water

MDPI

Review

IoT-Based Solutions to Monitor Water Level, Leakage, and Motor Control for Smart Water Tanks

Farmanullah Jan [1,*], Nasro Min-Allah [1], Saqib Saeed [2], Sardar Zafar Iqbal [2] and Rashad Ahmed [3]

[1] Department of Computer Science, College of Computer Science and Information Technology, Imam Abdulrahman Bin Faisal University, P.O. Box 1982, Dammam 31441, Saudi Arabia; nabdullatief@iau.edu.sa
[2] Department of Computer Information Systems, College of Computer Science and Information Technology, Imam Abdulrahman Bin Faisal University, P.O. Box 1982, Dammam 31441, Saudi Arabia; sbsaed@iau.edu.sa (S.S.); saiqbal@iau.edu.sa (S.Z.I.)
[3] ICS Department, King Fahd University of Petroleum and Minerals, Dhahran 31261, Saudi Arabia; othmanr@kfupm.edu.sa
* Correspondence: fzmjan@iau.edu.sa

Abstract: Today, a large portion of the human population around the globe has no access to freshwater for drinking, cooking, and other domestic applications. Water resources in numerous countries are becoming scarce due to over urbanization, rapid industrial growth, and current global warming. Water is often stored in the aboveground or underground tanks. In developing countries, these tanks are maintained manually, and in some cases, water is wasted due to human negligence. In addition, water could also leak out from tanks and supply pipes due to the decayed infrastructure. To address these issues, researchers worldwide turned to the Internet-of-Things (IoT) technology to efficiently monitor water levels, detect leakage, and auto refill tanks whenever needed. Notably, this technology can also supply real-time feedback to end-users and other experts through a webpage or a smartphone. Literature reveals a plethora of review articles on smart water monitoring, including water quality, supply pipes leakage, and water waste recycling. However, none of the reviews focus on the IoT-based solution to monitor water level, detect water leakage, and auto control water pumps, especially at the induvial level that form a vast proportion of water consumers worldwide. To fill this gap in the literature, this study presents a review of IoT-controlled water storage tanks (IoT-WST). Some important contributions of our work include surveying contemporary work on IoT-WST, elaborating current techniques and technologies in IoT-WST, targeting proper hardware, and selecting a secure IoT cloud server.

Keywords: internet-of-things; leakage detection; level monitoring; sensors; smart water tank; smart city

check for updates

Citation: Jan, F.; Min-Allah, N.; Saeed, S.; Iqbal, S.Z.; Ahmed, R. IoT-Based Solutions to Monitor Water Level, Leakage, and Motor Control for Smart Water Tanks. *Water* **2022**, *14*, 309. https://doi.org/10.3390/w14030309

Academic Editor: Bruno Brunone

Received: 13 November 2021
Accepted: 3 January 2022
Published: 20 January 2022

1. Introduction

Water is necessary for life on our planet. Seventy-one percent of the earth's surface is covered with water. No doubt, this quantity is much smaller compared with the total earth volume [1–14]. The oceans contain around 97% of the total water on earth [15–28]. Unfortunately, ocean water also has very heavy salt content and thus cannot be used directly for many household needs such as drinking, cooking, etc. [29,30]. The rest of the total water is available as freshwater [31].

Fresh water, which has relatively low salt contents and other dissolved solids [32], is the main component in many human activities, including agriculture, industries, and domestic applications. Unfortunately, this small quantity is not entirely available to us because around 69% of freshwater is trapped in the glaciers and polar icecaps. Around 30% of fresh water exists as groundwater, i.e., water under the earth's surface. Collectively, around 1% of total freshwater is available for all forms of human consumption [30,32].

Research shows that worldwide, freshwater resources are rapidly diminishing [4,12,30,32–35], causing tension among different nations. The primary reasons for this crisis are (i) uneven distribution of the water resources around the globe, (ii) rapid growth in urbanization, (iii) drastic increase in industrial activities, and (iv) a lack of awareness about water consumption. In addition, many countries around the world are facing droughts and flood-like conditions due to the global warming phenomenon. To overcome the freshwater shortage, some water-scarce countries (e.g., Saudi Arabia) are already receiving fresh water from the seawater through the desalination process, which requires energy, workforce, and funding.

Food and freshwater are linked, and both are necessary for survival. Agriculture activities consume a large portion of fresh water to produce crops, fruits, plants, etc. As stated in [36], the demand for food is expected to rise by more than 50% over the coming 30 years due to the continuous growth of the world's population.

To mitigate the effect of the water crisis, many initiatives are taking place worldwide to take care of our freshwater resources [37–49]. For instance, the water monitoring day was established in 2003 by the America Clean Water Foundation [50,51], aiming to create public awareness of water contamination. In addition, the World Health Organization (WHO) and the United States Environmental Protection Agency (USEPA) have constantly been updating their recommendations and guidelines on water-borne diseases and pollution [52–54]. The "World Water Council" report indicates that the global population may rise by 40 to 50% over the next 50 years [55–57]. This super growth, along with urbanization and industrialization, might increase the overall demand for water resulting in a strong impact on the world ecosystem.

Water pollution and the use of unsafe water result in death and disease and are, therefore, significant health concerns [54,55,58]. Fortunately, numerous research efforts are going on to improve water quality monitoring [59,60]. Due to this fact, the scope of our study has intentionally been confined to the IoT application in smart water tank- monitoring.

As already noted, global freshwater supplies are continuously diminishing. In the rural regions, humans rely on water obtained from tube wells, wells, ponds, and lakes. In such localities, water is seldomly stored in proper water storage tanks. Instead, it is simply collected as needed from the respective sources. Undoubtedly, water is wasted through excessive use in personal hygiene, washing clothes, dishes, vehicles, watering plants, and other ways. Providing relevant information and educating these and other communities on issues around the quality and availability of freshwater resources may prevent or at least reduce water wasting.

On the other hand, in urban areas, water is supplied by the local water bodies, e.g., municipal corporations. Such water bodies are generally entrusted by the local or federal government to provide water for domestic applications. In general, this demand is fulfilled using water from tube-wells, lakes, ponds, or rivers purified through the Reverse Osmosis (RO) plants (if the water quality is not sufficient for direct domestic usage) [59,60].

Water is usually stored in the overhead or underground tanks. In addition, larger organizations (e.g., universities, factories, and malls) also fill tanks from their private tube wells. The use of tube wells and over-pumping water from the underground resources lead to decreasing levels of fresh water, water pollution, and increased overall expenses associated with deeper drilling and pumping out of water. Proper water storage is a challenge in hilly terrains due to the road infrastructure. There may also exist no underground reserves, or it could be extremely deep in some desert-type regions, e.g., Saudi Arabia. In such cases, water is generally transported in the water tankers, which may incur considerable costs to the end-users and waste water during transportation.

Given the overall challenges in the fresh water supply crisis, preventing water wastage seems not only prudent but necessary. In this respect, researchers are continuously working on designing and developing smart systems to detect water leakage in the supply lines, control water theft, and monitor water storage tanks [31,51,61,62]. In these global efforts, the topic of IoT-WST has gained more attention [31,34,59,60,63–65], and the following parameters are considered while designing a typical IoT-WST:

- Automatic monitoring of water levels in tanks,
- Automatic detection of water leakage from tanks and supply pipes in the proximity,
- Automated tanks refilling while avoiding the dry run of motors or pumps, and
- Providing access to the end-users to control and observe relevant activities remotely.

Li et al. [66] published a review paper on redefining the framework for smart systems water monitoring. Its main aim is to define what a smart water system is and develop a systematic framework. While supplying good insight into different layers of the smart system, this article is not focused on smart monitoring of water storage tanks (IoT-WST). Yuvaraj et al. [67] published a review paper on water leakage detection and monitoring. However, its scope is limited because it offers little information on water leakage monitoring and lacks comparative analysis. In addition, the review considered only articles discussing specifically water leakage detection in the main supply pipes.

Ali and Choi [68] published a review article on the wireless sensors network (WSN) based water monitoring. This article concentrated on contemporary methods developed to check water leakage in the underground pipelines, sinkholes produced by leakage, and the feasibility of WSN to check sinkholes and leakage. Though it supplies an in-depth analysis of these issues, the review offers nothing on IoT-WST.

In [69], the authors published a review on water monitoring systems utilizing IoT technologies. Though the main interest of the discussion was the possibility of reducing water wastage using IoT technology, the review only touched on the issues of water quality, level, and leakage control at a very abstract level. Sheltami et al. [70] published a survey focused on WSN-based monitoring for leakage in water supply pipelines. This article is well documented and provides analyses of different water leakage detection schemes, but its scope is also confined to only WSN-based schemes. It offers no insight into IoT-WST.

The Public Utilities Board Singapore [71] published a review article focused on managing the water distribution network through a smart water grid. It is undoubtedly well documented, and its scope is relatively broad. For example, it covers various aspects such as assets management, water quality, leakage detection, water conservation, and automated meter reading. However, while offering much in the way of critical analysis of these parameters, it covers nothing on IoT-WST. Finally, in [72], the authors composed a comprehensive review on IoT-based water quality monitoring in underground and overhead tanks.

Literature survey reveals (Table 1) that there exists a plethora of review articles on smart water monitoring, including water quality monitoring, leakage monitoring in supply pipes, and recycling water waste, among other issues. However, it also shows that none of the recent reviews has focused on the IoT-based solution to monitor water level, detect water leakage, and auto-control water pumps, especially at the level of the induvial or private users who comprise a large portion of water consumers worldwide. To fill this research gap, this study presents a review of the IoT-controlled water storage tanks (IoT-WST). Some motivational factors of our study are as follows:

- A comprehensive survey of related work,
- Reviewing recent technologies and techniques,
- Exploring existing software and hardware platforms for IoT-WST, and
- Highlighting the cyber security threats.

The remaining of this article is structured as follows: Section 2 explains the background of the water storage tanks monitoring. Section 3 explains the research method and offers a critical survey of contemporary work on IoT-WST. Section 4 elaborates on some technologies and techniques commonly used while designing and developing IoT-WST. In Section 5, the authors offer specific details about the potential challenges, trends, and limitations of IoT-WST. Finally, Section 6 concludes this study. Table 2 has the list of acronyms utilized in this article.

Table 1. Contemporary reviews on smart water monitoring.

Reference	Major Features	Comments
[66]	Redefining framework for smart systems to monitor water	Not focused on smart monitoring of water storage tanks
[67]	Water leakage detection in the main supply pipes	Not focused on smart monitoring of water storage tanks
[68]	Feasibility of WSN to monitor sinkholes and leakage	Not focused on smart monitoring of water storage tanks
[69]	Reducing water wastage using IoT technology; In addition, less coverage is also given to water quality, level, and leakage control	Not focused on smart monitoring of water storage tanks
[70]	WSN based monitoring for leakage in water supply pipelines	Not focused on smart monitoring of water storage tanks
[71]	Management of the water distribution network through smart water grid	Not focused on smart monitoring of water storage tanks
[72]	IoT-based monitoring of water quality in the underground and overhead tanks.	Not focused on smart monitoring of water storage tanks

Table 2. Commonly used acronyms.

Acronym	Definition	Acronym	Definition
IoT	Internet-of-Things	WSN	Wireless Sensors Network
IoT-WST	IoT Controlled Water Storage Tank	ADC	Analogue-to-digital Convertor
WHO	World Health Organization	MCU	Microcontroller Unit
USEPA	The United States Environmental Protection Agency	I/O	Input/Output
RO	Reverse Osmosis	DAC	Digital-to-Analog Converter
UIDs	Unique Identifiers	LCM	Liquid Crystal Module
WAN	Wide Area Network	SNU	Sensors Node Unit
GU	Gateway Unit	CSU	Cloud Server Unit
UIU	User Interface Unit	GSM	Global System of Mobile Communication
GPRS	General Radio Packet Service	CPU	Central Processing Unit
SMS	Short Message Service	USB	Universal Serial Bus
IDE	Integrated Development Environment	SoC	System-on-Chip
UML	Unified Modeling Language	DC	Direct Current
MIT	Massachusetts Institute of Technology	SQL	Structured Query Language
LED	Light Emitting Diode	HTML	Hypertext Markup Language
I2C	Inter-Integrated Circuit	UART	Universal Asynchronous Receiver Transmitter
SPI	Serial Peripheral Interface	PCB	Printed Circuit Board
SRAM	Static Random-Access Memory	LDR	Light Dependent Resistor
ML	Machine Learning	WS	Water Sensor
CSMS	Capacitive Soil Moisture Sensor	LIDAR	Light Detection and Ranging
AI	Artificial Intelligence	PIN	Personal Identification Number

2. Background

This section briefly details some fundamental basics, which may offer convenience to readers of this article.

2.1. Traditional Monitoring

Most often, the water tank monitoring is performed manually [59,60,63,73]. For example, a consumer can refill a tank when it is empty and fix water leakage if any is detected. Though this method has been in use for a long time, it has some serious limitations. For instance, 24/7 h monitoring of tanks in person may not be feasible for individuals at private locations such as homes, schools, universities, organizations, mills, factories, etc. In fact, it is often the case that water overflows from tanks undetected. Moreover, the task of manually checking the water level in tanks if needed is tedious and often not impossible.

2.2. Off-line Automated Monitoring

As noted above, manual monitoring of water storage tanks may not be a comfortable experience, especially in the water-scarce regions such as Saudi Arabia [31,72,74]. Thus, researchers have devised off-line embedded systems to monitor water storage tanks [31,32,59,60]. In such approaches, researchers deploy a microprocessor-based system for monitoring tanks. A typical off-line tank monitoring system may include (1) sensors, (2) actuators, (3) processor(s), and (4) supportive electronic components. These units are briefed below:

- Sensor: It can detect modifications in its surroundings and transfer collected data to relevant electronic modules (e.g., a microprocessor). Notably, a sensor is always supplemented with other electronic modules (e.g., analog-to-digital converter (ADC)) for proper signal conditioning [75].
- Actuator: it is a device (e.g., transistor, electromechanical relay, and thyristor), which is capable of causing machines or devices to run.
- Processor: In embedded systems, dedicated microprocessors (also called microcontrollers) are utilized. In general, a microcontroller unit (MCU) is called a true computer on a single chip, which has all necessary peripherals (e.g., memory, timers/counters, digital and analog input/output (I/O) ports, ADC, and digital-to-analog converter (DAC)) on-chip. This unit can easily read sensors data, process, store, and update output devices (e.g., liquid crystal module (LCM)) if required, and transfer data to other devices and machines if needed.
- Supportive electronic components: (e.g., power supply unit, buffers, resistors, and diodes) are always required to power up the target system, integrate sensors with I/O ports, etc.

To summarize, off-line automated monitoring systems are suitable for monitoring water storage tanks, but locally. They may not have the requisite electronic interface or modules to transfer sensor data to remote devices through wired or wireless communication channels. For this reason, their scope is limited to individual usage only.

2.3. WSN-Based Monitoring

To extend the capabilities of the off-line monitoring systems, researchers approached towards usage of the WSN technology [68,76,77]. In a typical WSN (Figure 1), a sensor node (an MCU-based kit) first reads in the sensors data (e.g., leakage) being installed on-site. After reconditioning and processing, the data is sent to the main station (also called a server) wirelessly using different wireless channels (e.g., Lora WAN, Xbee, Wi-Fi, Bluetooth, nRF24L01, or RF 433) [78]. On the reception site, the main station performs further analysis to find out hidden patterns and anomalies, if any. Finally, the processed data is stored, results are revised, and, if needed, feedback is initiated to the relevant authorities or end-users via an email, SMS, etc. In this scenario, the main station may also have full control over sensors and actuators being connected to each sensor node.

While resolving some of the limitations of off-line monitoring systems, this technology also has some limitations, such as low spatial resolution due to private network infrastructure, compromised security, energy requirements, storage issues, and high maintenance and installation costs.

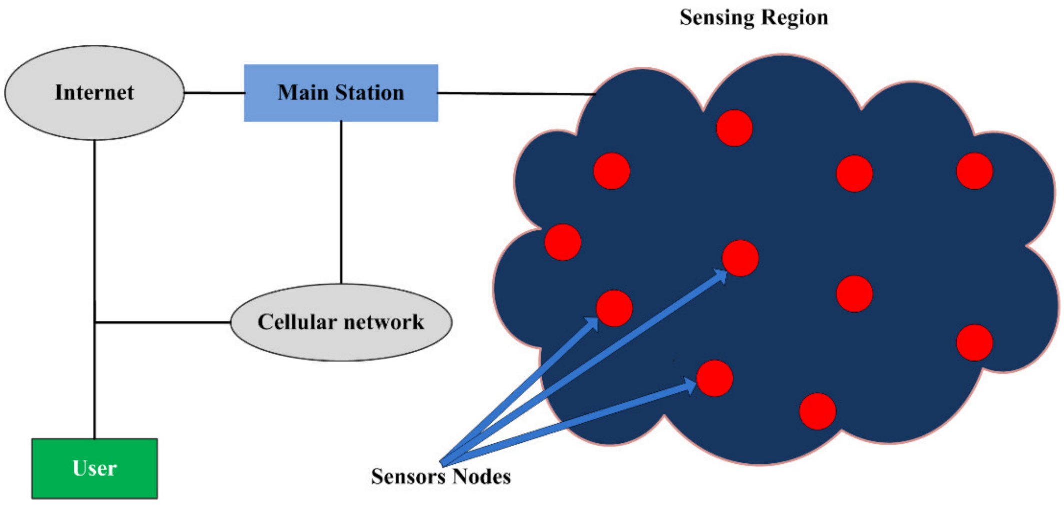

Figure 1. A typical block diagram of WSN.

