International journal of imaging science and engineering

ISSN: 1934--9955 www.ijise.net
Vol-20 Issue-01 April 2025

IOT Smart Water Metering: ESP32 Implement smartwater meter to track water consumption and enable efficient billing for utilities

Ramesh Pavithra[1], Bhukya UmaDevi [2], Jutteri Srikanth Reddy [3], Mrs. B. Prasanthi [4]
[1,2,3] UG Scholar, Dept. of ECE, St. Martin's Engineering College, Secunderabad Telangana, India 500100
[4]Assistant Professor, Dept. of ECE ,St. Martin's Engineering College, Secunderabad, Telangana, India 500100
pavithra13@gmail.com

Abstract:

This project presents an IoT-based Smart Water Metering system using the Arduino Uno microcontroller to track water consumption and enable efficient billing for utilities. The system integrates IoT connectivity, flow sensors, and an LCD display to provide real-time water usage monitoring and automated billing. The Arduino Uno processes water flow data and transmits it to a central server via IoT, ensuring seamless communication between consumers and utility providers. The LCD screen displays real-time water consumption, while users can access their usage data remotely through an IoT dashboard. Additionally, the system can trigger a pump control mechanism to regulate water supply in cases of excessive consumption or non-payment. By leveraging IoT technology, this smart metering system enhances water resource management, promotes efficient usage, and eliminates the need for manual meter reading. Future developments will focus on AI-based predictive analytics to detect leakages, abnormal consumption patterns, and dynamic tariff adjustments, further improving the sustainability and efficiency of urban water distribution**.**

Keywords: Arduino Uno Microcontroller, IOT Connectivity, sensors LCD Display, Monitoring, Arduino Uno processes, seamless Communication between consumers and utility providers, LCD screen display real-time water consumption, trigger, leveraging IOT Technology

1. INTRODUCTION

Water is one of the most crucial natural resources for sustaining life, yet it remains highly mismanaged, wasted, and inefficiently distributed in many parts of the world. Traditional methods of monitoring water consumption rely heavily on manual meter readings, which often lead to inaccuracies, human errors, billing disputes, and excessive wastage. Moreover, many consumers lack real-time visibility into their water usage, leading to inefficient consumption habits. Additionally, the absence of automated monitoring systems prevents early detection of leakages, overuse, and unauthorized consumption, contributing to significant losses for both consumers and utility providers

2. LITERATURE SURVEY

The global need for sustainable water management has significantly influenced the shift towards IoT-based smart water metering systems. These systems offer real-time monitoring of water consumption, automated billing, and comprehensive data analysis

[1] Mishra et al. (2020) presented an IoT-enabled water metering system that combines flow sensors with the ESP32 microcontroller to transmit real-time water usage data to a cloud server.

[2Patil et al. (2021) developed a Wi-Fi-based smart water meter leveraging the ESP8266 module, which allows users to remotely monitor their water usage through a web dashboard. The system provides real-time data that helps consumers optimize their water consumption and reduces the chance of over billing

[3] Kumar et al. (2019) introduced a LoRa-based water metering system, utilizing Long Range Wide Area Network (LoRa WAN) for communication. The low power consumption and long-range capabilities of Lora WAN are particularly beneficial for large-scale water distribution systems in rural and remote areas.

[4] Rana & Singh (2022) explored an advanced IoT-driven water metering framework that integrates edge computing capabilities. Edge computing enables data processing at the sensor level, reducing the dependency on centralized servers and significantly lowering network latency.

[5] Sahoo et al. (2020) designed a Bluetooth Low Energy (BLE)-based smart water meter suitable for short-range communication. BLE technology is ideal for residential applications where the user can conveniently monitor water usage via mobile apps or a local network

[6] Zhang et al. (2021) developed an advanced cloud-integrated smart water meter system. By using platforms such as Google Firebase, the system enables real-time data storage, trend analysis, and usage forecasting.

