | /100 | /100 | /100 | /100 |
| Weightage Lab Grade (50) |  |  |  |  |

# 1. Abstract / Executive Summary

### 1.1 Overview

This project introduces an automated IoT-based smart water level monitoring system enhanced with machine learning, which can intelligently manage water usage and predict trends.

### 1.2 Objectives

The objective of this project is to create an automated system to measure and monitor water levels and predict correct usage of water using ML.

# 2. Table of Contents

1. Abstract / Executive Summary  
2. Table of Contents  
3. Introduction  
4. Literature Review  
5. Methodology / System Design  
6. Implementation  
7. Results & Discussion  
8. Testing & Validation  
9. Conclusion & Future Work  
10. References  
11. Links

# 3. Introduction

3.1 Background & motivation

Water scarcity and wastage are global concerns. Traditional tank monitoring systems require manual inspection, which is inefficient and error prone. This project introduces an automated IoT-based smart water level monitoring system enhanced with machine learning, which can intelligently manage water usage and predict trends.

3.2 Problem statement

Manual monitoring and control of water levels in tanks often leads to wastage, overflows, and dry tanks. This project aims to solve this by developing an IoT-based smart water level monitoring system with real-time alerts, motor automation, and on-device ML prediction.

### 3.3 Project goals

* Monitor real-time water levels using an ultrasonic sensor
* Automate water pump control based on predefined thresholds
* Provide local alerts using a buzzer and OLED display
* Use machine learning to predict water usage patterns and tank status
* Store and visualize data using InfluxDB, and Grafana

