WWW.IJRPR.COM

International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

IoT-Based Intravenous Fluid Level Monitoring and Low-Level Alert System Using ESP32 and a Load Cell

Varshini R[1],Rudreshwari S[2],Sinchana M[3],Pooja D[4],Dr. Sundresha M P[5]

Department of Electronics and Communication Engineering, Kalpataru Institute of Technology, Tiptur 572201, Karnataka, India Affiliated to Visvesvaraya Technological University,Belgavi,Karnataka,India

A B S T R A C T :

Manual monitoring of intravenous (IV) fluid containers requires repeated visual checks by nursing staff and may delay the identification of a nearly empty container. This paper presents the design and prototype implementation of a low-cost, weight-based IV fluid monitoring and alert system built around an ESP32 microcontroller. A load cell with an HX711 24-bit amplifier measures the weight of the IV container; a predefined empty-container weight of 33 g is subtracted to obtain the fluid weight. A push button records the current fluid weight as the 100% reference, after which the ESP32 continuously computes the remaining fluid percentage, shows it on a 16×2 I2C LCD, and transmits the percentage and a status string (NORMAL/LOW) to the Blynk IoT platform over Wi-Fi through virtual pins V0 and V1. When the remaining fluid reaches 10% or below, the buzzer is activated and a LOW FLUID warning is displayed. The firmware, wiring, and functional behaviour of the prototype are described. Quantitative measurement accuracy, alert latency, and communication reliability have not yet been characterised, and the system has not been clinically validated; it is therefore presented as a proof-of-concept prototype for further evaluation.

Keywords: IV fluid monitoring; ESP32; load cell; HX711; Internet of Things; Blynk

Introduction

Intravenous (IV) therapy is used in hospitals and healthcare centres to deliver fluids, medicines, and nutrients directly into the bloodstream. Continuous administration depends on timely replacement of the fluid container; in routine practice this is achieved by nurses or attendants who periodically inspect the container and the drip chamber. Such inspection is labour-intensive, particularly when several patients receive infusions simultaneously, and a container that is not checked in time may run low or empty before it is noticed.

Advanced infusion pumps and automated monitoring devices can control or monitor delivery, but they may be costly and are not necessarily available for every patient or in every facility. Gravity-fed IV sets, which are widely used, do not by themselves provide a low-fluid alarm or any remote indication of the remaining volume. This gap motivates inexpensive sensing systems that make the remaining fluid quantity visible and generate a warning at a defined threshold.

The present work describes a prototype that addresses this need using commonly available components: an ESP32 microcontroller, a strain-gauge load cell with an HX711 amplifier, a 16×2 I2C LCD, a push button, a buzzer, and the Blynk IoT platform. The principal contributions of this work, as documented in the project, are: (i) an integrated weight-based monitoring prototype that reports the remaining fluid weight and percentage locally and remotely; (ii) a simple push-button procedure that sets the current fluid quantity as the 100% reference, together with subtraction of a predefined 33 g empty-container weight; and (iii) a threshold alert that activates a buzzer, an LCD warning, and a remote status change at 10% remaining. The system monitors and alerts only; it does not control fluid flow.

The remainder of this paper is organised as follows: Section 2 reviews related work, Section 3 states the problem and objectives, Section 4 describes the methodology, Section 5 presents the system design and implementation, Section 6 discusses the results available so far, Section 7 lists limitations, and Section 8 concludes with future work.

Related Work

Several recent studies have applied load-cell sensing and IoT communication to IV fluid monitoring. Afiyat et al. [1] developed a monitoring system based on a NodeMCU ESP8266, an HX711 module, and the MQTT protocol, with remote display through the MQTT Dash application. They evaluated it with a 500 mL Ringer's lactate infusion and reported that the load cell measured fluid weight with relatively low error. The approach relies on network connectivity and sensor calibration, and requires MQTT-specific configuration.

Nayak et al. [2] proposed an IoT-based IV fluid level indicator using a NodeMCU, load cell, HX711, and the Blynk platform, in which healthcare staff observe the fluid level remotely and receive notifications on a low-fluid condition. Its performance depends on calibration, wireless availability, and stable positioning of the container. Paranthaman and Annalakshmi [3] used an ESP32 with a load cell, HX711, 16×2 LCD, buzzer, and Blynk, and

additionally included a solenoid valve to stop the flow automatically at a critical level; this adds hardware complexity and introduces further safety and reliability requirements. Vasundara et al. [4] described a low-cost ESP32-based system that compares the measured fluid weight with a predefined threshold and generates a warning, and noted that additional testing is required before clinical use.