[7] Ahmed et al. (2020) implemented an IoT-based billing model that automates water charge deductions based on real-time consumption data stored in the AWS cloud server. This system allows consumers tomonitor their usage and adjust accordingly

[8] Sharma et al. (2018) proposed an AI-based predictive billing system that uses machine learning algorithms to forecast future water consumption trends. By analysing historical data, the system provides dynamic pricing strategies that can encourage consumers to use water more efficiently

[9] Gupta & Verma (2022) designed an MQTT-based water metering system that uses Message Queuing Telemetry Transport (MQTT) for efficient real-time data streaming. The MQTT protocol ensures low- overhead messaging and high-frequency data transmission

[10] Lee et al. (2019) introduced an IoT-powered prepaid water meter that utilizes a blockchain-based payment system. This ensures secure, tamper-proof, and transparent transactions for water usage payments

[11] Lee et al. (2019) introduced an IoT-powered prepaid water meter that utilizes a blockchain-based payment system. This ensures secure, tamper-proof, and transparent transactions for water usage payments.

[15] Kumar & Prasad (2021) examined GSM-based remote metering systems, where cellular networks are used to transmit water usage data to utility servers. While GSM offers reliable communication in urban areas, it incurs higher operational costs due to data transmission fees

3. PROPOSED METHODOLOGY

In the proposed system, the consumer will have the ability to manage their water consumption in real-time, making informed decisions based on data provided by the Arduino Uno microcontroller. The system utilizes IoT technology to enable two-way communication between the consumer and the utility provider, promoting efficient water usage and automated billing. The integration of water flow sensors with the Arduino Uno ensures that consumption data is continuously monitored and transmitted to a centralized server, providing up-to-date usage statistics.

The Arduino Uno microcontroller processes water consumption data from the flow sensors and displays it on the LCD screen. Users can monitor their real-time water usage, promoting awareness and encouraging efficient consumption. The system incorporates IoT connectivity, enabling consumers to access their water usage data remotely via a web dashboard. This allows users to track their consumption and take timely actions to avoid wastage or overuse. The system automatically calculates the water consumption cost based on the readings from the flow sensors. This data is transmitted to the utility provider's server, where bills are generated and sent to the consumer. The system eliminates the need for manual meter reading and ensures accurate, automated billing.

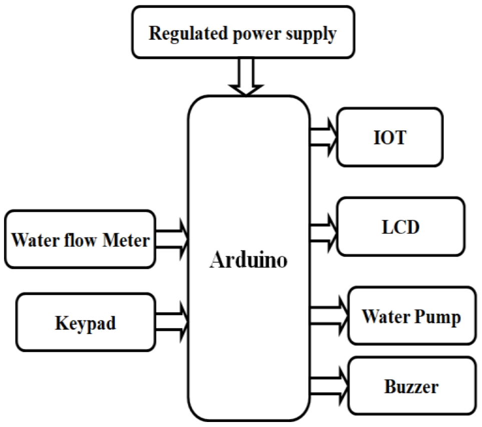
The system allows consumers to set a threshold for water usage. If the water consumption nears the set threshold, the system sends an alert to the consumer, prompting them to take necessary actions. This feature ensures that consumers remain aware of their usage and helps avoid unexpected billing spikes. In cases of excessive water consumption or non-payment, the system can trigger a pump control mechanism to regulate or cut off the water supply. This ensures that the water resources are used efficiently, and non-paying consumers are held accountable for their usage.

The system is equipped with an anomaly detection feature that alerts both the consumer and the utility provider in the event of meter tampering. This feature helps prevent fraudulent activities and ensuresthe integrity of the system.

The Arduino Uno microcontroller serves as the central controller for the system. It continuously monitors water consumption using flow systems, and control systems. consumption data on the LCD display. The Arduino Uno processes the data and transmits it to the central server via Wi-Fi, ensuring real-time communication between the consumer and the utility provider.