# 4. Literature Review (Optional)

Relevant IoT/ESP32 concepts  
Similar projects/research

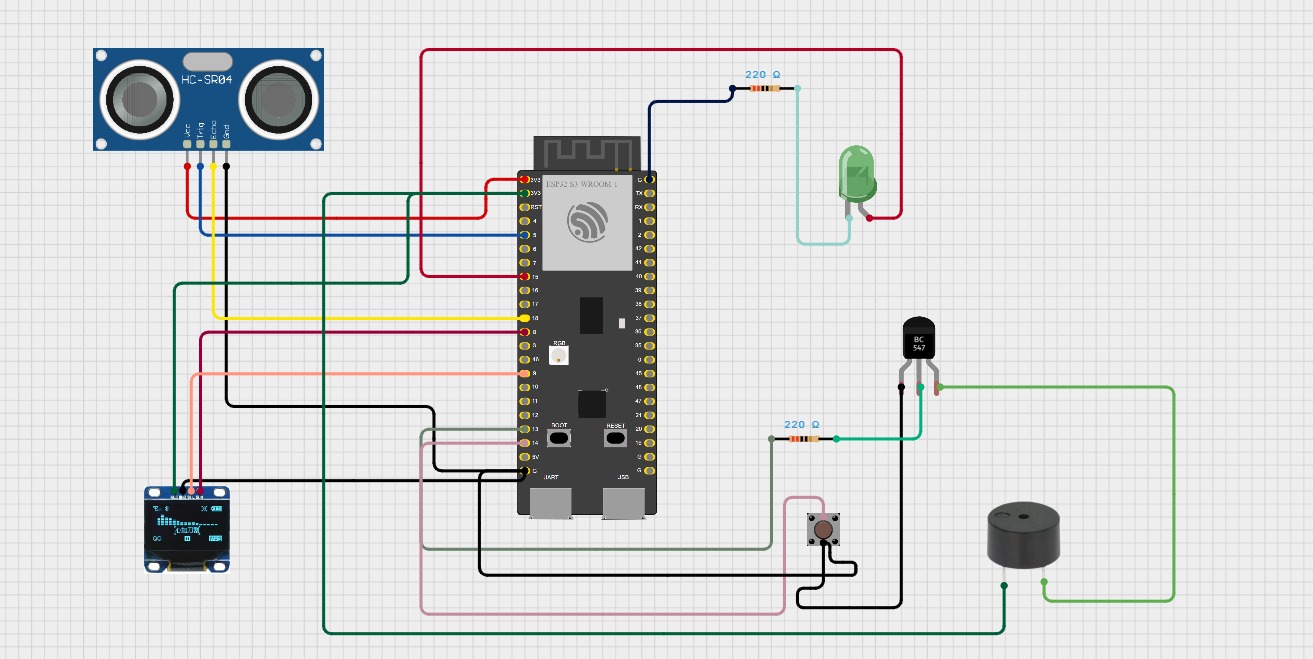
# 5. Methodology / System Design

## 5.1 Hardware Components

### 5.1.1 Components:

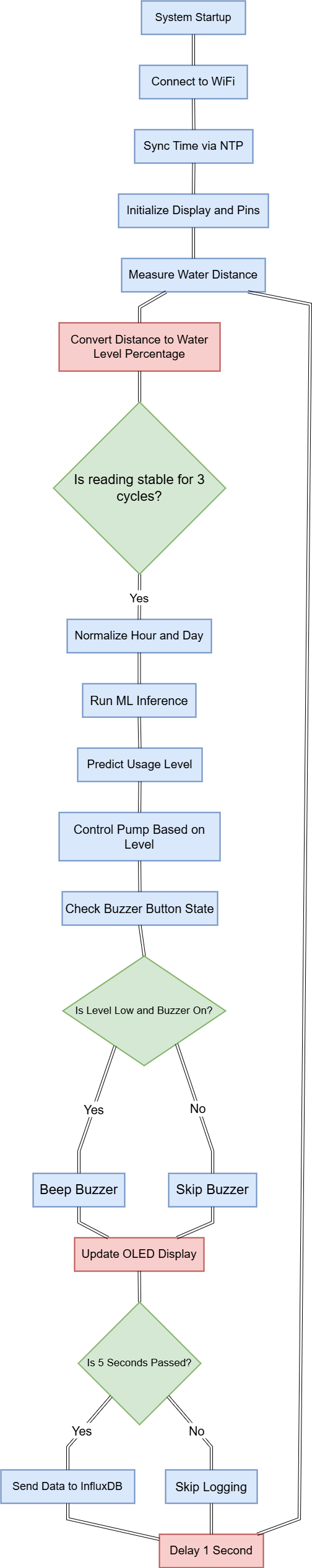
* Ultrasonic Sensor (HC-SR04) for measuring distance from water surface
* ESP32-S3 microcontroller as the central processing unit
* OLED Display to show live tank level and status
* LED For alerts
* Buzzer for high/low-level alerts

### 5.1.2 Circuit Diagram:



## 5.2 Software Design

### **5.2.1 Flowchart/system architecture**



### **5.2.2 Libraries/tools used (Arduino IDE, PlatformIO, MQTT, etc.)**

* **Arduino IDE** – For writing, compiling, and uploading firmware to the ESP32-S3 board.
* **Python**  – For Machine Learning prediction.
* **InfluxDB** – For storing timestamped sensor data.
* **Grafana** – For visualizing water level data on dashboards.
* **TensorFlow Lite Micro** – For running the ML model on ESP32 (converted .tflite to C header file).

### 5.2.3 Pseudocode (if applicable)

START

1. Initialize Libraries and Components

- Connect to Wi-Fi

- Initialize OLED display

- Initialize ultrasonic sensor (TRIG and ECHO pins)

- Initialize buzzer, button, pump relay, LED

- Load TFLite model into memory

- Sync time using NTP

2. Define Constants

- Tank height, minimum sensor distance

- Buzzer interval time

- Normalization constants for hour and weekday

- InfluxDB endpoint and token

3. LOOP (Runs continuously)

A. If Button is pressed and debounce interval passed:

- Toggle buzzer ON/OFF state

- Print buzzer state to Serial

B. Read Distance from Ultrasonic Sensor:

- Take multiple readings (e.g., 10)

- Calculate the mode of the valid readings

C. Convert Distance to Water Level Percentage:

- Use: level = ((tankHeight + minDistance - distance) / tankHeight) \* 100

D. If level is stable for 3 readings:

- Store it as lastValidLevel

1. Get current hour and weekday

2. Normalize input features

- hourNorm = (hour - mean\_hour) / scale\_hour

- dayNorm = (day - mean\_day) / scale\_day

3. Run Inference with ML Model

- Input: [hourNorm, dayNorm]