2.4. Smart Monitoring

Due to the spatial limitations of WSN technology, sensor nodes are monitored through a local server. Thus, global access to individual nodes in such networks is impossible. To address this issue, researchers resorted to using IoT technology, i.e., smart monitoring. In such technology, each node can directly send data to an IoT cloud server [79], or nodes may also forward sensors data to a master node, subsequently transmitting it to the IoT cloud server for further processing, analysis, etc. Moreover, the concerned authority or end-user may also have direct access to each node and may thus control its functionality whenever needed.

IoT technology involves portable sensors, computing devices, and the internet and communication infrastructure to control things (e.g., motor, patient, and robot) from any corner of the world, ideally with no spatial limitations. Smart systems have potential applications in smart cities, municipal waste recycling, aquaculture, agriculture, health and care, education, flood monitoring, and other areas [51,80,81]. WSN is a subset of IoT [82–84]. Figure 2 illustrates the block diagram of a typical IoT–WST. In general, it comprises four basic units: (i) Sensors node unit (SNU), (ii) Gateway unit (GU), (iii) Cloud server unit (CSU), and (iv) User interface unit (UIU).

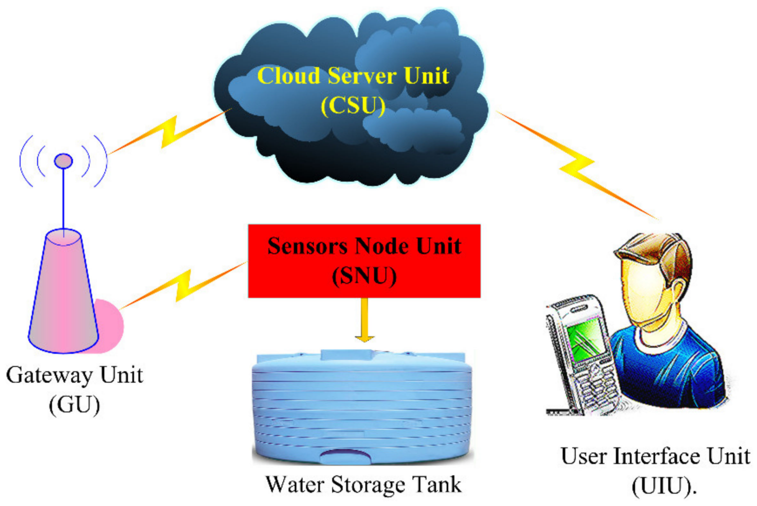

Figure 2. A typical diagram of IoT-WST.

Briefly, the SNU captures sensors data (e.g., water level and leakage), reconditions and processes it, local displays are updated accordingly and then sent to CSU through a Hotspot (local Wi-Fi), ethernet channel, or GSM/GPRS modem. The GU is responsible for the communication between SNU and CSU. Almost all internet service providers currently facilitate their customers through the hired cloud servers and other facilities whenever needed [79]. Today, third parties offer cloud servers for many IoT applications [79], e.g., Ubidots, Blynk, and Adafruit. IoT developers can more comfortably develop an extensive range of IoT products using commercial cloud servers for many useful applications, e.g., health, automobiles, water monitoring, and aquaculture. Some major highlights of this technology are as follows:

1. Reduced Cost: As it uses the existing communication infrastructure of the internet, the overall cost for the system's development has been reduced, e.g., no personal communication network is generally required.
2. Higher Spatial Resolution: As its backbone is based on the internet, its spatial resolution is ideally infinite. It implies that monitoring water storage tank is possible from any corner of the globe wherever access to the internet is possible.
3. Reduced Computational Cost: In general, a sensor node should be equipped with an ordinary MCU/CPU (Central processing unit) based kit (e.g., NodeMCU [85], ESP8266 Transceiver [85] or Arduino Nano 33-IoT [86]) and any heavy computational load should be shed to IoT cloud servers, e.g., IBM, Adafruit, Blynk, Arduino, and Ubidots IoT platform [79]. Thus, the use of hi-tech computing devices such as the DE1 SoC FPGA board [75] and the Raspberry Pi 4 Model-B [87] could be avoided.
4. Lowered Energy Requirement: While shedding complex computations (heavy load) to cloud servers, sensors nodes could be in the relaxed mode, i.e., doing less work and staying mostly in the idle/sleeping modes. Therefore, a small battery could also be used to energize sensors nodes in the energy crisis sites.
5. Real-time Feedback: Embedded systems centered around IoT technology can supply real-time feedback to its end-users via a short message service (SMS), tweeter, email, and Facebook.

3. Methodology

As shown in Figure 3, in conducting this survey, the authors followed the PRISMA guidelines [88].

We focused on four research repositories during the identification stage, namely Scopus, Web of Science, Google Scholar, and IEEE Xplore. We found a total of 284 documents. Our qualification criteria included the potential search strings "Smart water tank monitoring", "Smart water level and leakage monitoring", "Smart water pump control", "Internet-of-Things based water tanks", IoT for water tanks", and "IoT-based water tank monitoring". Furthermore, we only focused on the papers published after 1 January 2016.

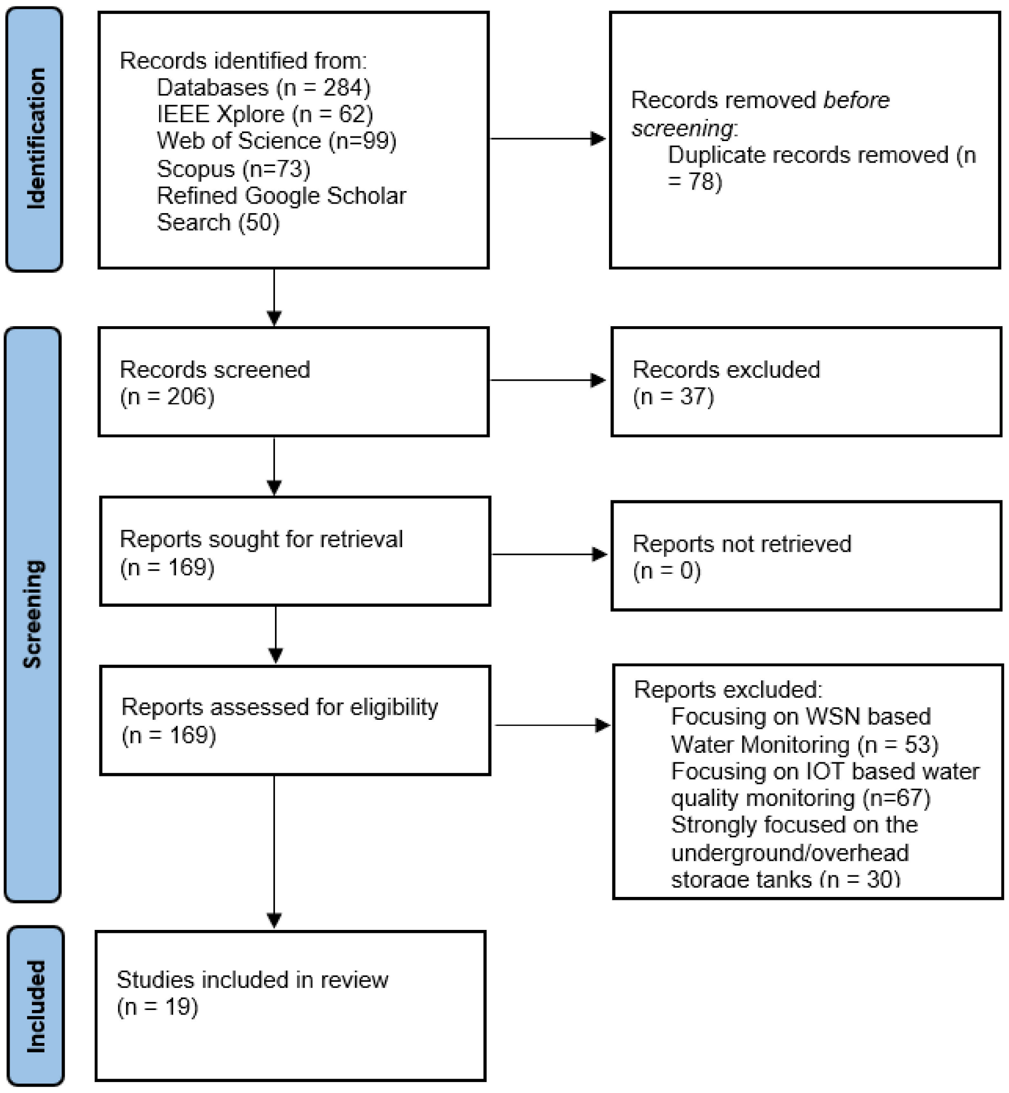
Identification
Records identified from:
Databases (n = 284)
IEEE Xplore (n = 62)
Web of Science (n=99)
Scopus (n=73)
Refined Google Scholar Search (50)
Records removed *before screening*:
Duplicate records removed (n = 78)
Screening
Records screened (n = 206)
Records excluded (n = 37)
Reports sought for retrieval (n = 169)
Reports not retrieved (n = 0)
Reports assessed for eligibility (n = 169)
Reports excluded:
Focusing on WSN based Water Monitoring (n = 53)
Focusing on IOT based water quality monitoring (n=67)
Strongly focused on the underground/overhead storage tanks (n = 30)
Included
Studies included in review (n = 19)

Figure 3. Workflow of the systematic literature review [89].

Out of these 284 documents, 123 were found on the Scopus and Google Scholar, 99 on the Web of Science, and 62 were from the IEEE Xplore. There were 39 documents common across these databases, which reduced our set to 206 documents. During the screening stage, the quality of the downloaded documents was decided based on the quality assessment criteria proposed by [90,91]. This way, we excluded further 37 documents and downloaded the remaining set of 169 articles through Google Scholar or from the individual journals' websites.

Among the downloaded articles, 53 and 67 were focused on WSN-based water monitoring and IoT-based water quality monitoring, respectively, so we excluded them from our study. The remaining 49 articles concerned specifically monitoring and controlling water leakage, pump, solenoid valve, and water levels. The selection included one book chapter, 11 conference papers, and 37 journal articles and was further filtered to create a pool of papers strongly focused on the underground and overhead storage tanks. As a result, the authors were left with 19 articles, including a book chapter, conference papers, and journal papers.

Smart Storage Tanks: Results and Discussion

Kumar et al. [92] published an article focused on developing a microgrid system to control water tanks at the town level. It aimed to reduce water wastage due to tanks overflowing and leakage and the workforce required to monitor water tanks manually. Herein, the tank-mounted unit powered by a solar panel reads water level in the tank via the Arduino Uno kit. This gadget is made of an 8-bit MCU (Atmega328P) surrounded by several peripherals such as the USB interface, ADC, input/output (I/O) pins, timers/counters, SPI (serial peripheral interface) module, etc.

To detect water level, the authors used an ultrasonic sensor (HC-SR04), which is a complete signal conditioning module. Next, they used a GSM/GPRS SIM900A shield plugged in the Arduino kit for transferring data to the ThingSpeak [79], which is a private IoT cloud server. It can store, analyze, visualize, and find hidden patterns in the acquired sensors data. While some limited resources are offered freely, IoT developers need to pay for the commercial activities if needed.

In addition, its water source unit automatically refills and controls the water level in the tank. The GSM/GPRS SIM900A plugged in Arduino Uno receives commands from end-user via smartphone having installed the IoT App. According to received commands, Arduino Uno turns the water pump on or off through an electromechanical relay. The authors used the Arduino IDE (integrated development environment) to test, simulate, and validate this system. In addition, they also used the ATOM and Ionic framework while developing a Mobile App. The authors developed an algorithm based on Bernoulli's principle to detect water leakage and its corresponding area. While leakage detection mode is active, a user could not consume water from the tank for six consecutive hours. During this mode, the concerned unit takes water level readings every 30 s and sends data to the ThingSpeak for further analysis and investigation.

Though this system is documented well, there is clearly room for further improvement, especially in hardware. For example, the authors could use a single relay unit instead of a readymade module having four such units. In addition, if both tank-mounted unit and water-source unit are in close range of each other, then one GSM/GPRS SIM900A gadget could be replaced with another suitable gadget such as the XBee device, Lora WAN, Bluetooth, nRF24L01, or RF433. This would undoubtedly reduce the system's overall cost because SIM900A relies on an active SIM card, and it uses the telephone company services for data communication. Table 3 offers a brief comparison of contemporary work on IoT-WST.

Table 3. Comparison of contemporary IoT-WST.

Ref.	**Sensors**	**Actuators**	**Processing Units**	**IDE**	**Comments**
[92]	HC-SR04	Electromechanical relay module; Solenoid valve; Water pump	Arduino Uno (8-bit MCU); GSM/GPRS SIM900A	Arduino IDE, ATOM, and Ionic framework; ThingSpeak IoT platform	Hardware needs further improvement to reduce the system's overall cost
[93]	Water sensor; Water flow sensor (YF-G1)	Solenoid valve; Water pump	Raspberry Pi 3 Model B+; MCP3008 (8-channels, 10-bit ADC Chip)	Not specified	Hardware not optimized; Sensor to detect water level is not reliable
[94]	HC-SR04	Motor pump	Arduino Uno; ESP8266 Wi-Fi transceiver	Arduino IDE; Webpage	Leakage detection not considered
[95]	HC-SR04	Electromechanical relay; Water pump	Arduino Uno; NodeMCU (ESP8266)	Arduino IDE; Blynk IoT-Platform	Leakage detection not considered; Hardware not optimized

Table 3. *Cont.*

Ref.	Sensors	Actuators	Processing Units	IDE	Comments
[96]	HC-SR04	Not used	Arduino Uno; GSM/GPRS SIM900A	Arduino IDE; UML	Only water level is monitored
[97]	HC-SR04; Water flow sensor	Solenoid valve; Water pump	Arduino Uno; GSM/GPRS SIM900A; Power bank	Arduino IDE; ThingSpeak IoT-Platform	Promising idea presented, but leakage is not entertained
[51]	US-020 (Ultrasonic sensor); Water sensor	Solenoid valve; DC micro diaphragm pump; Electromechanical relay	Arduino Mega 2560 kit (8-bit MCU); SIMCOM SIM900 modem; ULN2003 to control Relay	Arduino IDE; Android application	The overall system is sound, but its leakage detection unit needs further improvement.
[98]	Water sensor; Water flow sensor	Solenoid valves; Submersible water pumps; Relays	Arduino Nano kit (8bit MCU, ATmega328); Raspberry Pi2; GSM/GPRS SIM900A.	Arduino IDE; Adafruit Cloud IoT platform	Hardware has redundancy. Water levels are discrete, four only
[99]	HC-SR04	-	Raspberry Pi 3 Model B+	ThingSpeak IoT platform; Python.	Only water level is monitored
[100]	Magnetic float sensors	Water pumps; Relays	AT89C51 (8-bit MCU); GSM SIM800 Module	MIT App Inventor	Leakage not considered
[101]	HC-SR04	Water pumps; Relays	NodeMCU	SQL server	Leakage not considered. In addition, technical details are not enough
[102]	HC-SR04	Water pumps; Relay module	NodeMCU	Firebase real-time database; Fusion chart; Webpage (CSS, HTML and JavaScript)	Leakage not considered
[103]	HC-SR04; 5V Analog water pressure sensor	220VAC 2-way motorized electric ball-valve; Water pump; Relays	Arduino Uno; SIM800 GSM shield; Raspberry Pi 3 Model B+; micro-SD card; LED monitor, mouse, and keyboard	Arduino IDE; Raspbian operating system, Apache (Linux version), PHP (LAMP), MySQL, and Python	Leakage monitoring method is not effective
[104]	HC-SR04	Water pump; Relays	NodeMCU; Wi-Fi hot spot	Firebase IoT platform; MIT App Inventor	Leakage not considered
[65]	HC-SR04	Submersible water pump; Relay module	Arduino Uno; Raspberry Pi 3 Model B+	Python2; Flask; Webpage	Leakage not considered; technically poor. Hardware redundancy exist

Table 3. *Cont.*

Ref.	Sensors	Actuators	Processing Units	IDE	Comments
[105]	APG—Series PT-500-P1—Level Transmitters (proposed)	-	Arduino Uno R3; Arduino Ethernet Shield	Arduino IDE; Carriots IoT platform; Freeboard; REST API	Only water level is monitored
[106]	Single-stranded wire (as water level sensor) connected with transistor, BC547	Water pumps; Relays	CC3200MOD Simple Link (32-bit RISC ARM processor); Local Wi-Fi router	Energia IDE; CC320 Launchpad drivers; Uniflash software	Only four levels are monitored
[107]	HC-SR04	Solenoid valve; Water pump	NodeMCU; NI DAQ (USB-6009); Desktop PC	LABVIEW; Google IoT platform	Leakage not considered; Hardware is not optimized
[108]	HC-SR04; LED/LDR for turbidity; Water flow sensor	Water pump and Relays	NodeMCU	ThingSpeak	Technical details could be improved

Nikeeta et al. [93] proposed an IoT-based water management system to reduce water wastage in residential buildings. The authors considered monitoring water level, leakage control, and auto refilling tank. It is centered around the Raspberry Pi 3 Model B+ kit [87]. This kit is based on a 64-bit Quad-core processor, @ 1.2 GHz. In addition, it is also abundant in peripherals such as HDMI port, CSI camera port, USB ports, 40 I/O pins, Wi-Fi/Bluetooth/ethernet, 1GB memory, and more. However, it does not offer any on-chip ADC unit; due to this reason, the authors used an eight-channel, 10-bit ADC chip (MCP3008) while acquiring the relevant sensors data.