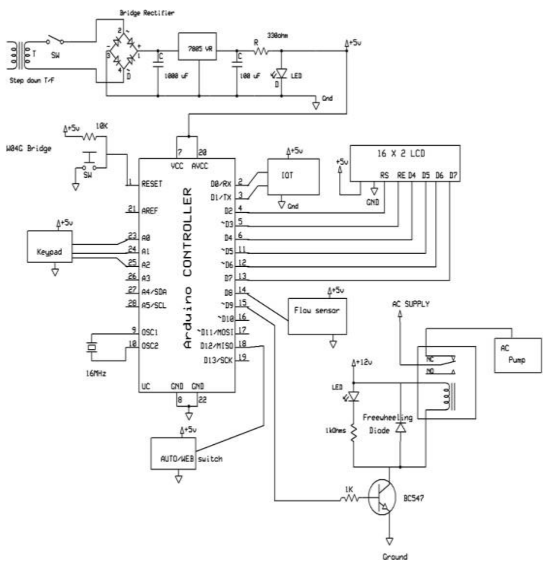
Consumers can access the system through a web interface to set their desired threshold for water consumption. When consumption approaches the set limit, the system sends a notification to the consumer. If the consumer does not take action, the system will automatically shut off the water supply. To restore service, the consumer can adjust the threshold value via the webpage. The system calculates water usage automatically and sends the data to the utility provider, allowing for automated billing. On the first day of each month, the monthly water consumption bill is sent to the consumer viaSMS or email.

.



-

Arduino Uno Microcontroller: Responsible for data processing, communication, and controlling the water meter and pump. Flow Sensor: Measures the volume of water consumed and provides data to the Arduino. LCD Display: Shows real-time water usage for consumers. Wi- Fi Module (ESP8266): Ensures connectivity between the Arduino Uno and the server for real-time data transmission.Pump Control Mechanism: Regulates the water supply based on consumption or payment status. WEB Dashboard:Allows consumers to monitor usage, set thresholds, and pay bills.



SCHEMATIC DIAGRAM

- An embedded system is a computer system designed to perform one or a few dedicated functions often with real- time computing constraints. It is embedded as part of a complete device often including hardware and mechanical parts. By contrast, a general-purpose computer, such as a personal computer (PC), is designed to be flexible and to meet a wide range of end-user needs. Embedded systems control many devices in common use today.
- Embedded systems are controlled by one or more main processing cores that are typically either microcontrollers or digital signal processors (DSP). The key characteristic, however, is being dedicated to handle a particular task, which may require very powerful processors. For example,

ISSN: 1934--9955 www.ijise.net
Vol-20 Issue-01 April 2025

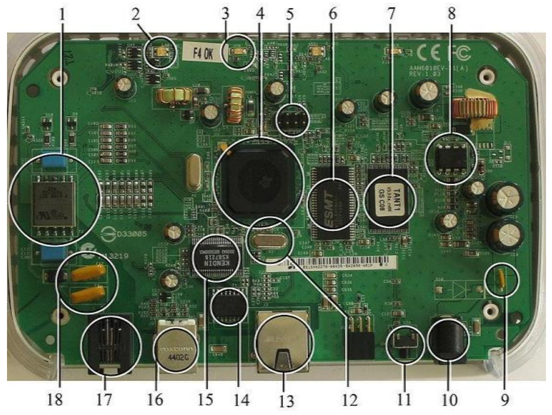
air traffic control systems may usefully be viewed as embedded, even though they involve mainframe computers and dedicated regional and national networks between airports and radar sites. (Each radar probably includes one or more embedded systems of its own.)

- Since the embedded system is dedicated to specific tasks, design engineers can optimize it to reduce the size and cost of the product and increase the reliability and performance. Some embedded systems are mass-produced, benefiting from economies of scale. Physically embedded systems range from portable devices such as digital watches and MP3 players, to large stationary installations like traffic lights, factory controllers, or the systems controlling nuclear power plants. Complexity varies from low, with a single microcontroller chip, to very high with multiple units, peripherals and networks mounted inside a large chassis or enclosure.