Table 1 - Comparison of related systems and the present prototype (summarised from the cited works).

System	Controller	Sensing	Remote interface	Local display / alert	Flow control
[1]	NodeMCU ESP8266	Load cell + HX711	MQTT / MQTT Dash	Not stated	No
[2]	NodeMCU	Load cell + HX711	Blynk (notifications)	Not stated	No
[3]	ESP32	Load cell + HX711	Blynk	16×2 LCD, buzzer	Solenoid valve
[4]	ESP32	Load cell + HX711	IoT (platform not stated)	Warning (type not stated)	No
This work	ESP32	Load cell + HX711	Blynk (V0, V1)	16×2 LCD, buzzer	No

All reviewed works agree on the basic principle of weight-based sensing with an HX711 front end and on the dependence of accuracy on calibration and mounting. The reviewed systems differ in controller, communication protocol, and in whether automatic flow control is included. **Research gap:** the cited studies, particularly [3], share most of the hardware used here, so the contribution of this work is not a new sensing principle. It is a minimal monitoring-and-alert configuration with push-button reference initialisation, container-weight compensation, and a single defined alert threshold, without flow control. **[Supervisor: confirm that this positioning is acceptable and, if a stronger novelty claim is intended, specify it (e.g., a measured accuracy or cost advantage).] [Verify all four citations against the original publications before submission; this review is based only on the summaries in the project report.]**

Problem Statement and Objectives

Problem statement. Manual observation of IV fluid containers provides no automatic low-fluid alert and no remote visibility of the remaining quantity, which increases the workload of healthcare staff and may delay container replacement. The problem addressed is the design of an inexpensive embedded system that measures the remaining IV fluid quantity continuously, presents it locally and remotely, and warns staff when it falls to a critical level.

The research objectives are:

- to build an automated IV fluid monitoring prototype around an ESP32 microcontroller;
- to measure container weight with a load cell and HX711 module and obtain fluid weight by subtracting the 33 g empty-container weight;
- to compute the remaining fluid percentage relative to a user-initialised 100% reference;
- to display fluid weight and percentage on a 16×2 LCD and publish them to the Blynk IoT application;
- to activate a buzzer and display a warning when the remaining fluid is 10% or below.

Scope and significance. The work covers a single-container prototype used for monitoring and alerting. It does not regulate flow rate or stop infusion, and it is not a certified medical device. Its significance lies in showing that a low-cost, ESP32-based design can make fluid level visible and flag low-fluid conditions without frequent visual checks.

Proposed Methodology

1.1. System architecture

The system comprises five functional blocks: sensing (load cell and HX711), processing (ESP32), local interface (LCD and push button), alerting (buzzer), and cloud monitoring (Blynk over Wi-Fi). The IV container is supported by the load cell; the HX711 amplifies the differential bridge signal and digitises it with 24-bit resolution; the ESP32 reads the data through a two-wire (DT/SCK) interface and performs all computation and control.

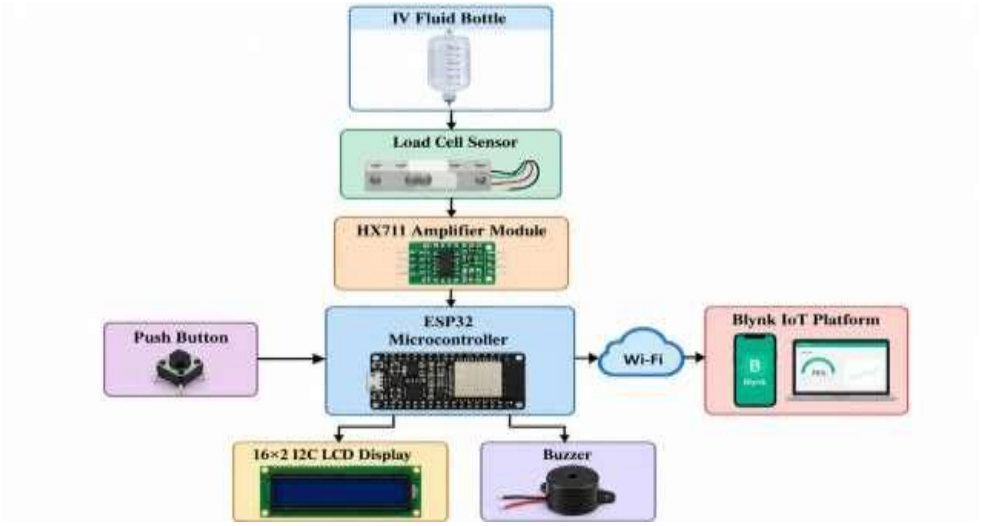


Fig. 1 - Block diagram of the proposed monitoring system.