To monitor water levels, the authors used a water sensor (Figure 4). It has two types of metallic layers, i.e., power and sensors layers, isolated from each other and exposed to water. Touching these layers by the water body completes the circuit, and then voltage develops across a high ohmic resistor on this card. This change is read by Raspberry Pi via ADC chip to predict the water level in the tank. Data is then sent to a cloud server, not specified. However, to monitor water leakage in the tank, the authors installed two water flow sensors in the supply pipes of the water tank. The output of these sensors is read via an ADC by Raspberry Pi, and an alert is also generated if there is any significant difference. Though the block diagram of this system tips about the end-user interface module, it is not detailed explicitly.

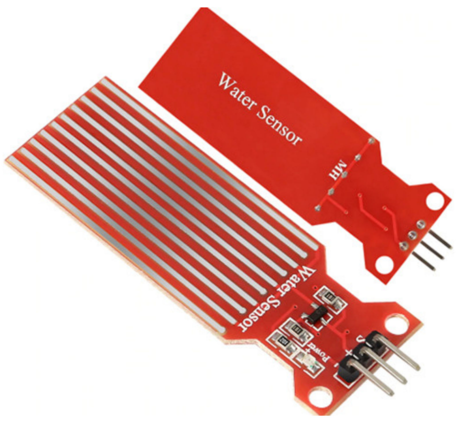

Figure 4. Water sensor used in [93], Courtesy of the Shenzhen Ke Zhi You Technology Co., Ltd., China.

In addition, this article is composed at the abstract level, where some important technical details are also missing. It has the following limitations. Firstly, Raspberry Pi 3 Model B+ 260 (price 260 USD) is costly compared with other IoT kits such as Arduino Nano 33 IoT (price 14 USD) and NodeMCU ESP3266 (prices 24 USD); prices were taken from the Google chrome, dated: 15 September 2021. Secondly, it is clocked @1.2GHz, which is far beyond the requirements of this application. Thirdly, the water sensor (Figure 4) used here is unreliable; its sensing part is exposed to water, due to which its metallic layers most often get rusty and erode. With this sensor, the resolution of water level is not as satisfactory as it could be obtained using an ultrasonic sensor.

Lade et al. [94] proposed an IoT-based water management system targeting water regulation in offices and buildings. In this study, both water level and tank auto refilling are monitored. This design is based on Arduino Uno, which works as follows: (1) Arduino Uno measures the water level in the target tank using an ultrasonic sensor; (2) after local processing, data is sent to an ESP8266 transceiver being interfaced with Arduino Uno. The ESP8266 is an SoC (System-on-Chip) having an embedded TCP/IP protocol stack necessary for an MCU while accessing Wi-Fi. In addition, it has four general-purpose I/O pins as well. It sends data to the webserver via a Wi-Fi router.

The cloud server compares this data with the preset tank's levels. If the water level is found below a lower threshold, the motor pump is turned on via Arduino Uno; otherwise, no action is taken. In case the motor is already on, the data is checked against an upper threshold. If it is found above the preset threshold, the motor is immediately turned off. Though the hardware used here is quite optimized, it has offered nothing to detect water leakage.

Lakshmi et al. [95] developed an android application for IoT monitoring water tanks. It allows the end-user to monitor the water level in the tank and control water pumps whenever required. This design is also based on Arduino Uno, supplemented with the NodeMCU (ESP8266). Note, the heart of NodeMCU is an ESP8266 transceiver centered around a 32-bit MCU, with numerous peripherals such as a Wi-Fi module, I/O pins, one ADC pin, etc. Figure 5 shows a typical ESP8266 transceiver and NodeMCU (ESP8266) module.

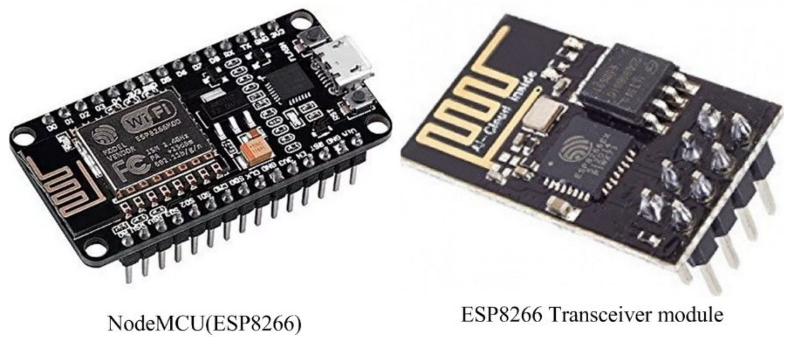

Figure 5. NodeMCU (ESP8266) and ESP8266 Transceiver [85], Courtesy of the Espressif Systems, China.

In this system, Arduino Uno first measures water level through an ultrasonic sensor. After local processing, data is shifted to NodeMCU for uploading to the Blynk IoT platform [79] through a Wi-Fi router. The authors developed an android application (Blynk App) for end-users' smartphones to monitor water levels and control water pumps. While using Blynk App on smartphones, end-users can easily monitor water levels and turn the pump on and off whenever required. This system has a few limitations. Firstly, it has no provision to detect water leakage. Another drawback is its hardware redundancy. For example, Arduino Uno is an 8-bit MCU, and NodeMCU is based on 32-bit MCU; thus, one of these could be skipped without any loss of generality.

In [96], Parimala et al. proposed a solution to monitor water storage facilities based on IoT principles. In this study, only the water levels in tanks are monitored. Arduino Uno supplemented with GSM/GPRS SIM900A shield is the central part of this system. Uno first measures the level of water in the storage container (e.g., Tank) through an ultrasonic sensor via an ADC pin. Next, it processes this data locally and updates a local LCM. In case of any irregularity, it can also turn on the local buzzer. To update data on the web portal, it uses GSM/GPRS SIM900A. End-users can also inquire about water levels through a smartphone. This system was developed in Arduino IDE, and for the webpage, the designing authors used the unified modeling language (UML). It was also validated on the water in the bottle, tank, and pool. Though its architecture and method are well-explained, it is not the best solution for smart monitoring of water tanks. Authors need to incorporate modules for leakage detection and tank auto refill.

Durga et al. [97] published an article on automatic tank refilling using the IoT addressing the issues such as turning on the water pump when the main water supply and power (electricity) are both available. In addition, they also focused on controlling a situation in which tank filling is in progress when the main supply of water stops or electricity is cut off. The proposed measures may help resolve issues such as a DRY-RUN motor. If electricity is in cut off, a relevant notification is initiated to its end-users, allowing them to make an alternate arrangement if possible.

This system is energized by a power bank made of several dried DC batteries. First, Arduino Uno checks the water level in the tank through an ultrasonic sensor. If the water level is above the maximum threshold, no action is taken, and the status is updated on the local LCM and in ThingSpeak IoT platform via GSM/GPRS SIM900A. Otherwise, it first checks the availability of electricity and the presence of water in the main supply pipes using a water flow sensor. If both entities exist simultaneously, the system starts filling the tank; otherwise, it notifies end-users about this anomaly. In addition, this notification is also updated on ThingSpeak and the local LCM.

To summarize, the authors presented a workable idea to prevent the motor from running when the main supply has no water and notify the end-users through an SMS in case power is in cut off mode. However, this system does not consider water leakage.

Daadoo and Daraghmi [51] worked on smart water leakage detection in homes and buildings and monitored water levels and storage tanks auto-refill, which is centered around the Arduino Mega2560 kit [86]. This gadget is an advanced version of Arduino Uno. It is based on an 8-bit MCU, offering relatively more features. To communicate wirelessly between a smartphone and Arduino Mega2560, the authors employed the SIMCOM SIM900. They used US-020 (an ultrasonic sensor) while monitoring the water level in the overhead water tank. In addition, they also used the ULN2003 chip to control the solenoid valves.

To detect water leakage, the authors installed multiple water sensors along the water supply pipes running to the garden, living room, bathroom, master bathroom, kitchen, and main bathroom. To fill the water tank, they used a high-quality DC micro diaphragm pump. To monitor water leakage, level, and tank refill, they developed an android application for smartphones. The ATmega2560 reads the water level using an ultrasonic sensor. A preset threshold is used to decide when to turn on the motor to refill the tank and forward an alert to end-users. In case any water leakage is detected, the system shuts off the corresponding water supply or other venues through the solenoid valves. The event is also reported to end-users. Though this system demonstrated optimal results during experimentations, it has some limitations. For example, water sensors used for leakage detection are not reliable because of the reasons mentioned earlier. In addition, it does not monitor water leakage.

Jaiad and Ghayyib [98] developed an IoT system for monitoring water levels, tank auto refill, and water theft. In this system, the authors monitored refilling an overhead tank from an underground reservoir via an electric motor interfaced with an electromechanical relay. In addition, they also monitored water theft or significant leakage in supply pipes using water flow sensors. To monitor water levels at discrete levels, the authors installed four water sensors (Figure 5) at different points in the tank. Similarly, they used two water flow sensors installed in main supply pipes to detect water supply cutoff, leakage, or water theft.

All sensors are read by the Arduino Nano card [86] centered around ATmega328, an 8-bit MCU. While similar to Arduino Uno, this card is specifically customized for smaller-size products. Data from Arduino Nano is transferred to the Raspberry Pi2 card [87], which is centered around a 32-bit CPU, i.e., 900 MHz BCM2836 Quad-core ARM CortexA7, with 1GB RAM, micro-SD card, USB, 40 I/O pins, etc. Raspberry Pi2 updates a local LCM and shifts data to the Adafruit IoT platform [49] through the GSM modem. Data is stored, analyzed, and visualized in this cloud server using the readymade visual widgets.

In case the water supply is shut off or water in the supply lines is reduced to a low level due to high leakage or illegal connection to pipes, the difference in the output of water flow sensors is detected by the Arduino Nano card. For water control, the authors used the solenoid valves. A major disadvantage of this system is its hardware redundancy. The authors involved two processor-based kits in conducting this simple task. Since data is analyzed in the Adafruit IoT platform, there is no need to involve Raspberry Pi2 [87].

Sowmya et al. [99] proposed an IoT system for monitoring water tanks. In brief, the authors used an ultrasonic sensor to gauge the water level in the tank. To read water levels, they used the Raspberry Pi3 Model B+ kit [87]. As mentioned earlier, this is a complete System-on-Chip (SoC) centered around a 64-bit microprocessor. Utilizing the system's built-in Wi-Fi, the sensors' data is transferred to the ThingSpeak IoT platform [79] via a local Wi-Fi network. Authors developed a dashboard in the ThingSpeak server to provide visual displays for the water level, accessible by end-users with the ThingSpeak App installed on their smartphones. Data on the dashboard is refreshed every two seconds. The article shares little in terms of technical detail and focuses mainly on the water level monitoring, offering no consideration of the leakage detection and tank auto refilling.

Pawaskar et al. [100] proposed a hybrid of GSM and a web-based solution for monitoring the water level in tanks. In this article, the authors used an 8-bit MCU (Atmel89C51), GSM SIM800, magnetic float sensors (Figure 6), water pump equipped with actuators, etc. To measure the water level at three discrete levels, the authors installed three magnetic float sensors at different positions in the water tank. As shown in Figure 6, the magnetic float sensor has a movable annular object, which can move towards the wiring-end when water in the tank arises. If water does not touch this object, then the switch remains in its closed state, and the two electrical wires show continuity; otherwise, the circuit remains open. While this sensor is robust against corrosion and biofilms (deposits such as fungi), its main flaw is poor resolution. For every discrete level in the tank, one such unit is needed. An ultrasonic sensor would be its best fit.

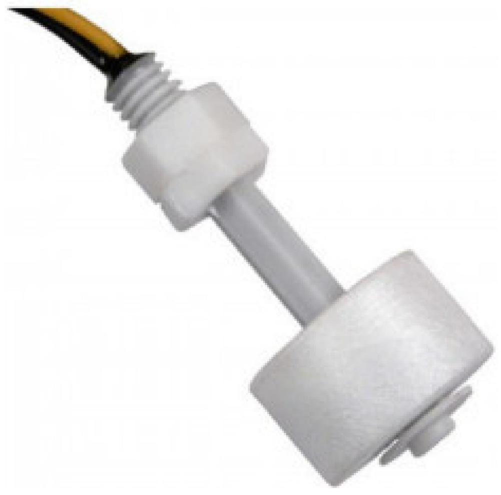
Figure 6. Magnetic float sensor [100], Courtesy of the Dongguan Fuen Electronics Co., Ltd., China.

While monitoring water levels in the tank and controlling the water pump, the authors developed a mobile application using the MIT App inventor [109]. This tool is essentially a web application, an IDE initially developed by Google and now maintained by the Massachusetts Institute of Technology (MIT). In brief, this tool allows developing applications for smartphones using the web browser and a connected phone or emulator. The main limitations of this scheme involve not using the proper cloud services to analyze sensors' data and not monitoring the water leakage.

Gupta et al. [101] proposed an IoT-based system for monitoring and controlling water consumption in residential buildings, organizations, and corporations, where the target area is divided into different blocks (e.g., A, B, C, . . . , and Z). In each block, tanks are labeled according to building numbers, e.g., Building1, Building2. For each tank, the authors used the NodeMCU (ESP8266) and an ultrasonic sensor to monitor the water level and share data with the server. Every node is equipped with NodeMCU and local Wi-Fi.

To record, analyze, and visualize data, the authors developed numerous databases in the SQL (Structured query language) server. Water in each tank/building is controlled by a motor installed in the maintenance block. Each motor is actuated through an electromechanical relay. If the water in the tank falls below a predefined threshold, the server sends a command to the maintenance block to turn on the corresponding motor. The motor switches off when the respective tank is full. Though this is a good scheme, it has some limitations; the system does not detect water leakage from the tanks. In addition, the authors did not supply specific technical details, raising the question of how the system could be reimplemented if required.

Dissanayaka and Wickramaarachchi [102] proposed NodeMCU (ESP8266) based system for monitoring water tanks. While an ultrasonic sensor monitors the water level, the tank is auto refilled through NodeMCU interfaced with a motor through a relay module. For sending data to the cloud server, the authors utilized the built-in Wi-Fi function of NodeMCU and a local Wi-Fi router. The water level data, motor on and off status, and the volume of water in the tank are visualized in the Fusion-Chart package, which is accessed through the Firebase real-time databases and tools such as the CSS, HTML (Hypertext Markup Language), and JavaScript. The authors utilized the following Equation (1) to compute the water volume in a circular tank:

$$V_{Water} = \left(\pi r^2\right) L_{Water}, \tag{1}$$

with $L_{Water} = h_{Tank} - L_{Empty}$. Here, V_{Water}, $\left(\pi r^2\right)$, r, L_{Water}, h_{Tank}, and L_{Empty} represent the volume of water present in the tank, tank cross-sectional area, tank radius, current water level, tank height, and the total empty portion of the tank, respectively. The same Equation could also be used for the rectangular or other types of tanks but with slight modifications according to the tanks' structure. End-users can also monitor water tanks via the webpage, which is accessible through smartphones having connectivity to internet. This system did not consider the issue of water leakage.

Natividad and Palaoag [103] proposed an IoT-based model emphasizing providing a low-cost and efficient system to improve water distribution for communities. Its overall model includes two stages: (i) client site and (ii) control room. In brief, authors in this model tried to monitor water level, auto-refill tanks, regulate water pressure in supply pipes to avoid leakage and burst, and give full access to the control room.

As presented in Table 3, the water level in tanks is gauged through the ultrasonic sensors, pressure in the supply pipes via a 5V analog pressure sensors, water in pipes is regulated through solenoid valves, and the tank is refilled via an electric motor actuated by electromechanical relay. On the client side, all sensors are looked after by Arduino Uno supplemented by a GSM SIM800 modem to report data to a control room server. The authors configured Raspberry Pi3 Model B+ as a server in the control room, equipped with an LED (Light Emitting Diode) monitor, USB mouse, keyboards, and internet/GSM capabilities.

