# When Water Keeps Flowing: A Low-Cost ESP32 System for Preventing Household Water Waste

Mahesh Sanjay Shinde, Dinesh Balasaheb Surve, Rahil Chirag Patel, and Sujal Santosh Sahu

*Department of Mechanical Engineering, Rajarshi Shahu College of Engineering, Pune, India*

*Project Guide: Prof. P. S. Patil | B.Tech (2024-25)*

**Abstract**

This paper presents Each Drop Counts, a low-cost, ESP32-based domestic water-conservation prototype designed to allocate and safeguard a fixed water quota for each flat or household member, for example, a 500-liter daily allowance per flat, resetting automatically at 00:00 local time. Beyond simple quota tracking, the controller distinguishes ordinary, intermittent household water use from a continuous, unattended flow event. When water flows without interruption past 50 liters, the system issues a buzzer warning and closes a servo-actuated 3D-printed valve for 30 seconds; if the flow resumes and continues uninterrupted for a further 10 liters, a second warning and closure follow; and if flow persists for an additional 3 liters, the valve locks permanently, requiring an authorized Bluetooth command to reopen. This staged escalation was designed to mirror the behavior of commercial tank-filling systems that shut off automatically once filling completes, while still giving the user two clear opportunities to intervene before restriction. An emergency Bluetooth override allows an authorized user to claim an additional allowance, nominally around 200 liters, for genuine high-need situations, with this limit configurable per user requirement. A pay-to-recharge mechanism, allowing a user to top up an exhausted quota, was also considered as part of the system's intended design and is discussed as a configurable extension, though it remains unimplemented in the current firmware. A 16×2 LCD provides local feedback on accumulated volume and system state, while the flow sensor's placement upstream of the valve reduces the chance of downstream drain-down being misread as post-closure consumption. The prototype demonstrates working volume display and valve-closing behavior on hardware, built around efficient, near-complete utilization of the ESP32's onboard GPIO, I2C, PWM, and Bluetooth capabilities. The paper documents the control logic, hardware architecture, test evidence, limitations, and a roadmap toward IoT, Android, and edge-AI integration.

**Keywords —** Water conservation, ESP32 microcontroller, flow sensor, automatic valve control, flat-wise water allocation, daily quota reset, continuous-flow protection, Bluetooth emergency override, pay-to-recharge system, IoT-enabled water management.

## 1. Introduction

Water is often wasted simply because a tap or pipeline is left running and no one notices in time. In shared residential setups, this problem is compounded further: a single undifferentiated water supply for a building makes it difficult to identify which flat or household is responsible for excessive consumption, and offers no built-in mechanism to keep individual usage within a fair or sustainable limit. A meaningful domestic water-management system, therefore, needs to do more than simply display a meter reading. It should allocate water fairly across users, track that allocation over time, and, critically, intervene when water keeps flowing without a clear reason.

The 2024 United Nations World Water Development Report underscores why careful water management matters at both the individual and community level [1]. Building on this motivation, the present project was conceived with a household- and flat-level water management problem in mind, structured around several interlinked design goals. The first was to allocate a fixed daily water quota to each flat or household member individually, for example, a 500-liter daily allowance per flat, resetting automatically at 00:00 local time so that every calendar day begins with a fresh and predictable quota. A further consideration was to allow a user whose allocation has been exhausted to recharge or top up their quota through a pay-to-recharge mechanism, conceived as an optional and configurable feature for future deployment.

Beyond quota allocation, a central concern of this work was distinguishing ordinary, intermittent household water use, such as showering or filling a bucket, from a genuinely unattended, continuous flow event, such as a tap left running or a valve that fails to close after a task is complete. Prior work has shown that low-cost sensors, microcontrollers, mobile applications, and controllable valves can be effectively combined for water monitoring [2]–[4]. This project adapts these ideas toward one specific, practical task: detecting and responding to water that keeps flowing without a break, in a manner inspired by commercial tank-filling systems that shut off automatically once filling is complete. Rather than reacting only to total daily consumption, the proposed controller watches for sustained, uninterrupted flow and responds in progressive stages, first warning the user, then pausing the supply on two separate occasions, and finally locking the valve if the flow still persists, with an authorized Bluetooth override available for genuine emergency needs. Throughout, the implementation was also guided by a broader goal of making efficient, near-complete use of the ESP32 platform's onboard capabilities, keeping the codebase lean and purposeful rather than adding complexity without benefit.