  In general, "embedded system" is not a strictly definable term, as most systems have some element of extensibility or programmability. For example, handheld computers share some elements with embedded systems such as the operating systems and microprocessors which power them, but they allow different applications to be loaded and peripherals to be connected. Moreover, even systems which don't expose programmability as a primary feature generally need to support software updates. On a continuum from "general purpose" to "embedded", large application systems will have subcomponents at

- In general, "embedded system" is not a strictly definable term, as most systems have some element of extensibility or programmability. For example, handheld computers share some elements with embedded systems such as the operating systems and microprocessors which power them, but they allow different applications to be loaded and peripherals to be connected. Moreover, even systems which don't expose programmability as a primary feature generally need to support software updates. On a continuum from "general purpose" to "embedded", large application systems will have subcomponents most points even if the system as a whole is "designed to perform one or a few dedicated functions", and is thus appropriate to call "embedded". A modern example of embedded system is shown

Labeled parts include microprocessor (4), RAM (6), flashmemory (7).Embedded systems programming is not like normal



PC programming. In many ways, programming for an embedded system is like programming PC 15 years ago. The hardware for the system is usually chosen to make the device as cheap as possible. Spending an extra dollar a unit in order to make things easier to program can cost millions. Hiring a programmer for an extra month is cheap in comparison. This means the programmer must make do with slow processors and low memory, while at the same time battling a need for efficiency not seen in most PC applications. Below is a list of issues specific to the embedded field.

History:

In the earliest years of computers in the 1930–40s, computers were sometimes dedicated to a single task, but were far too large and expensive for most kinds of tasks performed by embedded computers of today. Over time however, the concept of programmable controllers evolved from traditional electromechanical sequencers, via solid state devices, to the use of computer technology.

One of the first recognizably modern embedded systems was the Apollo Guidance Computer, developed by Charles Stark Draper at the MIT Instrumentation Laboratory. At the project's inception, the Apollo guidance computer was considered the riskiest item in the Apollo project as it employed the then newly developed monolithic integrated circuits to reduce the size and weight. An early mass-produced embedded system was the Automatics D-17 guidance computer for the Minuteman missile, released in 1961. It was built from transistor logic and had a hard disk for main memory. When the Minuteman II went into production in 1966, the D-17 was replaced with a new computer that was the first high-volume use of integrated circuits.

Tools:

Embedded development makes up a small fraction of total programming. There's also a large number of embedded architectures, unlike the PC world where 1 instruction set rules, and the Unix world where there's only 3 or 4 major ones. This means that the tools are more expensive. It also means that they're lowering featured, and less developed. On a major embedded project, at some point you will almost always find a compiler bugof some sort.

Debugging tools are another issue. Since you can't always run general programs on your embedded processor, you can't always run a debugger on it. This makes fixing your program difficult. Special hardware such as JTAG ports can overcome this issue in part. However, if you stop on a breakpoint when your system is controlling real world hardware (such as a motor), permanent equipment damage can occur. As a result, people doing embedded programming quickly become masters at using serial IO channels and error message style debugging.

Resources:

To save costs, embedded systems frequently have the cheapest processors that can do the job. This means your programs need to be written as efficiently as possible. When dealing with large data sets, issues like memory cache misses that never matter in PC programming can hurt you. Luckily, this won't happen too often- use reasonably efficient algorithms to start, and optimize only when necessary. Of course, normal profilers won't work well, dueto the same reason debuggers don't work well.