First, Arduino Uno reads all sensors via its I/O ports and transfers data to the control room through a GSM modem. In addition, it can also refill the tank if the water level falls below a lower threshold. On the other hand, the server has also been equipped with the Fuzzy logic algorithm to make a better decision, e.g., refilling a tank, regulating water flow in the pipe, and sending alerts to the overseer or end-users. To summarize, this model is well-planned and provides a good solution to control and monitor water distribution in towns, organizations, corporations, etc. However, a significant limitation of this design is the inadequate leakage detection or control method. Authors only regulate water pressure in the supply pipes to avoid bursts and leakage.

Shah et al. [104] proposed an IoT-based system for monitoring water tanks using an android application. The NodeMCU first fetches water levels from the Firebase IoT platform [49] using a local Wi-Fi router. Next, it gauges the water level in the tank through an ultrasonic sensor. If the water level is below a preset threshold specified in the Firebase server, the ESP8266 turns the water pump on via an electromechanical relay. The water tank is automatically refilled according to the preset threshold set in the Firebase cloud server by end-users using smartphones. To update the parameters on the cloud platform, the authors used the MIT App Inventor to develop an android application for smartphones. While the end-user has full control over the water tank, this system has no provision for water leakage monitoring.

Gunde et al. [65] proposed an IoT system for monitoring water distribution in the campus setting. The main interest of their work is the control and monitoring of the overhead tanks through the IoT. The system programming, testing, and validation were conducted in the Python2 and Flask platforms. The authors also developed a dashboard to display the water level in the ThingSpeak IoT platform. In addition, end-users can watch the water level in the tank using their smartphones. This system functions as follows. First, Arduino Uno reads water levels through an ultrasonic sensor. If it is below a lower threshold, the motor turns on; otherwise, it stays off. Sensor's information is shifted to Raspberry Pi kit through the serial port, which uploads this data onto ThingSpeak IoT-Platform using a local Wi-Fi router. End-users can also receive SMS alerts and watch the water in the tank through the ThingSpeak App installed on their smartphones. In terms of its shortcomings, this system does not monitor water leakage detection and comes with a higher cost. The authors used two kits for this simple task—the Arduino Uno and Raspberry Pi—each centered around a processor. However, this redundancy could be removed to reduce the system's overall cost.

Malche and Maheshwary [105] proposed an IoT system to monitor smart villages' water levels. This system centered around the Arduino Uno R3, supplemented with an Arduino GSM shield. For data storage and analysis, the authors used a hybrid of the Carriots IoT platform [49] and the Freeboard, a type of dashboard offering numerous widgets for the visual presentation of sensors' data.

With Carriots, end-users can receive SMS, emails, or tweets. The system sends data to the Freeboard in the JSON format to visualize it according to the chosen widgets. The REST API was used to set up communication between Carriots and Freeboard. While the authors used a simple liquid level-sensor with limited range, they also suggest a submersible pressure transducer (e.g., APG series PT-500-P1 level transmitters) capable of measuring water levels as deep as 450 feet.

Wadekar et al. [106] presented a smart solution for water management in overhead tanks, especially in urban areas. This system is centered around the Launchpad, CC3200MOD Simple Link, based on a 32-bit RISC ARM processor. While capable of running at 80 MHz, this processor has many on-chip peripherals such as Wi-Fi module, I/O pins, ADC, UART (Universal Asynchronous Receiver Transmitter), and I2C (Inter-integrated circuit). It is a suitable choice for many IoT applications.

The authors used four NPN transistors (BC547) configured in the switch mode to detect the water level in the tank through stranded wires tied to their base terminals. They inserted an electrical stranded wire tied to the positive terminal of a battery deep into the tank. In addition, they also installed four electrical stranded wires at different points in the tank, where each wire is bonded with the base terminal of a transistor. When the water level rises, the wires connected with transistor bases come in touch with positive voltage through the water as conducting media. This act triggers the concerned transistor to turn on. These states are displayed locally through four LEDs.

To visualize and store the data on the cloud, the authors used the Dweet.io IoT platform, with the JSOUP for API. In addition, they developed an android application for the smartphone to view the water level in the tank. Users have no control of the on/off motor switch, nor can they configure the water level. System programming was completed in the Energia IDE. In terms of its limitations, the system has no strategy for monitoring water leakage and can monitor only four discrete water levels in the tank. Most significantly, the electrical stranded wires used to detect water levels were undoubtedly a cost-effective short-term solution but may soon suffer from the biofilm effects and erosion due to the organic and inorganic materials present in water. In addition, the authors directly inserted a reference electrical wire tied to a positive battery terminal which may cause overconsumption of the battery energy.

Charles et al. [107] proposed an IoT-based system integrated with the LABVIEW software to monitor water levels in tanks for large areas.This system can monitor a number of tanks simultaneously. The authors used an ultrasonic sensor for each tank to detect water level and NodeMCU for transferring sensors data to the Google cloud server [49] via a local Wi-Fi router. In the Google IoT platform, data is stored and analyzed. In addition, each tank is equipped with a water pump and a solenoid valve. While controlling water level and refilling tanks, authors developed a separate station centered around the National instrument data acquisition card (NI DAQ (USB2009)). This card is interfaced with a desktop PC through the USB port. For this station, programming was performed in LABVIEW, a graphical programming media. The NI DAQ has many analog and digital I/O pins to which each tank's solenoid valve and water pump are connected.

First, NodeMCU reads the water level from the tank through an ultrasonic sensor, reconditions and updates data into the Google IoT platform. Next, the main station fetches data from the Google IoT platform and determines if any valve/motor needs to be served. While using LABVIEW and NI DAQ is a promising idea, the authors also introduced significant redundancy in the overall system hardware. Each tank is equipped with NodeMCU, a perfect choice for many standalone IoT applications because it has numerous futures such as the Tensilica 32-bit RISC CPU Xtensa LX106, ADC, DAC, digital I/O pins, SPI, I2C, flash memory, SRAM, USB interface, and the PCB antenna. Therefore, there is no need to control the valve and water pump using the NI DAQ card. Besides, they did not monitor water leakage and could not provide access to end-users, except via the main station only.

Asif et al. [108] proposed an IoT solution to monitor the household network of water. In this article, the authors monitored three parameters: the water level, the tank auto-refill, and water leakage. In addition, they also monitored the ho clean the tank was through a hybrid of LED and a light-dependent resistor (LDR) installed inside the tank. For leakage detection, they installed water flow sensors supplemented with NodeMCU at each branch of the water distribution system. In addition, they used an ultrasonic sensor to monitor water levels in the tank. Data is recorded into the ThingSpeak IoT platform using local Wi-Fi, and end-users can access it using smartphones. Water consumption and leakage detection are supervised using sensors data fusion and machine learning (ML) schemes. For example, data from water-flow sensors installed near the water tap and ultrasonic sensors could be used to predict water consumption.

While developing an ML scheme for leakage detection, authors first collect data from the sensors. Then, assuming no leakage occurred for some definite time, upload the data to the ThingSpeak IoT platform, and train the model to infer the decision-marking parameters. In this case, the authors used an implicit relationship between the water height in the tank and water flow in pipes. If leakage is found, the system informs the end-users via smartphones. To monitor water usage, data from sensors is collected and uploaded into the ThingSpeak. Full and empty tank status is also reported to end-users via mobile application.

In addition, data from the LED/LDR circuit is used to predict the tank dirtiness; status is reported to end-users.

4. Recent Technologies and Techniques

This section presents several well-known technologies and techniques commonly used in the IoT–WST.

4.1. Water Level Monitoring

Monitoring the water level in the tank is considered an essential parameter in smart monitoring of water storage tanks. As both cemented and portable tanks exist in different capacities and heights, precise water level measurement may be challenging due to the sensors' range limitations. As per contemporary literature [65,69,101–103,107,109], some commonly used devices for level monitoring are a water sensor (Figure 4), magnetic float sensor (Figure 6), ultrasonic sensor (Figure 7a), and light detection and range (LIDAR) sensors (Figure 7b).

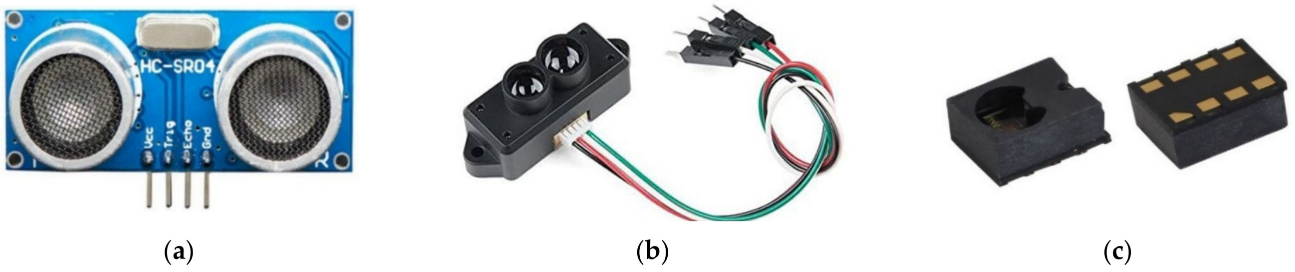

Figure 7. (**a**–**c**) typical models of an Ultrasonic sensor (e.g., HC-SR04), TF-mini-LIDAR (e.g., SJ-GU-TFmini-01), and Vertical-Cavity Surface-Emitting Laser (e.g., VCNL36826S), [86]; Courtesy of the Holykell Technology Company Limited, China; Shenzhen FEETECH RC Model Co., Ltd., China; and the Frankfurt Laser Company—Friedrichsdorf, Germany.

While monitoring the water level in tanks, developers also involved discrete type sensors (e.g., water sensor, Figure 4) that are generally installed at different points inside a tank. As mentioned earlier, in this case, the spatial resolution of the water level is not granular. More sensors need to be installed at different points to enhance the resolution, which is likely an expensive task and may increase the overall burden of maintenance and product cost.

To resolve this issue, researchers involved the distance (proximity) sensors. Such sensors use different technologies such as infrared (IR) triangulation, laser, ultrasonic, LED-TOF (light-emitting diode-time-of-flight), and others. Notably, the choice of the sensor strongly depends on the model of the target application. These sensors offer a range of different properties, e.g., resolution, frequency, field-of-view (FOV), transmission–reception durations, installation, and costs. To detect objects in proximity, these sensors first send waves (e.g., laser, infrared, or sound waves) and then wait for the reception of waves bounced back by the target objects. On reception, the distance of the target objects is gauged based either on the intensity or time taken by the waves to hit objects and come back to the source.

Figure 7a–c show typical models of the ultrasonic, TF-mini-LIDAR, and a fully integrated proximity sensor (VCNL36826S), respectively. Though each of these sensors can provide acceptable spatial resolution while monitoring water level, the range of LIDAR is relatively higher, as highlighted in Table 4. As LIDARs are costly, these devices are only recommended when the height of a water storage container is more than 4 m. To summarize, proximity/distance sensors are reliable tools for various applications requiring fast and accurate measurement, positioning, or discovery of solid or liquid matter.

Table 4. Typical ultrasonic, LIDAR, and VCSEL proximity sensors.

S. No:	**Ultrasonic Sensor**	**TF-Mini-LIDAR**	**Vertical-Cavity Surface-Emitting Laser (VCSEL)—VCNL36826S**
(a)	Signal: 40 kHz sound pulses	Operating center wavelength: 850nm (Infrared light centered around); Test frequency: 100 Hz	940nm peak wavelength of VCSEL; Maximum Frequency: 100 kHz
(b)	Supply voltage: 5 VDC	Supply voltage: 4.5~6 VDC	2.62 V–3.6 V
(c)	Working current: 15 mA	LED peak current: 800 mA; average power consumption: 0.12 watt	Typical current: 20mA
(d)	Range: 0.02~4.0 m	Range: 0.03~12 m Maximum operating range at 10% reflectivity: 5m	Range up to 200 mm (7.87402 inches)
(e)	Measuring angle: 15°	Acceptance angle: 2.3°	60° Proximity sensor view angle
(f)	Trigger input signal:10µS TTL pulse	-	-
(g)	Dimension: 45 × 20 × 15 mm^3	Dimension: 45 × 15 × 16 mm^3	2.55 × 2.05 × 1.0 mm^3 (L × W × H)
(h)	Typical cost: 2.13~7.7 USD	Typical cost: 40~53 USD	Typical cost: 2.77 USD
(i)	Communication interface: Digital I/O	Communication interface: UART	I2C bus output type
(j)	-	-	12-bit ADC for signal conditioning

Note: As minor variations could be possible in electrical and nonelectrical characters for different models, consult the specific datasheet for precise information.

4.2. Water Leakage Monitoring

In water-scarce countries, water is often stored in the overhead or underground tanks, filled through the water supply lines from a local municipality, tube wells, or bores. It is observed that water is often wasted from the tanks, supply lines, or tubes due to the aging infrastructure, misuse, or some other technical issues. In addition, water is also wasted through lack of attention and care, causing overflowing of water tanks. The following briefly explains some commonly used water leakage detection techniques:

- Portable Sensors: In numerous studies [59,60,110], water sensors were used (Figure 4) to detect water leakage. Generally, one sensor is sufficient for each potential point where water leakage is strongly expected to occur. As mentioned earlier, this type of sensor has two sets of parallel naked metallic layers, where one set is connected with a positive voltage terminal and the other one tied to the negative voltage terminal of the power supply. When its sensory part meets water, this event changes the analog output of this sensor.

Moreover, IoT developers [59,60,110] also utilized water moisture sensors (Figure 8) while detecting water leakage. As shown in Figure 8, these are of two types. The one shown in Figure 8a is not sufficiently reliable because it may rust when its naked metallic layers are exposed to water. The second type (Figure 8b) is more reliable as its sensing unit has no direct exposure to water.

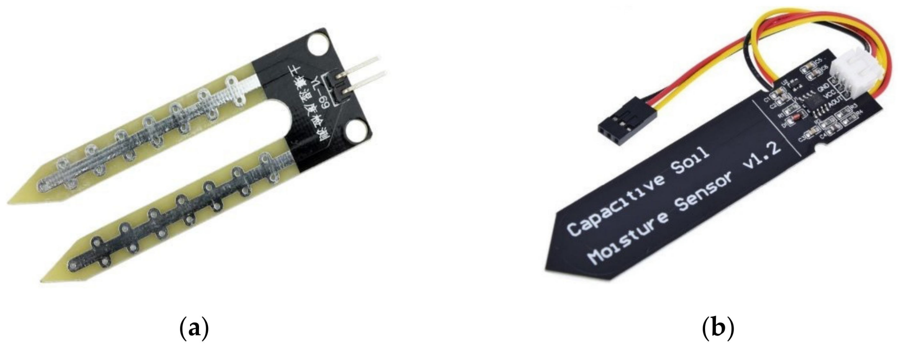

Figure 8. (**a**,**b**) a typical YL-69 water sensor and a capacitive soil moisture sensor, respectively [86], Courtesy of the Shenzhen Haiwang Sensor Co., Ltd., China.

- **Water Flow Sensors**: Figure 9 shows typical hall-effect-based flow sensors, which are used in many applications such as measuring the flow of water and oil, DIY coffee machines, water vending machines, etc. These sensors come in different sizes and ratings, per the system's requirements. The main components of a flow sensor include the hall-effect sensor, turbine wheel (also called rotor), and magnet. When water flows through the valve, it rotates the turbine that produces the magnetic field. This change in the magnetic field is sensed by the hall-effect sensor, producing square pulses.

To detect water leakage, developers use a set of two flow sensors that are installed at both starting and endpoints of the target pipe. To monitor these sensors, developers use IoT devices (e.g., BBC Micro-Bit, ESP32, NodeMCU, ESP8266, and Arduino Nano 33 IoT). These devices measure the quantity (liter per hour or cubic meter) of water flowing through flow sensors at both points and then communicate data to each other or a master unit using GSM module, Wi-Fi, Xbee, Bluetooth, LoraWAN, nRF24L01, RF334, or ethernet [64,100,111,112]. If both readings stay approximately the same, it implies no water leakage or theft. Otherwise, either water from the concerned pipe is illegally diverted, or some leakage occurs. To summarize, this technique seems workable because issues related to hidden water supply pipes (e.g., pipes buried in walls) could be monitored more conveniently.