- Output: water usage prediction class

4. Interpret Prediction:

- Map class index to usage level (e.g., Low, Medium, High)

5. Display Data on OLED:

- Show water level %, usage level, and tank status

6. Control Actuators:

- If level <= 10%

- Turn ON pump

- Beep buzzer if buzzer is enabled and interval passed

- If level >= 70%

- Turn OFF pump

- Else: Keep buzzer OFF

7. Send Water Level to InfluxDB using HTTP POST

E. Wait 1 second before next loop

END

# 6. Implementation

6.1 Step-by-step setup (wiring, configurations)

**1. Connect the HC-SR04 Ultrasonic Sensor**

* **VCC** → 5V on ESP32
* **GND** → GND on ESP32
* **TRIG** → GPIO5 on ESP32
* **ECHO** → GPIO18 on ESP32

**2. Connect the OLED Display (SSD1306 - I2C)**

* **VCC** → 3.3V on ESP32
* **GND** → GND on ESP32
* **SDA** → GPIO8 on ESP32
* **SCL** → GPIO9 on ESP32

**3. Connect the Buzzer**

* **Buzzer + (VCC)** → GPIO13 on ESP32
* **Buzzer + (GND)** → GND on ESP32

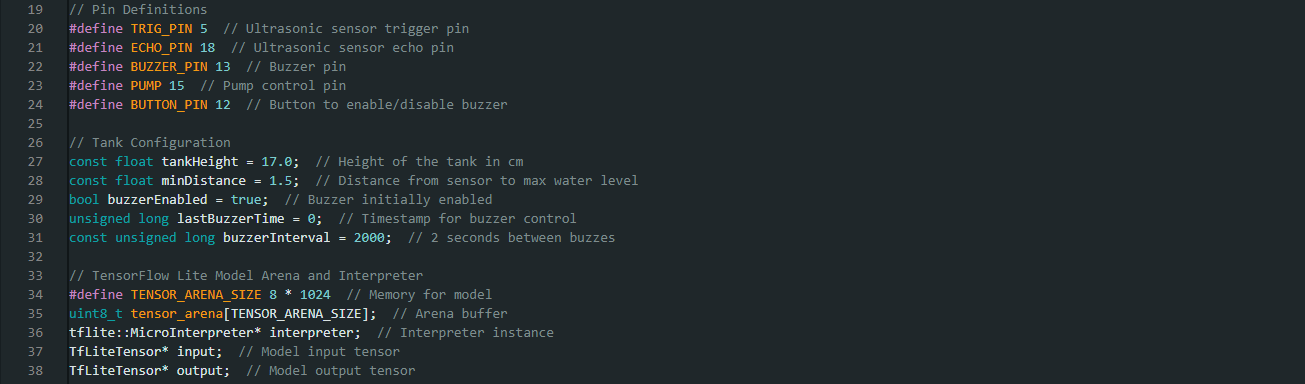
**4. Connect LED with resistor**

* **LED Anode** → GPIO2 on ESP32 via 220ohm resistor
* **Resistor end** → connected directly to GPIO2 output pin
* **LED Cathode** → Connected to GND