1.2. Fluid-level computation

At start-up the load cell is tared with no load. The fluid weight is obtained from the measured total weight Wm by subtracting the empty-container weight Wb (33 g):

$$Wc = \max(0,\ Wm - Wb) \qquad (1)$$

When the user presses the button with the filled container in place, the current fluid weight is stored as the initial weight Wi. The remaining fluid percentage is then

$$P = (Wc / Wi) \times 100,\ \ 0 \le P \le 100 \qquad (2)$$

where P is the remaining fluid percentage (%), Wc is the current fluid weight (g), and Wi is the initial fluid weight (g); P is limited to the range 0–100%. A low-fluid condition is declared when $P \le 10\%$.

1.3. Operating algorithm

The firmware executes the following sequence, repeated every second by a software timer:

- Initialise HX711, LCD, button (internal pull-up), buzzer, and Wi-Fi/Blynk; tare the scale.
- Read the weight (average of five samples) and compute Wc by Eq. (1).
- If monitoring has not started, display a prompt; when the button is pressed and $Wc > 0$, set $Wi = Wc$ and $P = 100\%$.
- If monitoring has started, compute P by Eq. (2); if $P \le 10\%$, switch the buzzer on and show LOW FLUID, otherwise keep the buzzer off.
- Update the LCD and, if connected, send P to virtual pin V0 and the status (WAITING, NORMAL, or LOW) to V1.
- If the HX711 is not ready, switch the buzzer off and display SENSOR ERROR.

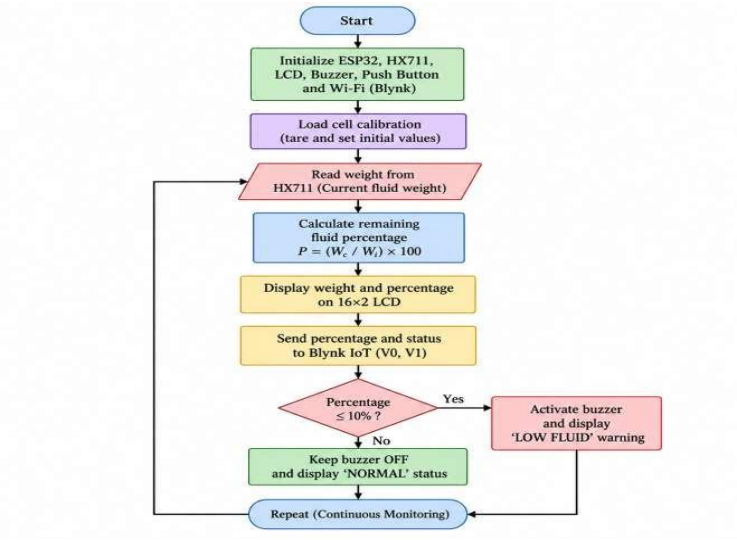


Fig. 2 - Flowchart of the monitoring algorithm.

System Design and Implementation

1.4. Hardware

The ESP32 development board (32-bit, up to 240 MHz, integrated 802.11 b/g/n Wi-Fi, 3.3 V logic, powered from 5 V USB) is the central unit. A strain-gauge load cell (typical capacity variants of 1, 5, or 10 kg; **[state the capacity used])** produces a small differential signal that the HX711 (24-bit ADC, 2.6–5.5 V supply, programmable-gain amplifier) converts to digital values. A 16×2 character LCD with an I2C backpack (address 0x27) shows weight, percentage, and warnings. A push button on GPIO 27 with the internal pull-up sets the 100% reference, and a buzzer on GPIO 25 provides the audible alert. Table 2 lists the connections.

Table 2 - ESP32 pin assignments.

Module	Signal	ESP32 pin
HX711	DT (data)	GPIO 18
HX711	SCK (clock)	GPIO 19
I2C LCD	SDA	GPIO 21
I2C LCD	SCL	GPIO 22
Push button	Input (pull-up)	GPIO 27
Buzzer	Output	GPIO 25 **[confirm; Fig. 4.2.6 of the report labels GPIO 26]**