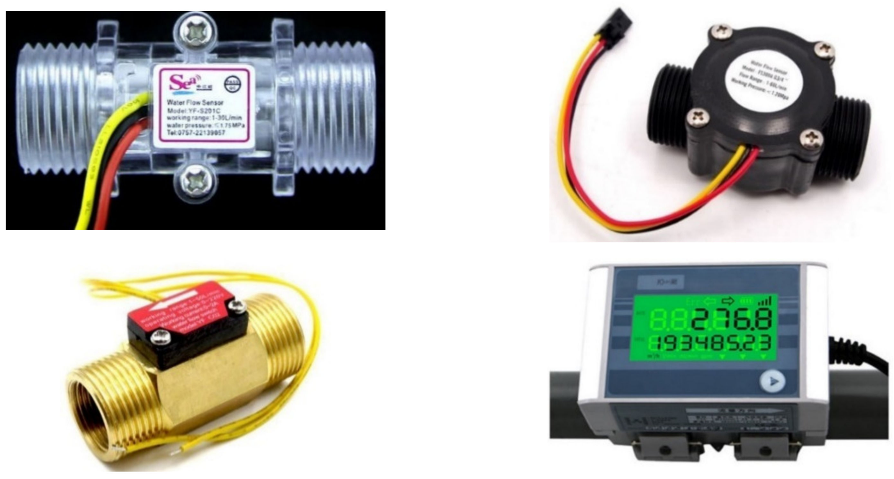

Figure 9. Some typical flow sensors [75,113], Courtesy of the Hunan Mac Sensor Co., Ltd., China; and Ningbo Mingrui Zhongxing Electronics Technology Co., Ltd., China.

- **Digital Signal Processing:** In addition to the above-mentioned schemes, researchers also devised numerous digital signal processing-based techniques to detect water leakage [51,67,68,80,108]. Whenever an abrupt leakage occurs in the supply line, this drastic change produces a spike in a digital filter (e.g., Kalman filter). This spike is a sign of water leakage.

As highlighted in Section 3, leakage from the water tank could be monitored [50] first by suspending the water supply from the tank to all the sinks for a finite duration. During this period, the water level in the tank is sampled at regular intervals. In this case, any difference in two consecutive samples would be an important indicator of water leakage from concerned tanks. Though this is a workable idea, its drawback lies in blocking water supply to relevant consumers.

4.3. Tanks Auto Refilling

Automated refilling of water storage tanks is a feature of IoT–WST. As highlighted in Section 4.1, the water level in tanks is monitored using portable sensors generally interfaced with computing devices, e.g., NodeMCU, Arduino Uno, and ESP8266 Transceiver.

- Dry-Run: This means the water pump or motor attempts to refill the tank, but water is not available in the source (e.g., supply line from local municipality). This state should be avoided because it may increase the electricity bills.
- Electromechanical Relays: Relays are often utilized to energize water pumps or motors. As shown in Figure 10a, the layout of each device is printed on its casing, or the device

may have a transparent casing through which designers may note its connection. For more details, readers are recommended to consult the technical specifications of target relays on its vendor's webpage or released documents.

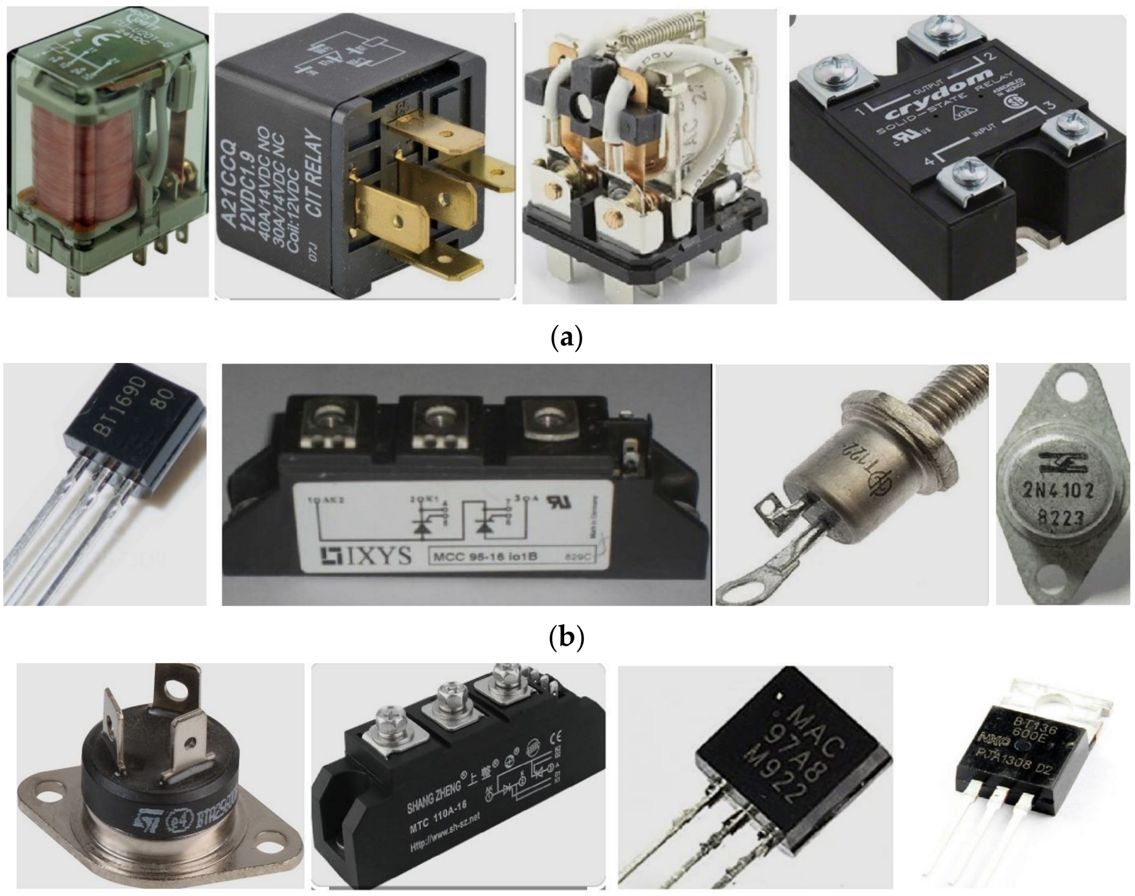

Figure 10. (**a**–**c**) Some typical electromechanical relay, SCRs, and TRIACs, respectively [75,113], Courtesy of the Zhejiang NCR Industrial Co., Ltd, China; and EASTRONIC THCHNOLOGY CO., LTD., China.

- Thyristors: As relays are electromechanical devices, their performance is defined in terms of switching. In addition, the metallic contact of inexpensive relays (except vacuum type relays) may also become damaged due to the electric arching.

To cope with these issues effectively, IoT developers should use thyristors (e.g., silicon-controlled rectifier (SCR) and TRIAC), Figure 10b,c. As these devices are based on semiconductor technology, they should function optimally for a long time if appropriately utilized. Notably, controlling water pumps or motors using SCR/TRIAC is complex compared with relays (refer to the proper datasheet of each device before its application).

5. Challenges and Trends in IoT

Both software and hardware platforms have advanced over the past decades, creating a challenge for IoT developers to select appropriate media while implementing a target project. Extensive experience or a quick but comprehensive review of contemporary solutions and developments in this area may help resolve the difficulty.

Due to the latest advances in the semiconductor industry, our world is now dominated by electronic devices, e.g., portable sensors, tablet PC, cellphones, Wi-Fi, etc. In addition, electronic devices also significantly benefit from miniaturization and improvement in silicon technology, power consumption, portability, and reliability. As developing a hardware system involves cost, IoT developers should select the best choice for the target hardware so that the balance between cost and performance is not a compromise. In addition, software platforms have gone through significant advances, improvements, and modifications over the past decades. For example, new languages and software development platforms have been launched over the past several decades. The following elaborates a few typical challenges IoT developers are facing today.

5.1. Hardware

In IoT design, the main task of a digital computing gadget is to read sensors data from its environment, process it locally if required, and then transfer it to the target cloud server via local gateway (Wi-Fi) router, ethernet, or cellular network (i.e., GSP/GPRS module). To accomplish this task, there exist a plethora of commercially available digital development kits, which are based on 8-bit, 16-bit, 32-bit, or 64-bit processors. Figure 11 shows some typically electronic kits often used in IoT designing. Table 5 shows a brief comparison of some expressive features of these kits.

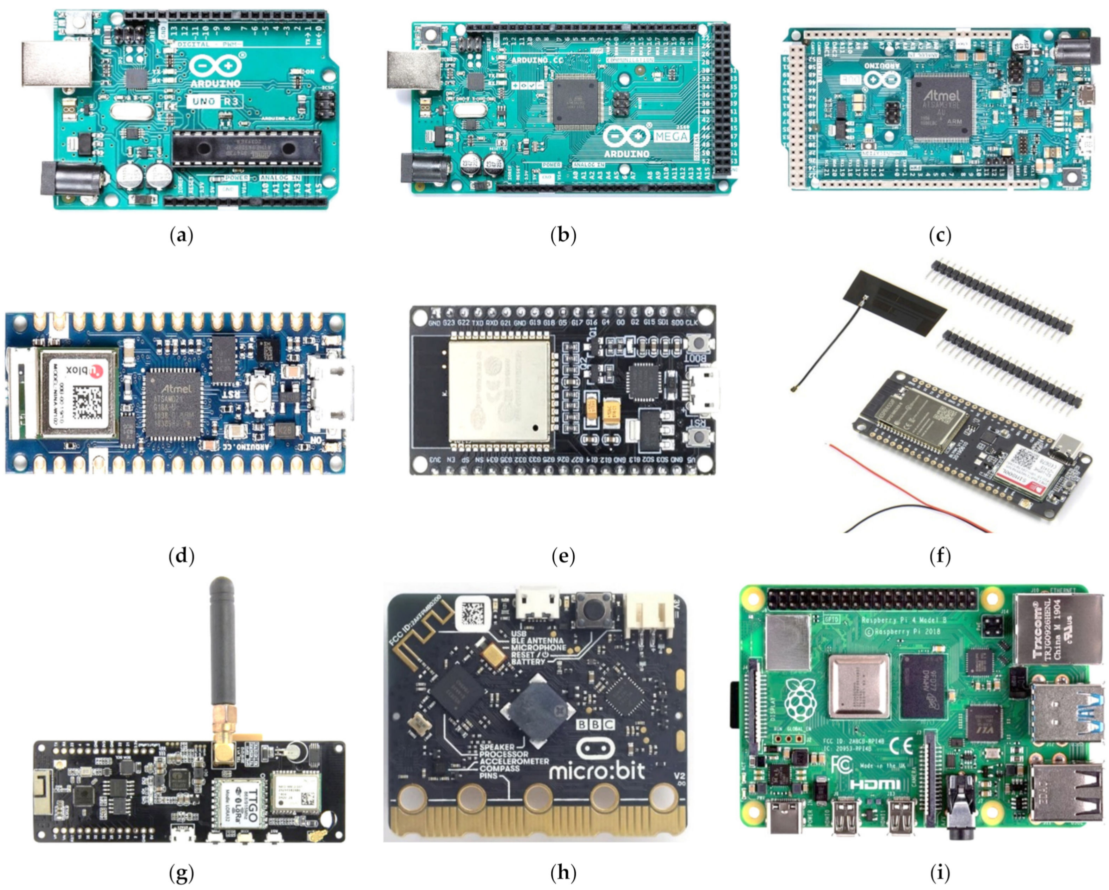

Figure 11. (**a**–**i**) Arduino Uno R3 [86], Arduino Mega 2560 [86], Arduino Due [86], Arduino Nano 33 IoT [86], ESP-WROOM-32S DEV (also called NodeMCU (ESP32S)) [85], TTGO T-Call ESP32 with SIM800L GPRS module [85], TTGO LoRa32 SX1276 board [85], ESP32 TTGO T-Beam V1.1 [85], BBC Micro: bit V2 [114], and Raspberry Pi 4 Model B—8 GB [87], respectively; Courtesy of the Sunhokey Electronics Co., Ltd., China; Espressif Systems, China; and Zhongshan Baijia Dagu Electronic Technology Co., Ltd, China.

Table 5. A comparison of contemporary electronics development kits.

S. No:	Development Kits (Significant Features of Kit, Not Processor)		Comments
(1)	Development kit:	Arduino Uno Rev3 [86]	ESP8266 Transceiver or GSM/GPRS needed for IoT applications.
	CPU:	ATmega328P (8-bit microcontroller)	
	Frequency:	16 MHz	
	Memory:	2KB SRAM, 1KB EEPROM, and 32KB Flash	
	Peripherals:	14 Digital I/O pins (16 PWM) and 6 ADC pins	
	Communication:	I2C and SPI	
	Built-in security:	Nil	
(2)	Development kit:	Arduino Mega 2560 Rev3 [86]	ESP8266 Transceiver or GSM/GPRS needed for IoT applications.
	CPU:	ATmega2560 (8bit microcontroller)	
	Frequency:	16 MHz	
	Memory:	8KB SRAM, 4KB EEPROM, and 256KB Flash	
	Peripherals:	54 digital I/O (15 PWM) and 16 ADC pins	
	Communication:	URAT0, UART1, UART2, UART3, I2C, and SPI	
	built-in security:	Nil	
(3)	Development kit:	Arduino Due [86]	ESP8266 Transceiver or GSM/GPRS needed for IoT applications.
	CPU:	32-bit Atmel SAM3 × 8E ARM Cortex-M3 CPU	
	Frequency:	84 MHz	
	Memory:	96KB SRAM and 512KB Flash	
	Peripherals:	54 digital I/O (12 PWM), 12 ADC pins, and 2 DAC pins	
	Communication:	SPI, UART, and I2C	
	built-in security:	Nil	
(4)	Development kit:	Arduino Nano 33 IoT [86]	It is specifically developed for the IoT products
	CPU:	SAMD21 Cortex®-M0+ 32-bit low power ARM MCU	
	Frequency:	48MHz	
	Memory:	32KB SRAM and 256KB Flash	
	Peripherals:	14 digital I/O (11 PWM)	
	Communication:	Wi-Fi and Bluetooth® via u-blox NINA-W102; UART, I2C, and SPI	
	built-in security:	Microchip® ECC608 crypto chip	
(5)	Development kit:	SP-01S ESP8266 Wi-Fi Module [85]	In standalone mode, it can control devices using four digital I/O pins. In general, it can provide services to other Kits such as Arduino Uno, Mega, and Due while controlling things in IoT.
	CPU:	Tensilica L106 32-bit RISC processor	
	Frequency:	80MHz, 160MHz	
	Memory:	1 MiB of built-in flash; 32 KiB instruction RAM, 32 KiB instruction cache RAM, 80 KiB user-data RAM and 16 KiB ETS system-data RAM; External QSPI flash: up to 16 MiB is supported (512 KiB to 4 MiB typically included).	
	Peripherals:	Four digital I/O	

Table 5. *Cont.*

S. No:	Development Kits (Significant Features of Kit, Not Processor)		Comments
	Communication:	Wi-Fi with on-board antenna; it supports Station/SoftAP/SoftAP + Station wireless network mode	
	built-in security:	IEEE 802.11 standard security features all supported, including WFA, WPA/WPA2, and WAPI; Secure boot; Flash encryption 1024-bit OTP, up to 768-bit for customers; Cryptographic hardware acceleration: AES, SHA-2, RSA, elliptic curve cryptography (ECC), random number generator (RNG).	
(6)	Development kit:	NodeMCU ESP8266 [85]	Specifically developed for the IoT applications
	CPU:	Tensilica 32-bit RISC CPU Xtensa LX106	
	Frequency:	80 MHz	
	Memory:	64KB SRAM and 4MB Flash	
	Peripherals:	11 digital I/O, PWM pins, 1 ADC pin,	
	Communication:	UART, SPI, I2C; Wi-Fi module, with onboard antenna	
	built-in security:	IEEE 802.11 standard security features all supported, including WFA, WPA/WPA2, and WAPI; Secure boot; Flash encryption 1024-bit OTP, up to 768-bit for customers; Cryptographic hardware acceleration: AES, SHA-2, RSA, elliptic curve cryptography (ECC), random number generator (RNG).	
(7)	Development kit:	ESP-WROOM-32 DEV KIT [85]	Specifically developed for the IoT applications
	CPU:	Tensilica Xtensa 32-bit LX6 microprocessor	
	Frequency:	Up to 240 MHz	
	Memory:	520 KiB SRAM, 448KiB ROM, 8KiB RTC Fast RAM, 8KiB RTC Slow RAM, and 1KiB eFuse	
	Peripherals:	30/36 I/Os, ADC/DAC, Hall sensor, Touch I/Os, Temperature sensor, etc.	
	Communication:	Wi-Fi, Bluetooth, onboard antenna; it supports STA/AP/STA + AP operation mode; I2C, UART, CAN2.0, SPI, I2S, and among others	
	built-in security:	IEEE 802.11 standard security features all supported, including WFA, WPA/WPA2, and WAPI; Secure boot; Flash encryption 1024-bit OTP, up to 768-bit for customers; Cryptographic hardware acceleration: AES, SHA-2, RSA, elliptic curve cryptography (ECC), random number generator (RNG).	
(8)	Development kit:	TTGO T-Call ESP32 with SIM800L GPRS Module [85]	This kit can publish data to a cloud IoT server with a Wi-Fi module or using GSM/GPRS module.
	CPU:	ESPRESSIF-ESP32 240MHz Xtensa® single-/dual-core 32-bit LX6 microprocessor	
	Frequency:	240 Mhz	
	Memory:	FLASH Memory: QSPI flash 4MB/PSRAM 8MB; SRAM: 520 KB SRAM	