## 2. Problem Considerations and Design Requirements

We did not design the prototype as a complete leak detector. It is meant to protect a flat-level water allowance when water continues to flow for a long time, for example when a tap is forgotten. The controller keeps a separate count for one continuous flow event. When the flow becomes zero, that event count is cleared. This means normal activities carried out at different times are not treated as one accidental-use event.

**Code Logic & Problem Solving**

- Maintain a daily volume total per flat/member, capped at the configured quota (e.g. 500 liters), reset automatically at 00:00.
- Maintain a separate continuous-flow counter, independent of the daily total.
- Reset the continuous-flow counter to zero the instant measured flow drops to zero.
- On continuous-flow counter reaching 50 L → trigger buzzer/indicator, close valve for 30 s, then reopen.
- If flow stays uninterrupted for a further 10 L → repeat warning + 30 s closure, then reopen.
- If flow stays uninterrupted for a further 3 L → lock valve permanently.
- Locked valve reopens only via authenticated Bluetooth command (also used for emergency ~200 L override).
- Keep flow-to-volume conversion ($f = 7.5Q$, ~450 pulses/liter) as an isolated, calibratable function separate from the state-machine logic.

| Requirement | Design Decision | Reason |
|---|---|---|
| Flat/member-wise allocation | Assign a fixed daily quota per flat or member, e.g. 500 liters | Ensures fair, individualized distribution rather than a shared undifferentiated supply |
| Daily allocation reset | Reset total consumption at 00:00 local time | Starts every calendar day with a fresh, easily understood quota |
| Quota recharge (optional) | Allow pay-to-recharge if allocation is exhausted | Configurable extension for genuine need beyond the daily quota; not implemented in current firmware |
| Normal use | Reset the continuous-flow counter when measured flow is zero | Prevents separated, everyday water uses from accumulating into a false continuous-flow event |
| First intervention | At 50 L of uninterrupted flow, activate buzzer/indicator and close the valve for 30 s | Creates a visible, audible pause that draws attention before stronger action |
| Second intervention | After reopening, a further 10 L of continuous flow triggers the same warning and another 30 s closure | Checks whether the demand remains unattended without treating the first warning as final |
| Final protection | After the second reopening, an additional 3 L of continuous flow causes locked closure | Limits further waste after two warning opportunities |
| Emergency access | An authorized Bluetooth command may reopen a locked valve, or grant an emergency allowance (~200 L) | Retains human control for genuine need while keeping automatic protection as the default |
| Sensor location | Place the flow sensor at the inlet, upstream of the valve | Measures water entering the controlled branch; reduces downstream drain-down being counted after closure |

*Table 1. Rules used in the system.*

# 3. Related Work and Positioning

Existing research broadly supports the use of flow measurement, warnings, controllable valves, and mobile access as effective tools for reducing water waste, and this project draws on that foundation while narrowing its focus to two specific gaps: fair, flat-wise allocation and detection of continuous, unattended flow. Singh and Ahmed reviewed the core components of IoT-based water-management systems, covering sensors, controllers, communication methods, and applications, and their survey helped frame the overall architecture adopted here [2]. Ho, Puika, and Kasih demonstrated a household water-reduction system built around flow sensors, alarms, a controllable valve, and Android-based access, establishing that a relatively simple hardware stack, sensor, actuator, and mobile interface, can meaningfully reduce domestic water use [3]. The present work follows this same general direction but keeps the local control rule deliberately simple and transparent, so that a non-technical user can understand exactly why a valve action was taken at any given moment.

This project also connects to a broader body of work on intelligent household water-management systems and edge-AI approaches to detecting water misuse. Mamede et al. proposed a prototype intelligent water-management system aimed at household-level monitoring and control, reflecting the same underlying motivation of moving beyond passive metering toward active management [4]. Loukatos et al. explored enriching IoT modules with edge-AI functionality to detect water-misuse events in a decentralized manner, pointing toward a future where flow patterns themselves are used to distinguish between different categories of use [5].

Where this project departs from the AI-driven approaches is in its current reliance on fixed, rule-based volume thresholds (50 L, +10 L, +3 L) rather than a learned model. This was a deliberate choice for a first build: every valve action can be traced back to a specific, explainable rule, which is valuable both for user trust and for debugging during prototyping. It also positions this work as a practical stepping stone rather than a competing alternative to edge-AI methods, since the flow data logged by this rule-based system could, in a later iteration, be used to train a model capable of distinguishing everyday use, an intentionally long activity, a likely leak, and a genuinely unattended tap, aligning directly with the direction proposed by Loukatos et al. [5]. In this sense, the present system is positioned as a foundation on which more adaptive, learning-based control could later be layered, rather than as a final endpoint.