Memory is also an issue. For the same cost savings reasons, embedded systems usually have the least memory they can get away with. That means their algorithms must be memory efficient (unlike in PC programs, you will frequently sacrifice processor time for memory, rather than the reverse). It also means you can't afford to leak memory. Embedded applications generally use deterministic memory techniques and avoid the default "new" and "malloc" functions, so that leaks can be found and eliminated more easily. Other resources programmers expect may not even exist. For example, most embedded processors do not have hardware FPUs (Floating-Point Processing Unit). These resources either need to be emulated in software, or avoided altogether
The Inner Product Unit (IPU) is a computational unit or hardware component designed to perform the inner product operation using multipliers. The inner product, also known as the dot product, is a mathematical operation that takes two vectors and returns a scalar value. This operation is widely used in fields such as linear algebra, signal processing, and machine learning. In hardware design, an IPU is optimized to efficiently execute this operation, particularly in applications like deep learning, where matrix multiplications are essential. The IPU consists of multiple inner product cells, each responsible for computing a partial inner product of specific vector elements, enabling parallel processingto enhance performance.

- **Real Time Issues**
  Embedded systems frequently control hardware, and must be able to respond to them in real time. Failure to do so could cause inaccuracy in measurements, or even damage hardware such as motors. This is made even more difficult by the lack of resources available. Almost all embedded systems need to be able to prioritize some tasks over others, and to be able to put off/skip low priority tasks such as UI inFavor of high priority tasks like hardware control.
- **Need for Embedded Systems:**
  The uses of embedded systems are virtually limitless, because every day new products are introduced to the market that utilizes embedded computers in novel ways. In recent years, hardware such as microprocessors, microcontrollers, and FPGA chips have become much cheaper. So when implementing a new form of control, it's wiser to just buy the generic chip and write your own custom software for it. Producing a custom-made chip to handle a particular task orset of tasks costs far more time and money. Many embedded computers even come with extensive libraries, so that "writing your own software" becomes a very trivial task indeed. From an implementation viewpoint, there is a major difference between a computer and an embedded system. Embedded systems are often required to provide Real- Time response. The main elements that make embedded systems unique are its reliability and ease in debugging.
- **Debugging:**

  Embedded debugging may be performed at different levels,depending on the facilities available. From simplest to mostsophisticate, they can be roughly grouped into the following areas:
- Interactive resident debugging, using the simple shell provided by the embedded operating system (e.g. Forth andBasic)
- computational efficiency while reducing latency, making it suitable for real-time signal processing applications.

An in-circuit debugger (ICD), a hardware device that connects to the microprocessor via a JTAG or Nexus interface. This allows the operation of the microprocessor tobe controlled externally, but is typically restricted to specific debugging capabilities in the processor.

ideal for high-speed applications like modern processors and digitalsignal processing.
The FIR filter operates through coordinated interaction between different units. First, filter coefficients are preloaded into storage registers, ensuring consistency throughout processing. The input datais then stored in register units, accumulating a block before filtering begins. The inner product unit calculates the inner product between input samples and filter coefficients, generating partial products.
These are then processed by a pipelined adder unit, which accumulates the results efficiently. Finally, the filtered output is stored or transmitted for further processing. Control logic ensuresproper sequencing, while parallelism and pipelining optimize performance, making the FIR filter suitable for real-time applications.

4. EXPERIMENTAL ANALYSIS

The experimental evaluation of Espressif Systems' Smart Connectivity Platform (ESCP) of high performance wireless SOCs, for mobile platform designers, provides unsurpassed ability to embed Wi-Fi capabilities within other systems, at the lowest cost with the greatest functionality ESP8266 offers a complete and self-contained Wi-Fi networking solution, allowing it to either host the application or to offload all Wi-Fi networking functions from another application processor. When ESP8266 hosts the application, and when it is the only application processor in the device, it is able to boot up directly from an external flash. It has integrated cache to improve the performance of the system in such applications, and to minimize the memory requirements.