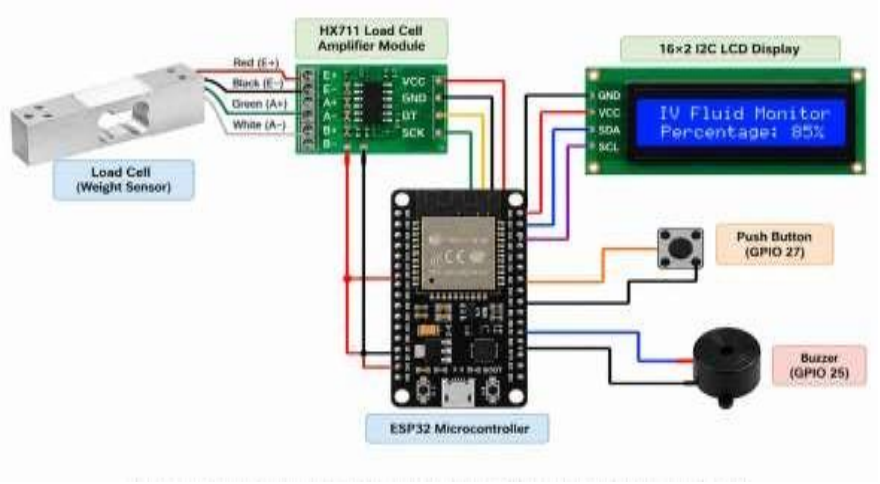


Fig. 3 - Circuit diagram of the prototype.

1.5. Software

The firmware is written in embedded C/C++ in the Arduino IDE using the ESP32 board support package, the HX711 library (bogde) [8], the LiquidCrystal_I2C library, the Wire library [7], and the Blynk library [5], [6]. Calibration uses a scale factor set through the library (362.7 in the reported listing) and a tare at start-up. Five-sample averaging is used for each reading to improve stability. Blynk virtual pin V0 carries the percentage and V1 carries the status text; Internet access is required for remote monitoring. The complete listing is given in the project **report [decide whether to include the code as an appendix; remove credentials before doing so].**

1.6. Assembly

As shown in Fig. 4, the prototype is mounted on a wooden base with an IV stand; the IV container is positioned on the load cell, and all modules are connected with jumper wires. The ESP32 is powered from a regulated USB source.

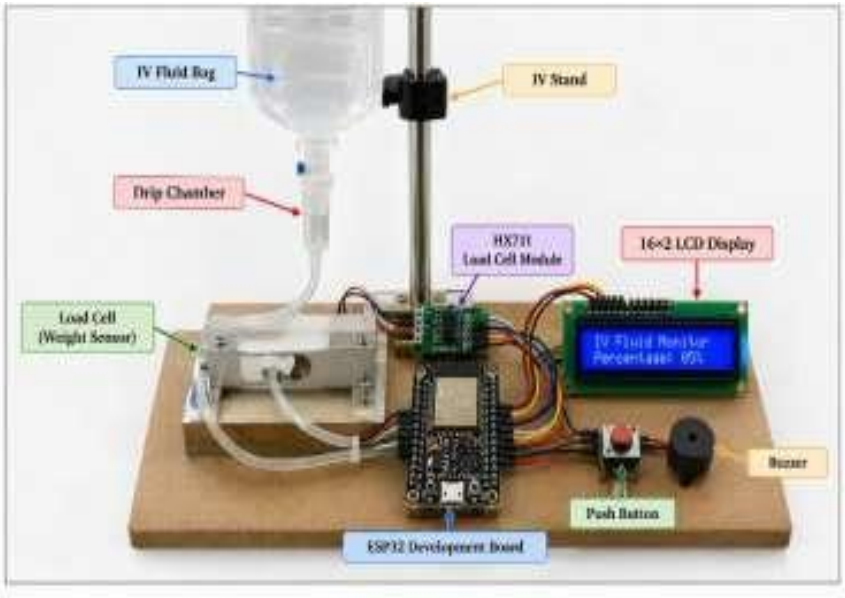


Fig. 4 - Assembled hardware prototype.

Results and Discussion

The project report describes the intended and implemented functions of the prototype: weight monitoring, push-button initialisation, LCD display, buzzer alert at P ≤ 10%, and Blynk monitoring with a NORMAL or low-fluid status. However, the report states that actual test results are still to be confirmed by hardware readings and screenshots, and it contains no numerical measurements. Accordingly, no accuracy, latency, or reliability figures are reported here, and none are inferred.

Table 3 - Functional verification summary (to be completed with recorded results).

Test	Expected behaviour	Observed result
Load cell / HX711	Reading changes with applied known loads	[Provide readings]
LCD	Shows weight, percentage, prompt, LOW FLUID	[Provide photographs]
Push button	Current fluid weight becomes 100%	[Provide LCD photograph]
Buzzer	OFF above 10%; ON at or below 10%	[Provide observation]
Blynk	V0 = percentage; V1 = NORMAL / LOW	[Provide dashboard screenshots]

Table 4 - Weight measurement accuracy (data required).