# 4. System Architecture and Methodology

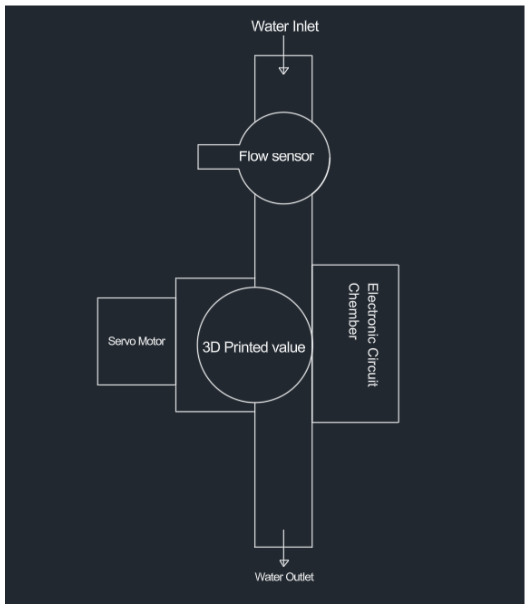


*Figure 1. Mechanical flow path supplied by the project team: inlet, flow sensor, servo-driven 3D-printed valve, electronic circuit enclosure, and outlet.*

## 4.1 Hardware Architecture

In our setup, water first passes through the flow sensor placed near the inlet. It then reaches the 3D-printed valve, which is moved by a servo motor. The ESP32 DevKit V1 counts the sensor pulses, calculates the volume, updates the LCD, gives a warning, and moves the valve. The ESP32 is suitable for this work because it provides GPIO, I2C, PWM, Wi-Fi, and Bluetooth in one board [6]. The LCD gives local feedback, while Bluetooth is kept for emergency opening of the valve.

| Subsystem | Prototype role | Key implementation note |
|---|---|---|
| YF-S201-type flow sensor | Produces pulse signals proportional to flow. | Installed near the inlet before the valve; use a 5 V-to-3.3 V-safe signal interface for ESP32 GPIO. |
| ESP32 DevKit V1 | Counts pulses, executes state machine, controls display and actuator. | Use NTP or a real-time clock for the 00:00 reset; Bluetooth is available on the ESP32 platform. |
| Servo and 3D-printed valve | Rotates the valve between open and closed states. | Supply servo from a suitable external 5 V source and connect all grounds together. |
| 16 x 2 LCD | Shows total volume and operational state. | I2C display should use 3.3 V-safe SDA/SCL levels. |
| Buzzer and visual indicator | Creates a local warning before automatic restriction. | Visual indication may be an LED or LCD status message; a suitable driver is required for higher-current devices. |

*Table 2. Main parts of the prototype and their roles.*

## 4.2 Flow Measurement

The controller counts the pulses coming from the flow sensor during each measurement interval. For a YF-S201-type sensor, the usual relation is $f = 7.5Q$, where f is frequency in hertz and Q is flow rate in L/min. The same data sheet gives a nominal value of 450 pulses per liter [7]. In the code, the flow rate is calculated as $Q = f/7.5$ and the volume for one second is $Q/60$. Since turbine sensors can change slightly with pipe fitting and flow conditions, the installed system should be checked using a known-volume container before it is used for any strict quota decision.

## 4.3 Continuous-Flow State Machine

The program keeps one counter only for the current no-break flow event. If the measured flow becomes zero, this counter returns to zero, but the total daily volume remains stored. As a result, small uses at different times can add to the daily total without starting the protection sequence. Table 3 shows the order of actions used when water keeps flowing.

| State | Entry condition | Action | Exit condition |
|---|---|---|---|
| Normal | No active continuous-flow event. | Count total daily volume and continuous volume only while water flows. | Continuous volume reaches 50 L, or flow becomes zero. |
| First warning hold | Continuous volume reaches 50 L. | Buzzer/indicator ON; close valve for 30 s. | After 30 s, reopen valve and monitor only persistent flow. |
| Second warning hold | A further 10 L passes without a zero-flow break. | Buzzer/indicator ON; close valve for 30 s. | After 30 s, reopen valve and begin final allowance. |
| Locked closure | A further 3 L passes without a zero-flow break. | Close valve and display locked condition. | Only an authorised Bluetooth emergency command reopens the valve. |
| Emergency override | Authenticated Bluetooth emergency request. | Open valve and log/indicate override status. | Manual end, timeout, or a defined safety rule in final firmware. |

*Table 3. Continuous-flow control sequence. The 50 L, +10 L, and +3 L values are project-set limits.*

### 4.4 Daily Quota and Emergency Mode

The total daily volume is reset at 00:00 local time, and the valve state returns to Normal. The time can come from NTP when Wi-Fi is available, or from a battery-backed real-time clock in a standalone version. For a real installation, the daily state should be stored in non-volatile memory so that a power cut does not remove a restriction. The daily quota and the continuous-flow rule serve different purposes: one tracks the day, while the other reacts to one ongoing event.