**5. Push Button**

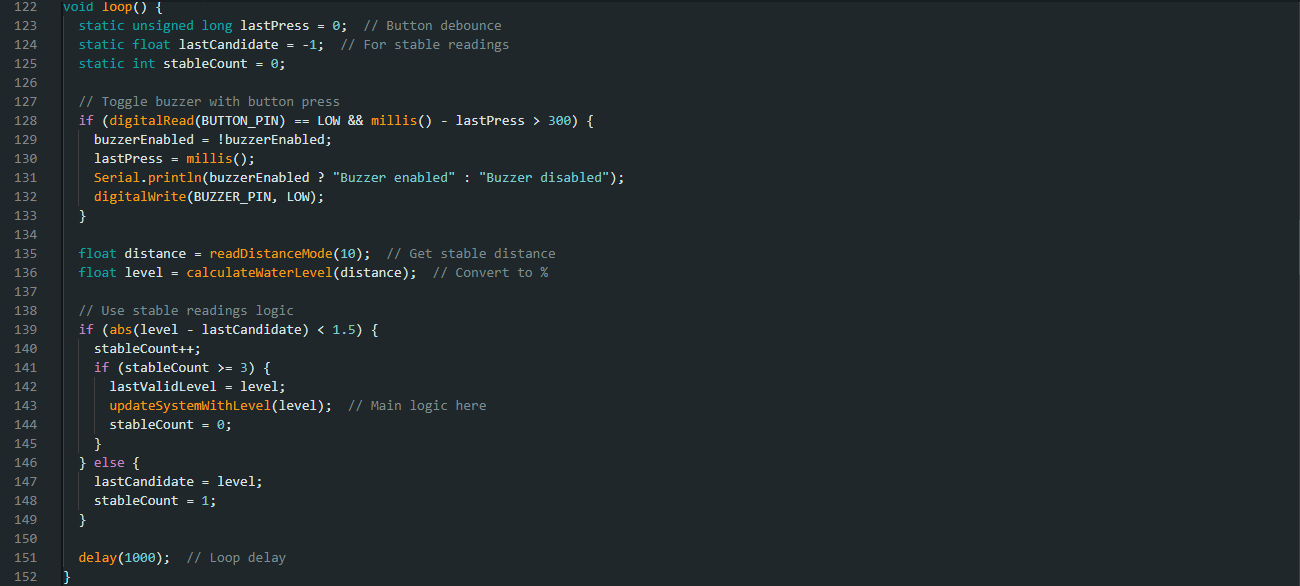
* **One terminal →** GPIO12
* **Other terminal →** GND
* **Internal pull-up resistor** used in code **(INPUT\_PULLUP)**

6.2 Code snippets (with comments)



A computer screen shot of a computer code

AI-generated content may be incorrect.



A screen shot of a computer program

AI-generated content may be incorrect.  
6.3 Challenges & solutions

**1. WiFi Dependence**

* **Problem:** WiFi signals can be weak or unreliable in factories or remote sites.
* **Solution:** Use more robust communication methods like LoRa, NB-IoT, or Ethernet.

**2. No Local Gateway or Buffering**

* **Problem:** If WiFi disconnects, the ESP32 cannot store or forward data - resulting in data loss.
* **Solution:** Add a **local gateway** (e.g., Raspberry Pi) that temporarily stores data and forwards it to the cloud when reconnected.

**3. Single Point of Failure**

* **Problem:** The entire system relies on one ESP32. If it fails, the whole monitoring system stops.
* **Solution:** Deploy **backup microcontrollers** or add **redundancy** for critical operations.

**4. Limited Alerting Mechanism**

* **Problem:** Alerts are only visible via OLED or buzzer; no remote notifications are sent.
* **Solution:** Integrate **mobile alerts** using platforms like **Firebase**, **Blynk**, or **Twilio** for SMS/email.