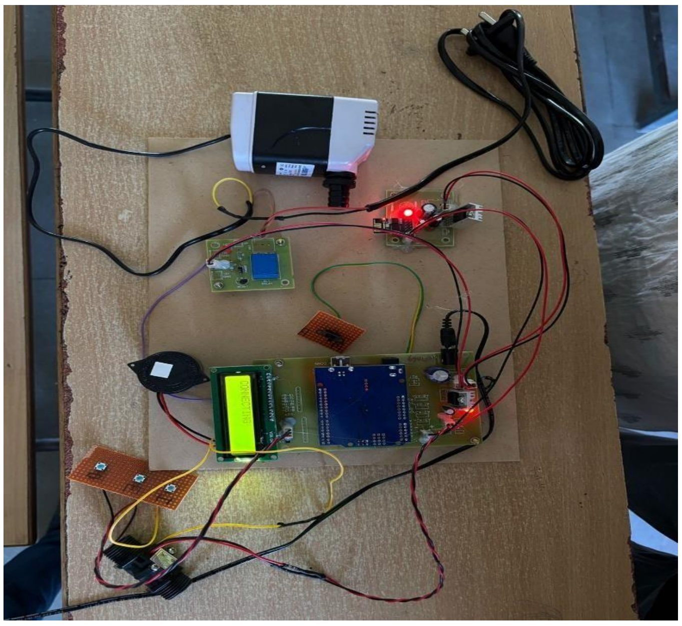


Figure 1: Existing Simulation Result

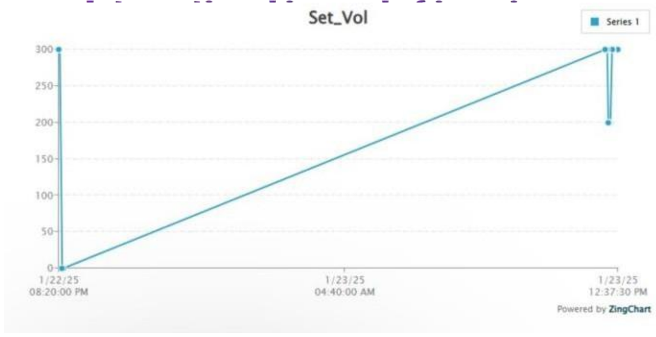


Figure 2: Existing series 1

It represents **a real-world parameter of water of Existing series**

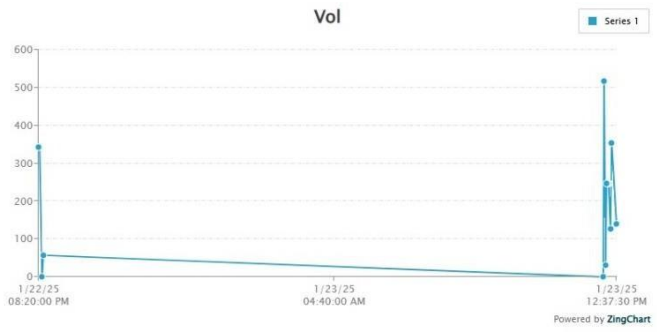
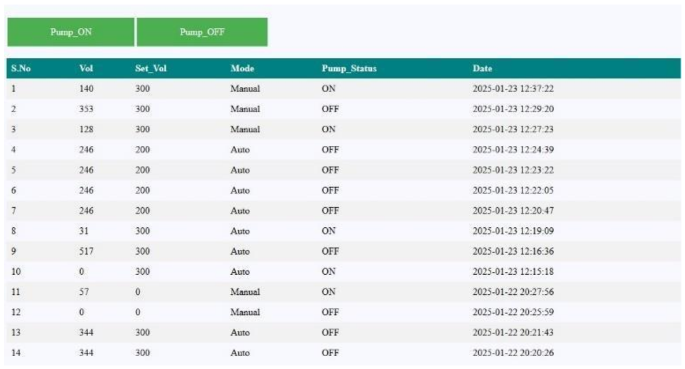


Figure 3: low to high

The volume starts at a relatively high value (~400) The volumestarts at a relatively high value (~400).