Table 5. *Cont.*

S. No:	Development Kits (Significant Features of Kit, Not Processor)		Comments
	Peripherals:	LED PWM, TV PWM, I2S, IRGPIO, capacitor touch sensor, ADC, DACLNA pre-amplifier; SIM card: Only supports Nano SIM card;	
	Communication:	UAR, SPI, SDIO and I2C; Bluetooth, Wi-Fi and SIM800L GSM/GPRS. Communication distance: 300 m; Wi-Fi Mode: Station/SoftAP/SoftAP+Station/P2P.	
	Security:	WPA/WPA2/WPA2-Enterprise/WPS; Encryption Type: AES/RSA/ECC/SHA.	
(9)	Development kit:	TTGO LoRa32 SX1276 Board [85]	Using the Lora module, this device can transfer data to other devices over a long distance.
	CPU:	ESP32, SX1279; 32-bit LX6	
	Frequency:	240 Mhz	
	Memory:	540 KB SRAM, QSPI 4 MB Flash,	
	Peripherals:	Digital I/O, ADCs, DACs, PWM, etc.	
	Communication:	UART, SPI, SDIO, I2C, PWM, I2S, ADC, DAC, Cap Sensor; Wi-Fi (Station/SoftAP/SoftAP+Station/P2P), Bluetooth, Bluetooth Low Energy (BLE); LoRa RF range for Europe: 433 MHz, 868 MHz, Australia, and North America: 915 MHz, India: 865 MHz–867 MHz and Asia: 923 MHz; range: 300 m.	
	Security:	WPA/WPA2/WPA2-Enterprise/WPS; Encryption type: AES/RSA/ECC/SHA.	
(10)	Development kit:	ESP32 TTGO T-Beam V1.1 [85]	With Lora, it can send data up to 16Km (LoS). With GPS, it can track the geolocation of things.
	CPU:	ESP32, SX1279; 32-bit LX6	
	Frequency:	240 Mhz of ESP32	
	Memory:	4 MB flash memory, 8 MB PSRAM	
	Peripherals:	Digital I/Os, PWM, ADCs, DACs, Touch pins,	
	Communication:	TTGO Lora module, GPS modules NEO-6M, Wi-Fi, and Bluetooth LE via a “3D antenna” on the PCB., UART, C2I, SPI, etc.	
	Security:	As per ESP32 chip	
(11)	Development kit:	BBC Micro: bit V2 [114]	Specifically, developed for IoT
	CPU:	Nordic Semiconductor nRF52833 ARM Cortex-M4 32bit microcontroller	
	Frequency:	64 MHz	
	Memory:	128 KB SRAM and 512 KB Flash	
	Peripherals:	19 digital I/Os, 6 ADC pins, 19 DAC pins; onboard 5 × 5 multiplexed LEDs; MMA8653FC 3-axis, 10-bit digital accelerometer; MAG3110 3-axis magnetometer; Pushbuttons;	
	Communication:	I2C, 2.4 GHz radio antenna, as well as Bluetooth 5.1.	
	Security:	Not specified.	

Table 5. *Cont.*

S. No:	Development Kits (Significant Features of Kit, Not Processor)		Comments
(12)	Development kit:	Raspberry Pi 4 Model B—8 GB [87]	This kit is a complete 64bit Computer on the single board
	CPU:	A 1.5 GHz quad-core 64-bit ARM Cortex-A72 CPU	
	Frequency:	1.5 GHz	
	Memory:	8 GB SDRAM	
	Peripherals:	40 digital I/Os, 4Kp60 hardware decode of HEVC video, Video Core VI graphics, supporting OpenGL ES 3.x, Dual monitor support, at resolutions up to 4K,	
	Communication:	Dual-band 2.4 GHz and 5.0 GHz IEEE 802.11b/g/n/ac wireless; two USB 3.0 and two USB 2.0 ports; bluetooth 5.0; full-throughput gigabit ethernet; dual-band 802.11ac wireless networking.	
	Security:	Not specified.	

A large number of contemporary development kits, e.g., Arduino Uno R3, Arduino Mega2560 [86], are centered around the 8-bit processor (more technically, these processors are called microcontrollers). These kits are more suitable for the less complex computational designs; they cannot be expected to run any AI or ML algorithms because they work at the lower frequency, e.g., 16MHz (see Table 5). In addition, most of these kits have no Wi-Fi, Bluetooth, and GSM/GPRS modules onboard or on-chip to shift data directly to the target cloud server if required.

To cope with these limitations, such kits are usually equipped with a suitable module such as the ESP8266 transceiver, GSM/GPRS module, Zbee, Bluetooth, and others. Moreover, such kits offer less memory, prompting developers to use flash memory or SD Cards to store valuable data. Notably, most of these kits are best suited for many embedded applications such as automobiles and home appliances and are abundant in numerous peripherals, including ADC, DAC, timer/counters, touch pins, PWM pins, UART, I2C, and SPI, among others. Though 16-bit processor-based kits are relatively better than 8-bit kits, they are not suitable for modern designs that inherit complex algorithms such as AI and ML.

There is a strong trend of using 32-bit processor-based electronic development kits in the latest IoT design and development. In the context of current IoT requirements, these kits are usually equipped with numerous onboard communication modules such as Wi-Fi, Bluetooth, low energy (BLE), LoraWAN, Zbee, or GSM/GPRS. Using these modules, these kits can send environmental data directly to the cloud server via a local gateway or cellular network.

Figure 11c shows Arduino Due, which is centered around a 32-bit processor. Compared with Uno and Mega, this kit is extremely rich in peripherals such as digital I/O, PWM, ADC, DAC, UART, SPI, I2C, and memory, among others. In addition, it runs on 84MHz, which implies this device is suitable for the less complex computational AI/ML applications. This device has no module to send data directly to the internet, so it is equipped with an ESP8266 transceiver or a GSM/GPRS module. However, it is less attractive than its siblings (e.g., Arduino Nano 33 IoT (Figure 11d), now well-equipped with the needs and requirements of modern IoT designs.

The demand for IoT products increases exponentially due to its vast applications, e.g., smart homes, villages, and cities; medicine, agriculture, water monitoring, wearables, spying devices, vehicle tracking, and children monitoring. Numerous stakeholders are now competing, making their devices smarter, more affordable, reliable, and accessible to all customers. In this respect, the ESPRESSIF SYSTEMS (SHANGHAI) CO., LTD. [85] has launched many electronic devices and kits highly suitable for AI, ML, and IoT applications. For example, ESP8266 transceiver, NodeMCU (8266), NodeMCU (ESP32), ESP32SCAM, TTGO T-Call ESP32 with SIM800L GPRS Module, TTGO LoRa32 SX1276 Board, and ESP32 TTGO T-Beam V1.1 are just some of the products are based on the ESP devices.

As shown in Table 5, ESP devices are abundant in peripherals (e.g., ADC, DAC, UART, SPI, I2C, Touch pins, Wi-Fi, Bluetooth, BLE, timers, counters, and memory). The majority of the ESP chips have dual processor cores, which can operate at higher clock frequencies such as 80~240 MHz.

Most RESSIF technology-based products are well-suited for modern IoT, AI, and IoT applications. For example, the TTGO T-Call ESP32 with the SIM800L GPRS module has Wi-Fi, Bluetooth, BLE, and an onboard GSM/GPRS to send environmental data to cloud server either via Wi-Fi, Bluetooth, BLE, or cellular network using GSM/GPRS module (enabled with an active SIM). The TTGO LoRa32 SX1276 board can send data to other Lora-enabled devices over relatively long distances, which could not be achieved using a Wi-Fi router, Bluetooth, BLE, or Xbee modules. Similarly, the ESP32 TTGO T-Beam V1.1 board has an onboard GPS module, which could be utilized to track any target thing.

Most importantly, ESP32 based kits could be programmed wirelessly (i.e., over-the-air (OTA) programming). These kits can be used as clients or in the server mode supplying services to other similar devices. Moreover, the developers can easily make a network of these devices to share and collect data from the environment if necessary. Last but not least, the ESP products are highly optimized for energy consumption. Using software routines, developers can put these devices into the idle, sleeping, or deep sleeping mode whenever needed.

Recently, the BBC Micro:bit has gained enough considerations from hobbyists, developers, and scientists worldwide [114]. Similar to ESP products, this kit is also based on a 32-bit processor that runs at 64MHz. These kits are abundant in peripherals such as SRAM, Flash, I/Os, ADC, DAC, I2C, and MAG3110 3-axis magnetometer. No doubt, this silicon crab offers strong performance, but its relevant forum and communities seem limited compared with the Arduino products.

In addition to 32-bit devices, there are many 64-bit processor-based electronic development kits, such as Raspberry Pi 4 Model B [87]. No doubt, such boards are more suitable for the IoT, AI, or ML applications, where more computations and throughput are needed locally. Though Raspberry Pi kits are powerful enough, they are costly and require operating systems in the SD cards.

To conclude this section, the authors highly recommend 32-bit electronic development kits (equipped with Wi-Fi, Bluetooth, BLE, Xbee, Lora, or GSM/GPRS) for the design and development of IoT products.

5.2. Cloud Servers

In IoT designing, IoT cloud servers play a crucial role. Briefly, the main task of an IoT cloud server is to facilitate IoT developers in storing, analyzing, and visualizing sensors data. In addition, it may also provide complete control over the functionality of the IoT-enabled devices (e.g., water pumps and vehicles) from any corner of the globe. The following elaborates on this aspect of IoT designing.

- Commercial IoT cloud server: Accompanying the current boom in the IoT-enabled products worldwide is a plethora of contemporary IoT cloud servers [79], such as ThingSpeak, SensorCloud, Blynk, Arduino cloud IoT, IBM IoT, Adafruit io, and others. In general, each platform can process, analyze, and store data. Moreover, most cloud servers have soft widgets for the visualization of sensors data in different formats. For example, in ThingSpeak and Blynk, developers can more comfortably use readymade widgets such as the LCM, buttons, switches, panels for live streaming video data, live tracking of GPS data on Google map, and others. Moreover, in some platforms (e.g., ThingSpeak), IoT developers can also apply ML or AI techniques to data to predict hidden patterns and features.

- While commercially available IoT cloud servers are suitable for IoT designs, they have some limitations; thus, the IoT developers may feel reluctant to use them when developing IoT-enabled products. First, most of these platforms are not cost-free. While using limited resources, the developers may still need to pay the license fee for commercial activities. Second, any valuable data is also in the hands of a third party and may not be secured against hacking hazards. Third, any malfunction or seizure of such platforms may strongly affect the reputation and work of the IoT developers. Finally, there may also exist some unwanted delays in the communication. For technical comparison and explicit details, readers are advised to consult the survey of IoT cloud servers [79], which provides instructive details.
- Private IoT cloud server: In the context of hazards involved in using commercial IoT cloud servers, developers are highly encouraged to develop their own webservers if possible. Due to a large IoT community, developers can easily custom-design cloud servers for their products, thereby having easy access to relevant books, websites, and technical documents. For example, GitHub is the largest and highly advanced development platform globally [115]. Many companies and software developers build, ship, and maintain their software-related material using this platform. In addition, many IoT development IDEs, such as Arduino cloud IoT, provide sample codes for the webservers to develop electronic gadgets, such as Arduino Nano 33 IoT, ESP8266, and NodeMCU (8266), and other numerous well-known electronic gadgets.

5.3. Cyber Security

In general, the majority of the IoT-enabled embedded products (e.g., smart TV) are usually not secure against cyber assaults [116]. The Forescout Research Labs in 2020 published a comprehensive report on enterprise IoT security [117]. This report involved data from eight million IoT-enabled devices deployed across the government, financial services, manufacturing, healthcare, and retail sectors. All the information was gathered from the Device Cloud. The report indicates that: (i) medical devices, VoIP phones, networking equipment, and smart buildings are the riskiest IoT device groups. (ii) Around six of the top ten riskiest IoT device types fall into categories of networking equipment and medical devices. (iii) Windows-enabled workstations pose a major cybersecurity risk to corporations and organizations, where around 30% of managed devices in manufacturing and about 35% in healthcare run unsupported Windows versions.

To resist cybercriminals, the IoT gateways [118] need to be adapted for secure communication. These gateways perform critical tasks such as protocol translation, security, updating, device connectivity, management, data filtering, and processing. Since end-users use typical mobile Apps (e.g., Blynk) in many IoT applications to monitor and control IoT, security measures should also be considered in this context. Notably, using loose passwords and PINs (personal identification numbers) are weak shields against cyber attackers. Besides, some manufacturers of IoT gadgets also incorporate security measures at the hardware layer, which is invoked whenever secure communication is required [85].

5.4. Significant Limitations

As highlighted in Table 6, the IoT application for smart water tank monitoring faces numerous serious limitations [119–126]. The following briefly describes some of these challenges.

Table 6. Significant limitations and challenges of IoT-controlled water tanks.

S. No.	Limitations/Challenges	Recommendations
1.	Development of biofilms on sensors	Search market for reliable products
2.	Most IoT gadgets do not have built-in security	Search for IoT kits that are equipped with the latest security threats or embedded chips
3.	Most contemporary IoT cloud servers are not free of cost	It is best to develop a private IoT cloud server if possible
4.	Public IoT cloud servers are not considered secure due to third party involvement	It is best to develop a private IoT cloud server if possible
5.	Implementation of AI/ML schemes on IoT gadgets	Select at least 32-bit IoT devices with higher clock beats
6.	Overall system cost	Cost should be minimized to make IoT products accessible to all consumers
7.	Developing secure Apps for smartphone	It is wise to develop a private App if possible

- False Access Control: It is observed that all IoT-related devices having the same model are generally delivered to end customers with the same default password, such as ‘admin′ or ‘admin123′. In addition, the default settings and firmware are often the same for all devices of the same model. It is also observed that most end-users do not even change these default passwords and settings, thus putting the IoT-enabled systems at risk. In addition, most IoT devices do not have deep layered security. In general, end-user can access these devices using an account or a password, neither of which are enough against seasonal cybercriminals.
- Obsolete Software: Most IoT vendors generally do not provide updates to protect IoT-enabled products against cyberattacks. Besides, end-users also ignore updating the IoT-enabled products, which may cause a system breakdown and consequently monetary loss. To avoid these issues, IoT devices must be shipped with up-to-date firmware/software without any known weaknesses. In addition, there should exist update functionality for end-users or developers to fix any vulnerabilities found after device deployment.
- Deficiency of Encryption: Whenever an IoT device performs communication in the plain text format, all valuable information exchanged with a target device or backend service could be acquired by a middleman. So, any person capable of obtaining a position on the network path between the IoT device and the terminal node may easily examine network traffic and obtain any sensitive data, e.g., login credentials. Moreover, if encryption is not done completely or is misconfigured, the attackers may gain access to IoT-enabled systems. In addition, encryption should also protect the sensitive data stored on a device (at rest). In this concern, usual flaws may also be the lack of encryption by credentials or storing API tokens in plain text on IoT devices. Other complications may be the usage of poor cryptographic techniques.
- Intrusion Ignorance: After being compromised, the IoT devices often keep working as usual from the user′s viewpoint. Notably, any power or additional bandwidth usage is generally not identified. It is also observed that most IoT devices do not inherit any alerting or logging functionality to inform end-users of security-related issues. In case IoT devices have these functionalities, the concerned hacker can overwrite or disable them when IoT devices are hacked; the end-user cannot take preventive measures to mitigate the effects of cyberattacks.
- Application Weaknesses: To secure IoT devices, it is important to acknowledge vulnerabilities, if any, in the software in the first place. It is observed that software bugs

can trigger malicious activity. Cybercriminals can run their own software on IoT devices and gain illegal access to the sensitive information stored on the concerned devices. Though avoiding software vulnerabilities altogether may not be possible, the developers can adopt best programming practices to escape application vulnerabilities, e.g., performing input validation consistently.

- Vendor's Security Stance: Whenever software vulnerabilities are detected, it is for the sake of utmost reliability that the concerned vendor finds a proper patch to mitigate their effects. In this concern, the IoT vendors should offer their contact information so that end-users and developers can communicate if any bugs or security loopholes are found. Otherwise, the end-users and developers would not cease using IoT devices in the intended method, resulting in less secure IoT systems. To avoid any catastrophic situation, concerned vendors must cooperate with the end-users and developers, providing frequent updates on the security of the IoT devices and recommendations on how to securely resell or dispose of IoT devices so that the sensitive data is not passed on.
- Deficiency of Reliable Execution Environment: Every IoT device is generally equipped with a dedicated microcontroller, which is a true computer on a single chip capable of running specific software programs. The cyber attacker may install any malicious programs. For instance, they can install a software routine performing a DDoS attack. While limiting the functionality of an IoT device and the software it can run, the potential for abuse of the device is limited. For instance, the concerned IoT device may be confined to connecting only to the vendor's cloud service. No doubt, this act of restriction may make it unproductive in a DDoS attack because such a device can no longer be able to connect target hosts arbitrarily.