**5. Limited ML Model Scalability**

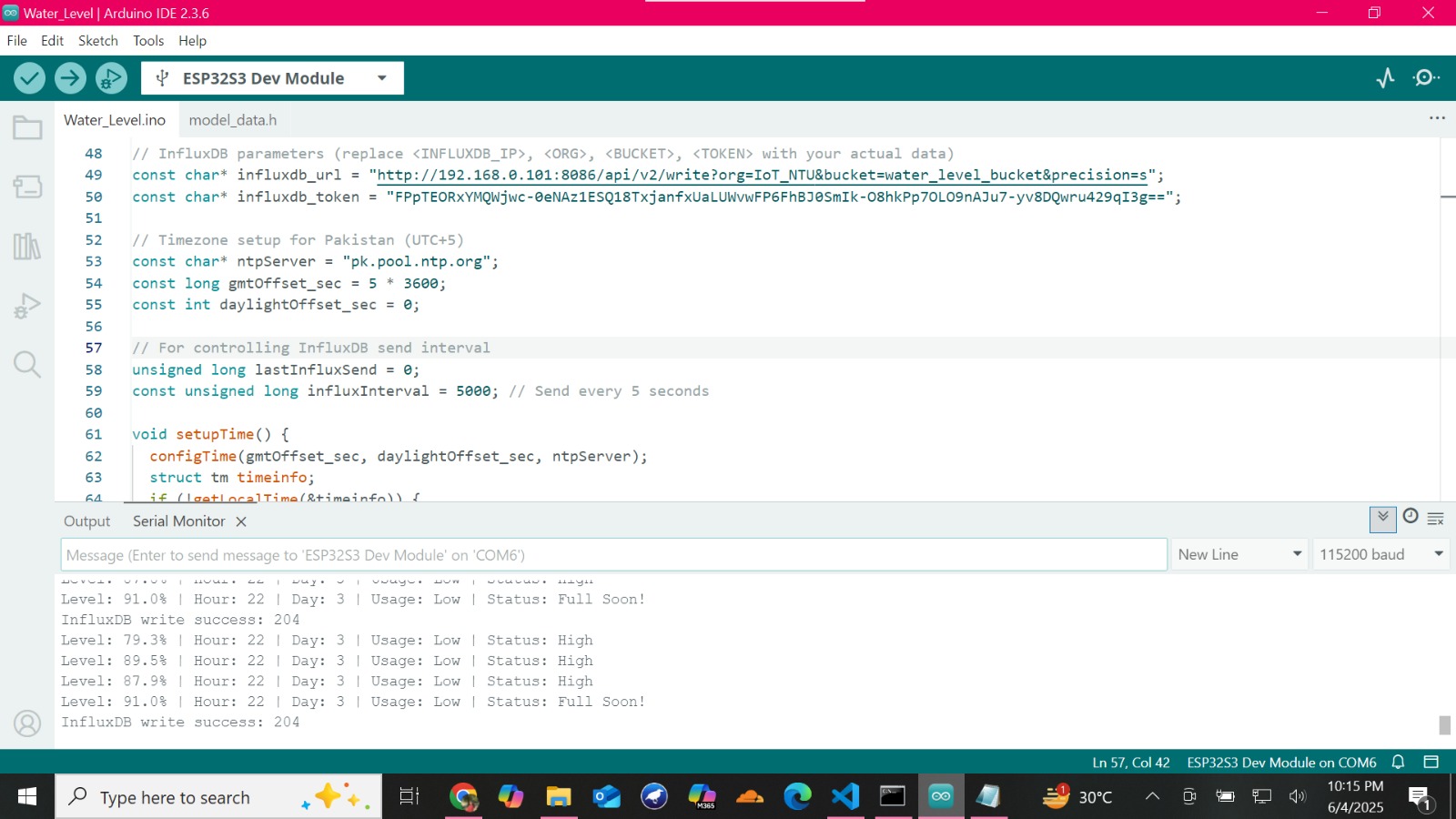
* **Problem:** The ML model is trained for a specific usage pattern and may not work well across different locations or habits.
* **Solution:** Enable **periodic retraining** using cloud-collected data or apply **federated learning** to improve across all devices.

# 7. Results & Discussion

* Successfully implemented an **Edge ML + IoT hybrid system**
* Achieved **accurate level monitoring**, **smart automation**, and **real-time cloud logging**
* System is fully functional and modular, ready for enhancement (like remote alerts or sensor upgrades)