Emergency mode should not become an open remote switch. A Bluetooth command should come from a paired device and should use a passcode or another form of authentication. The LCD should clearly show that emergency opening is active, and the mobile application can save the time and reason for the action. This gives access to water in a real need without making it easy to bypass the protection rule.

## 5. Prototype Evidence and Functional Evaluation

The project photographs show the LCD working at different volume readings and show a valve-closing message. They are useful proof that the interface and basic controller states were demonstrated. However, they are not a calibrated laboratory data set. For this reason, we do not claim a measured sensor accuracy, a water-saving percentage, Bluetooth range, or a measured response time other than the programmed 30 s pause.


*Figure 2. Prototype LCD showing accumulated water volume of 6.68 L during testing (project photograph).*


*Figure 3. Prototype LCD at a 0.00 L display state with time information (project photograph).*

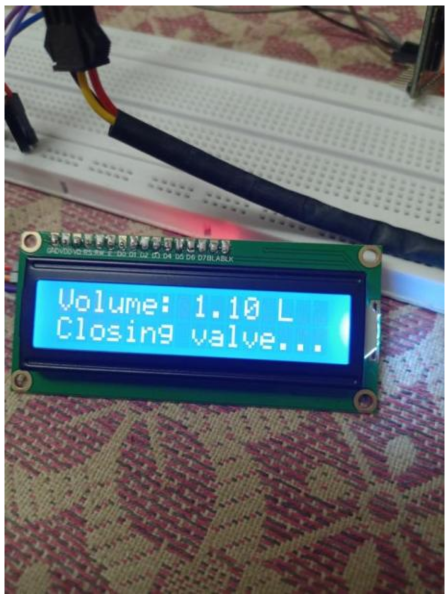


*Figure 4. Prototype LCD showing a valve-closing state at a configured test condition (project photograph).*

| Observed item | Evidence | Interpretation | Boundary of claim |
|---|---|---|---|
| Accumulated volume display | Figure 2 displays Total: 6.68 L. | The prototype presents a non-zero accumulated consumption reading. | No independent volume-calibration result is provided. |
| Zero-volume display and time | Figure 3 displays Volume: 0.00 L and time information. | The interface can present a reset/initial volume state with clock information. | The photograph alone does not prove a midnight reset event. |
| Valve-closing interface | Figure 4 displays Volume: 1.10 L and Closing valve... | The controller has a visible protective-closure state. | The displayed 1.10 L is treated as a bench-test configuration, not proof that the final 50 L threshold was tested. |
| Mechanical sequence | Figure 1 shows sensor upstream of the servo-driven valve. | The proposed plumbing order matches the inlet-placement consideration. | Hydraulic pressure-drop and long-term durability need separate tests. |

*Table 4. What the supplied prototype photographs show.*

## 6. Discussion

The step-by-step warning rule gives the user two chances to notice the warning before the final valve lock. The reset-on-zero condition is important here too: without it, normal use spread across the day could wrongly add up and look like one continuous event. With this rule, protection starts only when water keeps flowing without a break.

The flat-wise daily allocation, resetting at 00:00, adds a second layer of practicality on top of this, giving each household a transparent, predictable budget rather than treating the building's supply as one shared pool. The pay-to-recharge concept was considered as an extension of this same idea, but was left unimplemented for now, since a payment-linked mechanism raises security and transaction concerns beyond this prototype's scope.

Putting the sensor near the inlet is also an important practical choice. After a valve closes, water already present in the downstream pipe can still move toward the outlet, and a downstream sensor could count that movement as continued flow after closure. An inlet-mounted sensor avoids this and keeps a clearer link between the valve action and the reading. The final plumbing should still be tested for flow direction, filtration, pressure rating, and calibration in the actual pipe size.

A fixed limit cannot always tell a forgotten tap apart from a long activity that is genuinely needed. Bluetooth emergency access, whether to clear a lock or draw an emergency allowance, helps in that situation, but it must be tested carefully for security and ease of use. The final version should set a clear override time, authentication method, event record, and user notification before installation in a home.