The code that is typically signed with a cryptographic hash should be used to address this issue. As only the concerned vendor has the key to sign software, the IoT device will only run the vendor distributed software. The cybercriminal may no longer run their arbitrary code on the target device.

- **Inadequate Physical Security:** Cyber attackers may open the IoT devices and can attack the hardware if they have access to these devices. For instance, hackers can bypass protective software if they can directly read the contents of memory components. After opening up IoT devices, hackers may read the device debugging contacts and perform more fatal tasks. The physical attack may have more impact if it uncovered the device key shared with all IoT devices of the same model; this act would compromise many IoT devices.
- **User Interaction**

IoT vendors should provide proper documentation and guidelines on deploying their products more securely in IoT-based systems. Users must change the default passwords to help prevent hackers from gaining access to their devices.

6. Conclusions

Over the past several decades, water resources have been gradually decreasing, becoming scarce in several places worldwide. To cope with this issue, different nations worldwide are now taking serious measures to mitigate the effects of the water crisis. In this respect, smart monitoring of water resources has gained tremendous attention within the research communities.

In advanced developed countries (e.g., USA, UAE, UK, and Singapore), mega organizations, corporations, and municipalities currently use mature, smart water monitoring systems, such as Libelium products. In general, such commercially available products are capable of monitoring water quality, detecting water leakage, and generating utility bills, among other things. No doubt, such products are well-suited for smart monitoring of fresh water. However, they are also expensive and, thus, not easily accessible by individuals living in developing countries. To mitigate the effects of the water crisis, the relevant communities in developing countries are involved in devising affordable and reliable internet-of-things (IoT) based solutions to monitor water resources.

In this context, the authors of this study have gathered helpful information about smart monitoring of water storage tanks, often used for water storage in rural and urban areas. The authors provided an overview of the water monitoring techniques, from traditional ways through the wireless sensors network (WSN) to IoT-based monitoring (smart monitoring). Next, they conducted a critical survey of the contemporary IoT-based monitoring of water storage tanks (IoT-WST). Here, each IoT-WST was critically analyzed along with its potential pros and cons. The authors also elaborated on recent technologies and techniques in the IoT-WST. Within this framework, they considered monitoring water level, water leakage, and auto refill of water storage tanks. Finally, the authors explicitly detail a number of current challenges and trends in IoT-relevant hardware, cloud servers, and cyber security. The discussion provided here offers a valuable contribution to the ongoing efforts to address and mitigate the effects of the water crisis.

For future work and directions, the authors suggest relevant research communities to enhance IoT cybersecurity, mitigate the effects of biofilms, introduce artificial intelligence and machine learning techniques, and optimize the overall system cost.

Author Contributions: Conceptualization, F.J.; resources, F.J.; writing—original draft preparation, F.J.; writing—review and editing, S.S. and S.Z.I.; proofreading—language review and editing, R.A.; funding acquisition, N.M.-A. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Deanship of Scientific Research (DSR), Imam Abdulrahman Bin Faisal University (IAU), grant number 2019-381-CSIT. The APC was funded by the Deanship of Scientific Research (DSR), Imam Abdulrahman Bin Faisal University (IAU).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: Authors are thankful to the Deanship of Scientific Research (DSR), Imam Abdulrahman Bin Faisal University (IAU), for the generous funding under project numbered 2019-381-CSIT. Moreover, we are grateful to authors, companies, and organizations whose material has been utilized in the preparation of this article.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Zhu, G.; Zhao, G.; Zhang, Z.; Lu, X. Water quality of water source area in Taihu Lake and effect on water treatment process. In Proceedings of the 2011 International Conference on Consumer Electronics, Communications and Networks (CECNet), Xianning, China, 16–18 April 2011; pp. 3783–3786.
2. Zamberlan da Silva, M.E.; Santana, R.G.; Guilhermetti, M.; Filho, I.C.; Endo, E.H.; Ueda-Nakamura, T.; Nakamura, C.V.; Dias Filho, B.P. Comparison of the bacteriological quality of tap water and bottled mineral water. *Int. J. Hyg. Environ. Health* **2008**, *211*, 504–509. [CrossRef]
3. Yauri, R.; Rios, M.; Lezama, J. Water quality monitoring of Peruvian Amazon based in the Internet of Things. In Proceedings of the 2017 IEEE XXIV International Conference on Electronics, Electrical Engineering and Computing (INTERCON), Cusco, Peru, 15–18 August 2017; pp. 1–4.
4. Yasin, S.; Mohd Yunus, M.F.; Abdul Wahab, N. The development of water quality monitoring system using internet of things. *J. Educ. Learn. Stud.* **2020**, *3*, 14. [CrossRef]
5. Yaroshenko, I.; Kirsanov, D.; Marjanovic, M.; Lieberzeit, P.A.; Korostynska, O.; Mason, A.; Frau, I.; Legin, A. Real-Time Water Quality Monitoring with Chemical Sensors. *Sensors* **2020**, *20*, 3432. [CrossRef]
6. Xin, X.-K.; Li, K.-F.; Finlayson, B.; Yin, W. Evaluation, prediction, and protection of water quality in Danjiangkou Reservoir, China. *Water Sci. Eng.* **2015**, *8*, 30–39. [CrossRef]
7. WQI. Available online: https://www.un.org/waterforlifedecade/pdf/global_drinking_water_quality_index.pdf (accessed on 3 June 2021).
8. Wang, Z.; Wang, Q.; Hao, X. The Design of the Remote Water Quality Monitoring System Based on WSN. In Proceedings of the 2009 5th International Conference on Wireless Communications, Networking and Mobile Computing, Beijing, China, 24–26 September 2009; pp. 1–4.

9. Vinod, G.; Peter, A.V.; Rao, I.S.; Sailaja, S.; Babu, Y.S. IoT based Water Quality Monitoring System Using WSN. *Indian J. Public Health Res. Dev.* **2018**, *9*, 1575. [CrossRef]
10. Urs, A.C.; Shubha, J.; Sushmitha, P.B.; Vaishnavi, A.P. Design of Smart Sensors for Real-Time Water Quality Monitoring Using IOT Technology. *Int. J. Sci. Dev. Res.* **2017**, *2*, 348–350.
11. Tsihrintzis, V.A. Book Review Marcello Benedini and George Tsakiris. Water Quality Modelling for Rivers and Streams, Springer, Water Science and Technology Library Series, Volume 70, 288p, ISBN 978-94-007-5508-6. *Water Resour. Manag.* **2013**, *27*, 5299–5302.
12. Thiyagarajan, K.; Pappu, S.; Vudatha, P.; Niharika, A.V. Intelligent IoT Based Water Quality Monitoring System. *Int. J. Appl. Eng. Res.* **2017**, *12*, 5447.
13. Suryawanshi, V.; Khandekar, M. Design and Development of Wireless Sensor Network (WSN) for Water Quality Monitoring Using Zigbee. In Proceedings of the 2018 Second International Conference on Intelligent Computing and Control Systems (ICICCS), Madurai, India, 14–15 June 2018; pp. 862–865.
14. Spandana, K.; Rao, V. Internet of Things (Iot) Based Smart Water Quality Monitoring System. *Int. J. Eng. Technol.* **2018**, *7*, 259–262. [CrossRef]
15. Rahman, S.; Hossain, F. Spatial Assessment of Water Quality in Peripheral Rivers of Dhaka City for Optimal Relocation of Water Intake Point. *Water Resour. Manag.* **2008**, *22*, 377–391. [CrossRef]
16. Pule, M.; Ahadi, Y.; Chuma, J. Wireless sensor networks: A survey on monitoring water quality. *J. Appl. Res. Technol.* **2018**, *15*, 562–570. [CrossRef]
17. Priya, S.K.; Shenbaga, L.G.; Revathi, T. Design of smart sensors for real time drinking water quality monitoring and contamination detection in water distributed mains. *Int. J. Eng. Technol.* **2017**, *7*, 47. [CrossRef]
18. Prasad, A.N.; Mamun, K.A.; Islam, F.R.; Haqva, H. Smart water quality monitoring system. In Proceedings of the 2015 2nd Asia-Pacific World Congress on Computer Science and Engineering (APWC on CSE), Nadi, Fiji, 2–4 December 2015; pp. 1–6.
19. Pranata, A.; Lee, J.M.; Kim, D.-S. Towards an IoT-based water quality monitoring system with brokerless pub/sub architecture. In Proceedings of the 2017 IEEE International Symposium on Local and Metropolitan Area Networks (LANMAN), Osaka, Japan, 12–14 June 2017.
20. Peletz, R.; Kisiangani, J.; Bonham, M.; Ronoh, P.; Delaire, C.; Kumpel, E.; Marks, S.; Khush, R. Why do water quality monitoring programs succeed or fail? A qualitative comparative analysis of regulated testing systems in sub-Saharan Africa. *Int. J. Hyg. Environ. Health* **2018**, *221*, 907–920. [CrossRef] [PubMed]
21. Pasika, S.; Gandla, S.T. Smart water quality monitoring system with cost-effective using IoT. *Heliyon* **2020**, *6*, e04096. [CrossRef]
22. Niswar, M.; Wainalang, S.; Ilham, A.; Zainuddin, Z.; Fujaya, Y.; Muslimin, Z.; Paundu, A.; Kashihara, S.; Fall, D. IoT-based Water Quality Monitoring System for Soft-Shell Crab Farming. In Proceedings of the 2018 IEEE International Conference on Internet of Things and Intelligence System (IOTAIS), Bali, Indonesia, 1–3 November 2018.
23. Myint, C.Z.; Gopal, L.; Aung, Y.L. WSN-based reconfigurable water quality monitoring system in IoT environment. In Proceedings of the 2017 14th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON), Phuket, Thailand, 27–30 June 2017; pp. 741–744.
24. Moparthi, N.R.; Mukesh, C.; Sagar, P.V. Water Quality Monitoring System Using IOT. In Proceedings of the 2018 Fourth International Conference on Advances in Electrical, Electronics, Information, Communication and Bio-Informatics (AEEICB), Chennai, India, 27–28 February 2018; pp. 1–5.
25. Menon Kau, D.P.; Ramesh, M.V. Wireless sensor network for river water quality monitoring in India. In Proceedings of the 2012 Third International Conference on Computing, Communication and Networking Technologies (ICCCNT'12), Coimbatore, India, 26–28 July 2012; pp. 1–7.
26. Manoharan, S.; Sathi, Y.G.; Thiruven, K.K.; Vetriselvan, G.V.; Kishor, P. Water Quality Analyzer using IoT. *Int. J. Innov. Technol. Explor. Eng.* **2019**, *8*, 34–37.
27. Madhavireddy, V.; Koteswarrao, B. Smart Water Quality Monitoring System Using Iot Technology. *Int. J. Eng. Technol.* **2018**, *7*, 636–639. [CrossRef]
28. Loganathan, G.B.; Mohan, E.; Kumar, R.S. IoT based water and soil quality monitoring system. *Int. J. Mech. Eng. Technol.* **2019**, *10*, 537–541.
29. Viola, A.; Curto, D.; Franzitta, V.; Trapanese, M. Sea water desalination and energy consumption: A case study of wave energy converters (WEC) to desalination applications in sicily. In Proceedings of the OCEANS 2016 MTS/IEEE Monterey, Monterey, CA, USA, 19–23 September 2016; pp. 1–5.
30. Lin, Z.; Duo, W.; Gao, C.; Gao, Z. Impacts of seawater desalination on environment. In Proceedings of the 2011 Second International Conference on Mechanic Automation and Control Engineering, Hohhot, China, 15–17 July 2011; pp. 2228–2233.
31. Min-Allah, N.; Farooqui, M.; Alwashmi, A.; Almahasheer, S.; Alsufayyan, M.; Altulaihan, N. Smart Monitoring of Water Tanks in KSA. In Proceedings of the 2018 International Conference on Computational Science and Computational Intelligence (CSCI), Las Vegas, NV, USA, 12–14 December 2018; pp. 1044–1047.
32. Nasirudin, M.A.; Za'bah, U.N.; Sidek, O. Fresh water real-time monitoring system based on Wireless Sensor Network and GSM. In Proceedings of the 2011 IEEE Conference on Open Systems, Langkawi, Malaysia, 25–28 September 2011; pp. 354–357.
33. Suwaileh, W.; Johnson, D.; Hilal, N. Membrane desalination and water re-use for agriculture: State of the art and future outlook. *Desalination* **2020**, *491*, 114559. [CrossRef]

34. Shankar, S.; Dakshayini, M. IoT-Mobile Enabled Smart Water Level Controlling System to Regulate Water Wastage. In Proceedings of the 2018 International Conference on Advances in Computing, Communications and Informatics (ICACCI), Bangalore, India, 19–22 September 2018; pp. 2045–2048.
35. Adu-Manu, K.; Tapparello, C.; Heinzelman, W.; Katsriku, F.; Abdulai, J.-D. Water Quality Monitoring Using Wireless Sensor Networks: Current Trends and Future Research Directions. *ACM Trans. Sens. Netw.* **2017**, *13*, 1–41. [CrossRef]
36. The Nature Conservancy. Available online: https://www.nature.org/en-us/what-we-do/our-priorities/provide-food-and-water-sustainably/ (accessed on 30 October 2021).
37. Narayanan, L.K.; Sankaranarayanan, S. IoT-based water demand forecasting and distribution design for smart city. *J. Water Clim. Chang.* **2019**, *11*, 1411–1428. [CrossRef]
38. Nacht, S. Ground-water monitoring system considerations. *Ground Water Monit. Remediat.* **2007**, *3*, 33–39. [CrossRef]
39. Mohebbi, S.; Zhang, Q.; Christian Wells, E.; Zhao, T.; Nguyen, H.; Li, M.; Abdel-Mottaleb, N.; Uddin, S.; Lu, Q.; Wakhungu, M.J.; et al. Cyber-physical-social interdependencies and organizational resilience: A review of water, transportation, and cyber infrastructure systems and processes. *Sustain. Cities Soc.* **2020**, *62*, 102327. [CrossRef]
40. Mohan, S. Novel Water Monitoring System: A Python Based Aqua Monitoring System using Raspberry PI. *Int. J. Res. Appl. Sci. Eng. Technol.* **2019**, *7*, 1648–1651.
41. Li, Y.; Wang, Y.; Detai, Z.; Yin, J.; Su, J. HIRFL cooling water-monitoring system design and construct. *Radiat. Detect. Technol. Methods* **2018**, *2*, 35. [CrossRef]
42. Lambrou, T.P.; Anastasiou, C.C.; Panayiotou, C.G.; Polycarpou, M.M. A Low-Cost Sensor Network for Real-Time Monitoring and Contamination Detection in Drinking Water Distribution Systems. *IEEE Sens. J.* **2014**, *14*, 2765–2772. [CrossRef]
43. Khatri, P.; Gupta, K.K.; Gupta, R.K. Assessment of Water Quality Parameters in Real-Time Environment. *SN Comput. Sci.* **2020**, *1*, 340. [CrossRef]
44. Joustra, C.M.; Yeh, D.H. Demand- and source-driven prioritization framework toward integrated building water management (IBWM). *Sustain. Cities Soc.* **2015**, *14*, 114–125. [CrossRef]
45. Han, F.; Zheng, Y. Joint analysis of input and parametric uncertainties in watershed water quality modeling: A formal Bayesian approach. *Adv. Water Resour.* **2018**, *116*, 77–94. [CrossRef]
46. Giudicianni, C.; Herrera, M.; di Nardo, A.; Carravetta, A.; Ramos, H.M.; Adeyeye, K. Zero-net energy management for the monitoring and control of dynamically-partitioned smart water systems. *J. Clean. Prod.* **2020**, *252*, 119745. [CrossRef]
47. Geetha, S.; Gouthami, S. Internet of things enabled real time water quality monitoring system. *Smart Water* **2017**, *2*, 1. [CrossRef]
48. Franceschini, F.; Galetto, M.; Turina, E. Water and Sewage Service Quality: A Proposal of a New Multi-Questionnaire Monitoring Tool. *Water Resour. Manag.* **2010**, *24*, 3033–3050. [CrossRef]
49. Demetillo, A.T.; Japitana, M.V.; Taboada, E.B. A system for monitoring water quality in a large aquatic area using wireless sensor network technology. *Sustain. Environ. Res.* **2019**, *29*, 12. [CrossRef]
50. Water_Monitoring_Day: EarthEcho Water Challenge. Available online: https://wwwmonitorwaterorg/ (accessed on 10 September 2021).
51. Daadoo, M.; Daraghmi, Y.-A. Smart Water Leakage Detection Using Wireless Sensor Networks (SWLD). *Int. J. Netw. Commun.* **2017**, *2017*, 1–16.
52. Berman, J. WHO: Waterborne Disease is World's Leading Killer | Voice of America—English. Available online: https://www.voanews.com/archive/who-waterborne-disease-worlds-leading-killer (accessed on 19 May 2021).
53. Drinking-Water. Available online: https://www.who.int/news-room/fact-sheets/detail/drinking-water (accessed on 10 September 2021).
54. WHO: Water Borne Diseases. Available online: https://wwwwhoint/news-room/fact-sheets/detail/drinking-water (accessed on 10 September 2021).
55. Gleick, P. Dirty Water: Estimated Deaths from Water-Related Diseases 2000–2020. Available online: https://pacinst.org/publication/569/ (accessed on 10 September 2021).
56. Water_in_Crisis: WATER IN CRISIS/ES? 2020. Available online: https://wwwsolidaritesorg/wp-content/uploads/2020/03/solidarites_2020_water-hygiene-barometerpdf (accessed on 10 September 2021).
57. World_Water_Council: Water Crisis: Towards a Way to Improve the Situation. 2020. Available online: https://wwwworldwatercouncilorg/en/water-crisis (accessed on 10 September 2021).
58. Water_Borne_Diseases. Available online: https://www.voanews.com/archive/who-waterborne-disease-worlds-leading-killer (accessed on 10 September 2021).
59. Ahmed, U.; Mumtaz, R.; Anwar, H.; Mumtaz, S.; Qamar, A.M. Water quality monitoring: From conventional to emerging technologies. *Water Supply* **2019**, *20*, 28–45. [CrossRef]
60. Banna, M.H.; Imran, S.; Francisque, A.; Najjaran, H.; Sadiq, R.; Rodriguez, M.; Hoorfar, M. Online Drinking Water Quality Monitoring: Review on Available and Emerging Technologies. *Crit. Rev. Environ. Sci. Technol.* **2014**, *44*, 1370–1421. [CrossRef]
61. Klepka, A.; Broda, D.; Michalik, J.; Kulbat, M.; Małka, P.; Staszewski, W.; Stepinski, T. Leakage Detection in Pipelines—The Concept of Smart Water Supply System. In Proceedings of the 7th ECCOMAS Thematic Conference on Smart Structures and Materials, Ponta Delgada, Azores, Portugal, 3–6 June 2015.