### 7.1 Screenshots/output (e.g., sensor data on Serial Monitor, MQTT logs)

* **Serial Monitor**

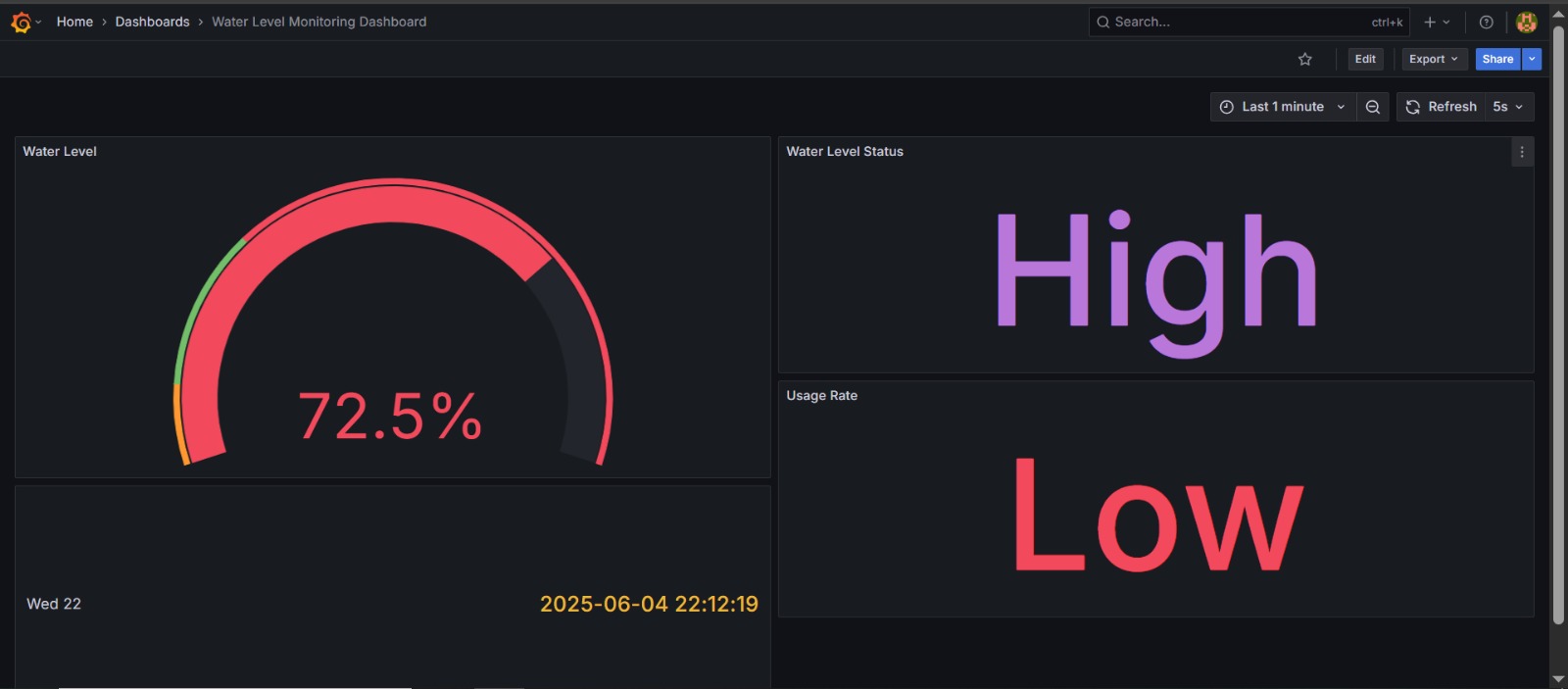


* **Influx DB**

A screenshot of a computer

AI-generated content may be incorrect.

* **Grafana Dashboard**



Performance analysis (accuracy, latency, reliability)  
Comparison with expected outcomes

# 8. Testing & Validation / Limitations

### 8.1 Limitations

1. **Sensor Instability**
   * Ultrasonic sensor may give inaccurate readings in narrow or turbulent tanks.
2. **Wi-Fi Dependency**
   * System relies on a stable internet connection for MQTT and cloud updates.
3. **Limited ML Complexity**
   * ESP32-S3 can only run lightweight models due to memory and processing limits.
4. **No Mobile App**
   * Currently lacks a user interface for remote manual control or live monitoring.

# 9. Conclusion & Future Work

# 

9.1 Key takeaways

* Edge AI on ESP32 predicts water usage intelligently.
* Water level is monitored using an ultrasonic sensor.
* Pump and buzzer are automatically controlled based on level.
* OLED shows real-time status; buzzer toggle via button.
* Data is sent directly to InfluxDB using HTTP (no MQTT).
* ML model achieves ~94% training accuracy.
* Safe circuit design using transistor for buzzer control.

### 9.2 Potential improvements (e.g., adding AI, cloud integration)

* Integrate MQTT protocol for real-time cloud-based communication and control.
* Enhance the Edge AI model with additional features like time-series trends or environmental factors.
* Add mobile push notifications using Firebase for low/high water level alerts.
* Enable Over-the-Air (OTA) firmware updates for easier remote maintenance.
* Include water flow sensors and anomaly detection using AI for leak detection.

# 10. References

<https://youtu.be/9geREeE13jc?si=JR47kx6lNE-m9OeA>

# 11. Links

GitHub Repository Link (links from each member)  
Video Demo (embedded link or QR code / optional with Bonus)