## 7. Future Scope

- IoT dashboard: transmit total volume, continuous-flow events, valve state, and warning history to a cloud service for each flat or branch.
- Android integration: provide secure monitoring, quota configuration by authorised users, push notifications, and controlled emergency override.
- Advanced controller and storage: add a more robust microcontroller/edge gateway, non-volatile event storage, real-time clock backup, secure firmware updates, and outage recovery.
- Edge AI: train an on-device model from labelled flow traces to classify household use patterns and distinguish likely unattended flow from intentional high-volume use. Prior research has demonstrated the relevance of edge-AI approaches for water-misuse events [5].
- Expanded sensing: add pressure sensing, leak detection, water-quality indicators, and multi-flat aggregation for improved fault diagnosis and campus or apartment-scale analysis.

## 8. Conclusion

This project set out to address two connected problems in domestic water management: fair, flat-wise distribution of a shared water supply, and protection against water wasted through an unattended, continuously running tap. The resulting prototype, Each Drop Counts, brings together an ESP32 controller, a flow sensor, a servo-driven 3D-printed valve, an LCD, and a buzzer/visual warning interface into a system that goes beyond simple metering to actively manage consumption.

At the allocation level, the system assigns a fixed daily quota per flat or member, for example, a 500-liter allowance, resetting automatically at 00:00 local time, with a pay-to-recharge mechanism considered as a configurable extension for cases where a flat's quota is exhausted before the day resets. At the behavioral level, the system's central contribution is its staged response to continuous, uninterrupted flow: a warning and 30-second valve closure at 50 liters, a repeated warning and closure after a further 10 liters, and a permanent lock after an additional 3 liters, with an authorized Bluetooth command required to reopen the valve, whether to clear a lock or to grant an emergency allowance of roughly 200 liters for genuine need. Because the continuous-flow counter resets to zero the moment flow stops, ordinary, separated household activities are never mistaken for a single wasteful event, allowing the system to target unattended flow specifically rather than restricting normal use.

The prototype photographs confirm that the LCD correctly displays accumulated volume and that the valve-closing state functions as intended, offering a working demonstration of the interface and control logic on real hardware. That said, these photographs represent bench-level evidence rather than a calibrated laboratory result, and the current build does not yet substantiate a measured sensor accuracy, a quantified water-saving percentage, Bluetooth operating range, or a response time beyond the programmed 30-second pause. The pay-to-recharge feature likewise remains a planned rather than implemented capability at this stage.

Taken together, the system demonstrates that a low-cost, rule-based controller can meaningfully address both fairness and waste in a shared water supply, using logic simple enough for a non-technical user to understand while still resting on a foundation, sensor placement, state-machine design, and event logging, that can support more advanced features in the future. Before real deployment, the system still needs calibration against a known-volume reference, pressure and long-term endurance testing, a secure Bluetooth authentication implementation, non-volatile storage for daily and lock states, and field testing to evaluate false-warning rates and actual water saved.

### Acknowledgement

We thank Prof. P. S. Patil and the Department of Mechanical Engineering, Rajarshi Shahu College of Engineering, Pune, for their guidance and support during the Tinkering Lab project.

### References

[1] UNESCO, The United Nations World Water Development Report 2024: Water for Prosperity and Peace. Paris, France: UNESCO, 2024. Available: https://www.unesco.org/en/reports/wwdr/2024

[2] M. Singh and S. Ahmed, "IoT based smart water management systems: A systematic review," Materials Today: Proceedings, vol. 46, pp. 5211-5218, 2021, doi: 10.1016/j.matpr.2020.08.588.

[3] H.-C. Ho, K. S. Puika, and T. P. Kasih, "Development of IoT-based water reduction system for improving clean water conservation," Scientific Review Engineering and Environmental Sciences, vol. 29, no. 1, pp. 54-61, 2020, doi: 10.22630/PNIKS.2020.29.1.5.

[4] H. Mamede, J. C. Neves, J. Martins, R. Goncalves, and F. Branco, "A Prototype for an Intelligent Water Management System for Household Use," Sensors, vol. 23, no. 9, art. 4493, 2023, doi: 10.3390/s23094493.

[5] D. Loukatos, K.-A. Lygkoura, C. Maraveas, and K. G. Arvanitis, "Enriching IoT Modules with Edge AI Functionality to Detect Water Misuse Events in a Decentralized Manner," Sensors, vol. 22, no. 13, art. 4874, 2022, doi: 10.3390/s22134874.

[6] Espressif Systems, ESP32-WROOM-32 Datasheet, version 3.7, 2026. Available: https://documentation.espressif.com/esp32-wroom-32_datasheet_en.html

[7] YF-S201 Water Flow Sensor, product data sheet. Available: https://files.amperka.ru/store-media/products/water-flow-sensor/media/YF-S201.pdf