62. Gama-Moreno, L.A.; Corralejo, A.; Ramirez-Molina, A.; Rangel, J.A.T.; Martinez-Hernandez, C.; Juarez, M.A. A Design of a Water Tanks Monitoring System Based on Mobile Devices. In Proceedings of the 2016 International Conference on Mechatronics, Electronics and Automotive Engineering (ICMEAE), Cuernavaca, Mexico, 22–25 November 2016; pp. 133–138.
63. Rao, K.R.; Srinija, S.; Bindu, K.; Kumar, D. IOT based water level and quality monitoring system in overhead tanks. *Int. J. Eng. Technol.* **2018**, *7*, 379.
64. Redwan, F.; Rafid, S.; Abrar, A.H.; Pathik, B.B. An Exploratory Approach to Monitor the Quality of Supply-Water Through IoT Technology. In Proceedings of the 2019 International Conference on Automation, Computational and Technology Management (ICACTM), London, UK, 24–26 April 2019; pp. 137–142.
65. Gunde, S.; Chikaraddi, A.K.; Baligar, V.P. IoT based flow control system using Raspberry PI. In Proceedings of the 2017 International Conference on Energy, Communication, Data Analytics and Soft Computing (ICECDS), Chennai, India, 1–2 August 2017; pp. 1386–1390.
66. Li, J.; Yang, X.; Sitzenfrei, R. Rethinking the Framework of Smart Water System: A Review. *Water* **2020**, *12*, 412. [CrossRef]
67. Yuvaraj, T.; Krishna, N.; Manish, P.; Naik, P.; Varsha, P. Review Paper on Water Monitoring and Leakage Detection. *Int. J. Res. Sci. Innov. (IJRSI)* **2019**, *4*, 31.
68. Ali, H.; Choi, J.-H. A Review of Underground Pipeline Leakage and Sinkhole Monitoring Methods Based on Wireless Sensor Networking. *Sustainability* **2019**, *11*, 4007. [CrossRef]
69. Damor, P.R.; Sharma, K.J. IoT based Water Monitoring System: A Review. *Int. J. Adv. Eng. Res. Dev.* **2017**, *4*, 1–6.
70. Sheltami, T.R.; Bala, A.; Shakshuki, E.M. Wireless sensor networks for leak detection in pipelines: A survey. *J. Ambient Intell. Humaniz. Comput.* **2016**, *7*, 347–356. [CrossRef]
71. Public Utilities Board, S. Managing the water distribution network with a Smart Water Grid. *Smart Water* **2016**, *1*, 4. [CrossRef]
72. Jan, F.; Min-Allah, N.; Düştegör, D. IoT Based Smart Water Quality Monitoring: Recent Techniques, Trends and Challenges for Domestic Applications. *Water* **2021**, *13*, 1729. [CrossRef]
73. Cheng, W.; Cheng, R.; Chou, S. Power-saving for IoT-enabled Water Dispenser System. In Proceedings of the 2019 42nd International Conference on Telecommunications and Signal Processing (TSP), Budapest, Hungary, 1–3 July 2019; pp. 736–739.
74. Val Ledesma, J.; Wisniewski, R.; Kallesøe, C.S. Smart Water Infrastructures Laboratory: Reconfigurable Test-Beds for Research in Water Infrastructures Management. *Water* **2021**, *13*, 1875. [CrossRef]
75. Wiki_Pedia. Available online: https://en.wikipedia.org/ (accessed on 10 September 2021).
76. Pujar, P.; Kenchannavar, H.; Kulkarni, U.P. Wireless Sensor Network based Water Monitoring Systems: A survey. In Proceedings of the 2016 2nd International Conference on Applied and Theoretical Computing and Communication Technology (iCATccT), Bangalore, India, 21–23 July 2016.
77. Chen, Y.; Hou, G.; Ou, J. WSN-based monitoring system for factory aquaculture. In Proceedings of the 2014 IEEE 5th International Conference on Software Engineering and Service Science, Beijing, China, 27–29 June 2014; pp. 439–442.
78. Morillo, P.; Orduña, J.M.; Fernández, M.; García-Pereira, I. Comparison of WSN and IoT approaches for a real-time monitoring system of meal distribution trolleys: A case study. *Future Gener. Comput. Syst.* **2018**, *87*, 242–250. [CrossRef]
79. Ray, P.P. A survey of IoT cloud platforms. *Future Comput. Inform. J.* **2016**, *1*, 35–46. [CrossRef]
80. Gopalakrishnan, P.; Abhishek, S.; Ranjith, R.; Venkatesh, R.; Jai, S.V. Smart Pipeline Water Leakage Detection System. *Int. J. Appl. Eng. Res.* **2017**, *12*, 5559–5564.
81. Vinoj, J.; Gavaskar, S. Smart City Underground Water Leak and Theft Detection Systemwith IOT. *Int. J. Sci. Res. Comput. Sci. Eng. Inf. Technol.* **2018**, *3*, 632–640.
82. Singh, R.; Baz, M.; Narayana, C.L.; Rashid, M.; Gehlot, A.; Akram, S.V.; Alshamrani, S.S.; Prashar, D.; AlGhamdi, A.S. Zigbee and Long-Range Architecture Based Monitoring System for Oil Pipeline Monitoring with the Internet of Things. *Sustainability* **2021**, *13*, 10226. [CrossRef]
83. Singh, R.; Baz, M.; Gehlot, A.; Rashid, M.; Khurana, M.; Akram, S.V.; Alshamrani, S.S.; AlGhamdi, A.S. Water Quality Monitoring and Management of Building Water Tank Using Industrial Internet of Things. *Sustainability* **2021**, *13*, 8452. [CrossRef]
84. Liu, P.; Wang, J.; Sangaiah, A.K.; Xie, Y.; Yin, X. Analysis and Prediction of Water Quality Using LSTM Deep Neural Networks in IoT Environment. *Sustainability* **2019**, *11*, 2058. [CrossRef]
85. Esspressif_Website. Available online: https://www.espressif.com/en/products/socs/ (accessed on 10 September 2021).
86. Arduino_WebSite. Available online: https://www.arduino.cc/en/Main/Products (accessed on 10 September 2021).
87. Raspberrypi_Website. Available online: https://www.raspberrypi.org/products/ (accessed on 13 September 2021).
88. Page, M.J.; McKenzie, J.E.; Bossuyt, P.M.; Boutron, I.; Hoffmann, T.C.; Mulrow, C.D.; Shamseer, L.; Tetzlaff, J.M.; Akl, E.A.; Brennan, S.E.; et al. The PRISMA 2020 statement: An updated guideline for reporting systematic reviews. *BMJ* **2021**, *372*, n71. [CrossRef]
89. Haddaway, N.; McGuinness, L.; Pritchard, C. PRISMA2020, R package and ShinyApp for producing PRISMA 2020 compliant flow diagrams. *Zenodo* **2021**. [CrossRef]
90. Pan, Y.; Ge, X.; Fang, C.; Fan, Y. A Systematic Literature Review of Android Malware Detection Using Static Analysis. *IEEE Access* **2020**, *8*, 116363–116379. [CrossRef]
91. Sohan, M.F.; Basalamah, A. A Systematic Literature Review and Quality Analysis of Javascript Malware Detection. *IEEE Access* **2020**, *8*, 190539–190552. [CrossRef]
92. Kumar, S.; Yadav, S.; Yashaswini, H.M.; Salvi, S. *An IoT-Based Smart Water Microgrid and Smart Water Tank Management System*; Springer: Singapore, 2019; pp. 417–431.

93. Nikeeta, P.K.; Kiranmayie, N.; Manishankar, P.; Nitej, C.K.; Murthy, V. Water Management System Using IOT. *Int. J. Comput. Trends Technol.* **2020**, *68*, 244–247.
94. Lade, S.; Vyas, P.; Walavalkar, V.; Wankar, B.; Yadav, P. Water Management System Using IoT with WSN. *Int. Res. J. Eng. Technol.* **2018**, *5*, 3079–3082.
95. Lakshmi, P.; Mounika, V.; Sri, V.; Pragna; Vikas, K. Smart Water Tank: An IoT based Android Application. *Iconic Res. Eng. J.* **2018**, *1*, 162–165.
96. Parimala, S.; Meesala, S.; Jyothi, N.A.; Dash, A. Monitoring the Water Storage Facilities using Internet of Things. *Int. J. Civ. Eng. Technol.* **2018**, *9*, 1507–1516.
97. Durga, S.; Ramakrishna, M.; Dayanandam, G. Autonomous Water tank Filling System using IoT. *Int. J. Comput. Sci. Eng.* **2018**, *6*, 1–4. [CrossRef]
98. Jaiad, A.T.; Ghayyib, H.S. Controlling and monitoring of automation water supply system based on IoT with theft identification. *Int. J. Res.-GRANTHAALAYAH* **2017**, *5*, 320–325. [CrossRef]
99. Sowmya, K.P.S.; Susmitha, K.; Sai, N.A.; Suma, N.; Sivaiah, N. Internet of Things (IoT) Enabled Water Monitoring SYSTEM. *Iconic Res. Eng. J.* **2018**, *1*, 40–43.
100. Pawaskar, M.C.; Gadikar, P.; Kambli, P.; Pawar, S.; Savardekar, S. GSM Based Water Control and Management. *Int. Res. J. Eng. Technol.* **2017**, *4*, 1927–1931.
101. Gupta, N.; Sasi, A.; Deep, A. IoT based Water Level Management System. *J. Xidian Univ.* **2020**, *14*, 622–633. [CrossRef]
102. Dissanayaka, R.M.S.M.; Wickramaarachchi, H. IoT Based Water Level Monitoring System Using NodeMCU. In Proceedings of the 11th Symposium on Applied Science, Business & Industrial Research, Kuliyapitiya, Sri Lanka, 31 March 2019; ISSN 2279-1558. ISBN 978-955-7442-27-3.
103. Natividad, J.; Palaoag, T. IoT based model for monitoring and controlling water distribution. *IOP Conf. Ser. Mater. Sci. Eng.* **2019**, *482*, 012045. [CrossRef]
104. Shah, P.P.; Patil, A.A.; Ingleshwar, S.S. IoT based smart water tank with Android application. In Proceedings of the 2017 International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC), Palladam, India, 10–11 February 2017; pp. 600–603.
105. Malche, T.; Maheshwary, P. Internet of Things (IoT) Based Water Level Monitoring System for Smart Village. In Proceedings of the International Conference on Communication and Networks, Ahmedabad, India, 19–20 February 2016.
106. Wadekar, S.; Vakare, V.; Prajapati, R.; Yadav, S.; Yadav, V. Smart water management using IOT. In Proceedings of the 2016 5th International Conference on Wireless Networks and Embedded Systems (WECON), Rajpura, India, 14–16 October 2016; pp. 1–4.
107. Charles, A.; Ayylusamy, C.; Raj, J.D.; Kumar, K.; Thiruvarasan, K.E. IOT Based Water Level Monitoring System Using Labview. *Int. J. Pure Appl. Math.* **2018**, *118*, 9–14.
108. Asif, S.S.; Doddaguni, S.R.; Kamagond, T.G.; Anala, M.R.; Mamatha, T. Household Water Resource Monitoring and Leakage Detection. *Int. J. Soft Comput. Eng.* **2020**, *9*, 12–17.
109. MIT_IoT: MIT App Inventor + IoT. Available online: http://iotappinventormitedu/#/LearnMore (accessed on 10 September 2021).
110. Saraswathi, V.; Rohit, A.; Sakthivel, S.; Sandheep, T.J. Water Leakage System Using IoT. *Int. J. Innov. Res. Eng. Manag.* **2018**, *5*, 67–69.
111. Li, K.; Jia, H.; Liu, M. ZigBee Wireless Sensor Network Using Physics-Based Optimally Sampling for Soil Moisture Measurement. In Proceedings of the 2011 Third International Conference on Communications and Mobile Computing, Qingdao, China, 18–20 April 2011; pp. 505–508.
112. Srivastava, S.; Vaddadi, S.; Sadistap, S. Smartphone-based System for water quality analysis. *Appl. Water Sci.* **2018**, *8*, 130. [CrossRef]
113. Alldatasheets. Available online: https://www.alldatasheet.com/ (accessed on 10 September 2021).
114. Micro_Bit_Website. Available online: https://microbit.org/ (accessed on 11 September 2021).
115. GitHub_Website. Available online: https://github.com/ (accessed on 13 September 2021).
116. Lee, I. Internet of Things (IoT) Cybersecurity: Literature Review and IoT Cyber Risk Management. *Future Internet* **2020**, *12*, 157. [CrossRef]
117. Forescout_Research_Labs: The Enterprise of Things Security Report: The State of IoT Security in 2020. Available online: https://www.forescout.com/the-enterprise-of-things-security-report-state-of-iot-security-in-2020/?utm_source=google&utm_medium=ppc&utm_campaign=iot_emea&gclid=CjwKCAjw7fuJBhBdEiwA2lLMYUYZ20MstFI0kRatlqyWTK888L49A488OMZYQlO-fGE8Z85MrwXdeRoCGA4QAvD_BwE (accessed on 13 September 2021).
118. IoT_Routers_and_Gateways: Cisco IoT Gateways. Available online: https://www.cisco.com/c/en/us/solutions/internet-of-things/iot-gateways.html#~{}benefits (accessed on 4 June 2021).
119. Langkemper, S. The Most Important Security Problems with IoT Devices. Available online: https://wwweurofins-cybersecuritycom/news/security-problems-iot-devices/ (accessed on 13 December 2021).
120. Zhou, B.; Li, L. RETRACTED ARTICLE: Security monitoring for intelligent water-saving precision irrigation system using cloud services in multimedia context. Multimed. *Tools Appl.* **2020**, *79*, 9705. [CrossRef]
121. Ogonji, M.M.; Okeyo, G.; Wafula, J.M. A survey on privacy and security of Internet of Things. *Comput. Sci. Rev.* **2020**, *38*, 100312. [CrossRef]

122. Jurcut, A.; Niculcea, T.; Ranaweera, P.; Le-Khac, N.-A. Security Considerations for Internet of Things: A Survey. *SN Comput. Sci.* **2020**, *1*, 193. [CrossRef]
123. Bae, W.-I.; Kwak, J. Smart card-based secure authentication protocol in multi-server IoT environment. Multimed. *Tools Appl.* **2020**, *79*, 15793–15811.
124. Al-Turjman, F.; Lemayian, J.P. Intelligence, security, and vehicular sensor networks in internet of things (IoT)-enabled smart-cities: An overview. *Comput. Electr. Eng.* **2020**, *87*, 106776. [CrossRef]
125. Ahmad, R.; Alsmadi, I. Machine learning approaches to IoT security: A systematic literature Rev. *Things* **2021**, *14*, 100365. [CrossRef]
126. Jun, P.; Lijuan, W.; Li, X.; Xin, Y. Optimal allocation of water resources based on water environment security in Shenbei region, Liaoning. In Proceedings of the 2011 International Symposium on Water Resource and Environmental Protection, Xi'an, China, 20–22 May 2011; pp. 562–565.