Known mass (g)	HX711 reading (g)	Absolute error (g)	Error (%)
[ ]	[ ]	[ ]	[ ]
[ ]	[ ]	[ ]	[ ]
[ ]	[ ]	[ ]	[ ]

Discussion. Architecturally, the prototype provides local and remote fluid-level visibility and an automatic low-fluid warning with few components. Compared with the related work in Table 1, its configuration is closest to [3] but omits flow control, which reduces hardware complexity and safety requirements. A quantitative comparison with existing systems is not possible because the cited studies' measurements were not available in the report

and the prototype's own measurements have not been recorded. The 10% threshold and 33 g container weight are design parameters taken from the project and should be justified clinically (e.g., relative to infusion rate and staff response time) before any claim of timely intervention is made.

Limitations

- Remote monitoring requires Wi-Fi and Internet access, and the system requires continuous power; no battery backup is included.
- Accuracy depends on load cell calibration, stable mounting, and freedom from vibration and tubing tension, none of which has been quantified.
- The percentage depends on the user pressing the button with the container correctly positioned; a premature or erroneous press sets a wrong reference.
- The empty-container weight is a fixed constant (33 g) and does not accommodate other container types.
- Only a single container is monitored, and flow is neither measured nor controlled.
- No experimental accuracy data, long-duration tests, or clinical validation have been reported.

Conclusion and Future Work

This paper described a low-cost ESP32-based prototype that monitors the remaining IV fluid by weight, displays it on an LCD, publishes it to Blynk, and raises a buzzer and LCD alert at 10% remaining. The design and firmware logic are complete, but quantitative performance is not yet documented, so the work should be regarded as a proof of concept pending experimental evaluation and, for any clinical use, safety validation under applicable medical-device requirements.

Future work includes: completing a calibration and accuracy study; adding push notifications through Blynk events; extending to multiple containers with a central dashboard; improving accuracy through filtering and better mounting; adding battery backup; estimating time-to-empty from consumption trends; and, only with rigorous validation, investigating flow-control mechanisms.

References

[1] N. Afiyat, R. H. Navilla, and M. Hariyadi, "IoT-based infusion fluid monitoring system using the MQTT protocol," Jurnal Nasional Teknik Elektro dan Teknologi Informasi, 2023**. [verify volume, pages, DOI]**

[2] S. Nayak, A. Agrawal, J. Yadav, J. Zalke, and S. Pandey, "IoT-based intravenous fluid level indicator enhancing patient safety in the era of smart healthcare system," in IEEE Int. Conf. Interdisciplinary Approaches in Technology and Management for Social Innovation, 2024. **[verify pages, DOI]**

[3] A. Paranthaman and K. Annalakshmi, "IV fluid bag monitoring, alert, and control system using ESP32 and IoT," Int. J. Science and Technology, 2025**. [verify volume, pages, DOI]**

[4] S. Vasundara, Chinnamani, S. Pooja Sri, Rajashree, and J. Sathishkumar, "Design and development of a low-cost IoT-based IV fluid monitoring and alert system," Int. J. Information and Electronics Engineering, 2026. **[verify volume, pages, DOI]**

[5] Blynk Inc., "Blynk virtual pins documentation." [Online]. Available: https://docs.blynk.io/en/blynk-library-firmware-api/virtual-pins

[6] Blynk Inc., "Send data from hardware to Blynk." [Online]. Available: https://docs.blynk.io/en/getting-started/how-to-display-any-sensor-data-in-blynk-app

[7] Arduino, "Wire library documentation." [Online]. Available: https://docs.arduino.cc/language-reference/en/functions/communication/wire/

[8] Bogde, "HX711 Arduino library." [Online]. Available: https://github.com/bogde/HX711

[9] Espressif Systems, "ESP32 series datasheet." [Online]. Available: https://documentation.espressif.com/esp32_datasheet_en.html

[10] SparkFun Electronics, "Load cell amplifier HX711 breakout hookup guide." [Online]. Available: https://learn.sparkfun.com/tutorials/load-cell-amplifier-hx711-breakout-hookup-guide

[11] Espressif Systems, "Arduino core for ESP32." [Online]. Available: https://docs.espressif.com/projects/arduino-esp32/en/latest/

[12] Arduino, "Arduino IDE software." [Online]. Available: https://www.arduino.cc/en/software