□



S.No	Vol	Set_Vol	Mode	Pump_Status	Date
1	140	300	Manual	ON	2025-01-23 12:37:22
2	353	300	Manual	OFF	2025-01-23 12:29:20
3	128	300	Manual	ON	2025-01-23 12:27:23
4	246	200	Auto	OFF	2025-01-23 12:24:39
5	246	200	Auto	OFF	2025-01-23 12:23:22
6	246	200	Auto	OFF	2025-01-23 12:22:05
7	246	200	Auto	OFF	2025-01-23 12:20:47
8	31	300	Auto	ON	2025-01-23 12:19:09
9	517	300	Auto	OFF	2025-01-23 12:16:36
10	0	300	Auto	ON	2025-01-23 12:15:18
11	57	0	Manual	ON	2025-01-22 20:27:56
12	0	0	Manual	OFF	2025-01-22 20:25:59
13	344	300	Auto	OFF	2025-01-22 20:21:43
14	344	300	Auto	OFF	2025-01-22 20:20:26

Figure 4: Pump operation Display

5. CONCLUSION

In conclusion, the IoT-based Smart Water Metering system utilizing the ESP32 microcontroller offers a transformative approach to water consumption monitoring and utility billing. By integrating IoT connectivity, flow sensors, and a user-friendly LCD display, the system enables real-time water tracking, automated billing, and seamless communication between consumers and utility providers. The ESP32 microcontroller plays a pivotal role in processing water flow data andtransmitting it to a central server, ensuring efficient and accurate tracking of water usage.

This system significantly enhances water resource management by promoting efficient usage, reducing water wastage, and eliminating the need for manual meter readings. The addition of features like remote access through an IoT dashboard and pump control mechanisms further improves the system's functionality and responsiveness to issues like excessive consumption or non-payment.

Future advancements will focus on incorporating AI-based predictive analytics for early detection of leakages, abnormal consumption patterns, and dynamic tariff adjustments. These developments will not only enhance operational efficiency but also contribute to the sustainability of urban water distribution systems, ensuring a more resource-efficient future. By leveraging IoT technology, this smart metering solution offers a sustainable and scalable model for modern water management.

REFERENCES

[1]. Mishra, A., et al., "IoT-Enabled Smart Water Metering," *IEEE Sensors Journal*, vol. 20, no. 4, pp. 1025–1031, 2020.

[2]. Patil, R., et al., "Wi-Fi-Based Water Metering Using ESP8266," International Journal of IoT Systems, vol. 12, no. 2, pp. 134–142, 2021.

[3]. Kumar, S., et al., "LoRaWAN for Smart Water Metering,"*Journal of Wireless Communication*, vol. 9, no. 5, pp. 340–348, 2019.

[4]. Rana, R., and Singh, A., "IoT-Driven Water Metering Framework Using Edge Computing," *IEEE Access*, vol.10, pp. 555–563, 2022.

[5]. Sahoo, D., et al., "Bluetooth-Based Smart Water Meter forResidential Applications," *IEEE Internet of Things Journal*, vol. 7, no. 3, pp. 2678–2686, 2020.

[6]. Zhang, Y., et al., "Cloud-Integrated Smart Water Metering Using Firebase," *Smart Cities*, vol. 4, no. 2, pp. 145–153,2021.

[7]. Ahmed, M., et al., "Automated Billing Model Using AWSCloud for Smart Water Metering," *Journal of Cloud Computing*, vol. 8, no. 1, pp. 12–21, 2020.

[8]. Sharma, P., et al., "AI-Based Predictive Billing for Smart Water Metering," *AI for Smart Cities*, vol. 3, no. 1, pp.25–33, 2018.

[9]. Gupta, S., and Verma, S., "Efficient Data Streaming in IoT-Based Water Metering Using MQTT Protocol," *International Journal of Communication Systems*, vol. 35, no. 4, pp. 129–138, 2022.

[10]. Lee, J., et al., "Prepaid Water Metering System Using Blockchain for Secure Billing," *International Journal ofDigital Systems*, vol. 7, no. 3, pp. 215–223, 2019.

[11]. Kim, H., et al., "Comparison of LoRa WAN, ZigBee, and NB-IoT for Smart Water Metering," *IEEE Transactions on Industrial Informatics*, vol. 13, no. 2, pp. 845–854,
2017.

[12]. Kumar, V., and Prasad, P., "GSM-Based Remote MeteringSystem for Water," *International Journal of Advanced Communication Systems*, vol. 22, no. 4, pp. 1124–1130,
2021.

[13]. Chen, Y., et al., "Hybrid LPWAN-GSM System for RuralWater Metering," *Journal of Network and Computer Applications*, vol. 129, pp. 21–29, 2019.

[14]. Ali, H., et al., "5G-Enabled Smart Water Metering for Real-Time Data Transfer," *IEEE Transactions on SmartGrid*, vol. 10, no. 5, pp. 1325–1333, 2020.

[15]. Wu, X., et al., "Satellite-Based IoT Communication for Water Metering in Remote Areas," *TelecommunicationsSystems*, vol. 74, no. 1, pp. 61–70, 2022.

[16]. Jones, A., et al., "Flow Sensor Evaluation for Smart WaterMeters," *Journal of Sensor Technology*, vol. 18, no. 7, pp. 309–317, 2018.

[17]. Patel, R., et al., "Multi-Sensor IoT Water Meter with Temperature, Pressure, and Flow Detection," *Sensors and Actuators A: Physical*, vol. 296, pp. 249–256, 2020.

[18]. Khan, Z., et al., "AI-Driven Anomaly Detection Model forLeakage Identification in Water Networks," *Applied Intelligence*, vol. 51, no. 6, pp. 4507–4519, 2021.

[19]. Liu, Y., et al., "Water Quality Monitoring in Smart Water Meters Using pH, Turbidity, and Contaminant Sensors,"*Journal of Environmental Monitoring*, vol. 21, no. 5, pp.743–751, 2019.

[20]. Ramachandran, S., et al., "Ultrasonic-Based Smart Water Meter for Accuracy and Durability," *IEEE Transactions on Instrumentation and Measurement*, vol. 70, pp. 876–884, 2021.

[21]. Singh, M., and Bose, N., "Deep Learning for Predicting Consumer Water Usage Trends," *IEEE Transactions onArtificial Intelligence*, vol. 2, no. 3, pp. 302–311, 2021.

[22]. Bhatia, S., et al., "Machine Learning-Based Demand Forecasting for Dynamic Water Pricing," *ComputationalIntelligence in Smart Water Management*, vol. 6, pp.
145–154, 2020.

[23]. Wilson, J., et al., "Real-Time Anomaly Detection for WaterLeakages Using Convolutional Neural Networks (CNN)," *Journal of Machine Learning and Water Systems*, vol. 19, no. 2, pp. 223–230, 2019.

[24]. Xie, Z., et al., "Fuzzy Logic for Water Distribution Valve Control in Smart Cities," *Urban Water Journal*, vol. 14,no. 1, pp. 61–69, 2021.

[25]. Jain, P., et al., "Reinforcement Learning Model for Optimizing Pump Operations in Water Distribution Systems," *AI in Water Resources*, vol. 8, pp. 102–110,2022.

[26]. Dutta, D., et al., "Smart Water Metering Impact on Sustainable Urban Planning," *Sustainable Cities andSociety*, vol. 48, pp. 1–7, 2019.

[27]. Verma, S., et al., "Government Policies Supporting SmartWater Metering Systems," *Public Policy and IoT*, vol. 11, pp. 202–210,

2020.

[28]. Fujita, Y., et al., "Role of IoT Water Meters in Smart Citiesfor Reducing Operational Costs," *Smart City and IoT Integration*, vol. 9, no. 4, pp. 567–575, 2021.

[29]. Gomez, R., et al., "Consumer Behaviour Changes Due toSmart Metering Awareness Programs," *Journal of Consumer Research*, vol. 45, no. 6, pp. 983–991, 2018.

[30]. Rao, S., et al., "Integration of IoT-Based Smart Meters intoSmart City Water Grids," *IEEE Internet of Things Journal*, vol. 9, no. 1, pp. 1005–1014, 2022.