Develop IoT-Based Automatic Water Gate Control Prototype with Fuzzy Logic Approach

Jefri Abdurrozak Ismail[1], Wigananda Firdaus Putra Aditya[2], Anies Ekawati[3], Anggraini Puspita Sari[4*], Dwi Arman Prasetya[5]
[1,2,3,4] Master Program Information Technology, Faculty of Computer Science, UPN "Veteran" East Java

Email: [1]jefriabdurrozakismail@gmail.com, [2]wiganandafirdaus@gmail.com,
[3]aniesipiems472@gmail.com, [4*]anggraini.puspita.if@upnjatim.ac.id,
[5]arman.prasetya.sada@upnjatim.ac.id

DOI: https://doi.org/10.31284/j.jtm.2025.v6i1.6758
Received 16 November 2024; Received in revised 17 December 2024; Accepted 19 December 2024; Available online 13 January 2025

Abstract

This research developed a prototype for an automatic water gate control system that integrates Internet of Things (IoT) technology with a Fuzzy Logic approach. The prototype is designed to monitor and regulate water levels in real-time using ultrasonic sensors connected to an IoT network. The water level data is integrated into Amazon Web Services (AWS) for cloud management. Fuzzy Logic was chosen to enhance the system's accuracy and responsiveness to dynamic and unpredictable water levels. The primary goal of this system is to minimize flood risk and ensure adequate water distribution across various sectors by automatically opening the water gates. In initial testing, the prototype successfully transmitted water level data from the sensors to the AWS cloud server and performed fuzzy calculations according to Fuzzy Logic formulas. The prototype demonstrated good results in managing the opening of the water gates based on the water levels detected by the ultrasonic sensors, showing significant potential for water resource management in urban areas through this system.

Keywords: *AWS, Cloud, Fuzzy logic, IoT, Water Gate.*

1. Introduction

A water gate is a structure built along a water flow to control the water level, discharge, and flow. It is used for various purposes such as irrigation, flood control, navigation, and power generation.[1] As the population grows, the role of water gates as flow regulators becomes increasingly crucial. Unfortunately, delays in opening or closing the gates often trigger flooding, especially during the rainy season. By utilizing technologies such as automated systems and sensors, water gate management can become more efficient and accurate, minimizing flood risks and ensuring equitable water distribution.

In this study, automatic water gates emerge as an alternative solution that can be developed to automate the opening of water gates, making water flow management more effective and dynamic. This approach aims to reduce flood risks while ensuring sufficient water availability across various water flow sectors.

Previous research on water level measurement was conducted in 2017, focusing on the water level in Bengawan Solo using ultrasonic sensors and microcontrollers connected via WiFi. This system provided information and alerts to operators when the water level reached the normal threshold through an SMS gateway.[2]

Another study on water level monitoring was conducted by Aruna using the HC-SR04 ultrasonic sensor and NodeMCU. The sensor measured water levels in a reservoir and was connected via WiFi, concluding that water level notifications could be implemented using this

*Corresponding Author

sensor. [3] In 2018, Ahmadil Amin researched water level control using Arduino Uno and an LCD Lm016l, which automatically turned off the water tank filling pump and displayed the water level data on the LCD.[4] The following year, Dave Michael conducted research on monitoring the water capacity of fish ponds. The study concluded that the system could automatically monitor water capacity and prevent evaporation caused by improper capacity levels. [5]

The main focus of this research is the development of a prototype for an automatic water gate integrating Internet of Things (IoT) technology with Fuzzy Logic. The device will be connected to Amazon Web Services (AWS) as a broker. IoT technology is chosen because the system is designed to monitor and regulate water levels in real-time through ultrasonic sensors connected to an IoT network integrated with AWS. Fuzzy Logic is employed to enhance the system's accuracy and responsiveness to dynamic and unpredictable environmental conditions. AWS was selected to connect the device to the server because it provides a reliable cloud infrastructure, enabling IoT devices to handle high data traffic while minimizing obstacles.

In 2021, Wahyu Triono conducted research on a water level monitoring and gate control system using the NodeCode MCU ESP8266 microcontroller, which allowed water control and level monitoring through a web application to minimize flooding.[6] In 2022, Azhar et al. researched an IoT-based water capacity monitoring and automatic filling system using the ESP8266 module, which demonstrated high accuracy and effective water filling control according to predefined limits.[7] The use of ultrasonic sensors combined with the ESP8266 microcontroller has been implemented in 2023, and it can be concluded that based on the test results of the sensor during reservoir water level measurement, the water level measurement performs quite well with an average error rate of 1.07%.[8] In 2024, a similar study revealed that monitoring water levels through the system can help reduce human workload in monitoring and controlling dams and increase the preparedness of communities around dams against potential floods.[9]

In another study, a similar topic was explored in a case study measuring river water levels. The conclusion from this research suggested that IoT devices should preferably be placed in calm water currents.[10] Water level monitoring studies have been conducted and it has been concluded that water levels can be measured through software and water levels can be monitored directly through LCDs.[11] Another study on water level monitoring was titled Design of a Water Level Monitoring System for Flood Detection Based on IoT Using Ultrasonic Sensors. The conclusion emphasized the necessity of a high and stable internet connection to ensure NodeMCU can connect to the server. [12]

Testing the water level sensor prototype developed in this research yielded promising results, where the system effectively managed water gate openings based on detected water levels. The implementation of this technology is expected to significantly contribute to water resource management at several water gates, reduce the risk of undesirable events such as flooding, and improve water distribution in densely populated urban areas. [13]

2. Literature Review

This research aims to develop a prototype for an IoT-based automatic water gate control system using the Fuzzy Logic method. IoT is implemented in this study because it can connect physical objects and devices embedded with sensors, actuators, and internet connectivity, enabling real-time communication and data exchange between devices and systems.[14] Additionally, according to other sources, IoT is a network consisting of physical objects embedded with sensors, actuators, and internet connectivity, allowing them to collect and share data in real time. [15] The Fuzzy Logic approach is chosen for this research because Fuzzy Logic allows values between true and false, providing flexibility in modeling uncertainty.[16] Fuzzy logic is expressed in degrees of membership and degrees of truth, meaning something can be partially true and partially false at the same time.[17] Based on this

concept, this research applies the Fuzzy Logic method to regulate the water gate opening size, enabling it to be adjusted without needing to be fully open or fully closed.

In this study, the water gate openings are managed in real-time so that Fuzzy calculations can be processed within a very short time or instantaneously.[16] To support the real-time processing of IoT data, a cloud service platform is essential. In this case, AWS is selected as the cloud service platform. [18] When deploying a Mosquitto server using an Amazon Machine Image (AMI) on an EC2 instance, Amazon Web Services (AWS) offers significant advantages over other leading cloud providers such as Microsoft Azure, Google Cloud Platform (GCP), and IBM Cloud. AWS Marketplace provides a dedicated Mosquitto AMI that simplifies the deployment process with pre-configured settings and regular updates, ensuring a streamlined and efficient setup. [19] In contrast, Microsoft Azure's Marketplace may offer similar MQTT broker solutions, but often lacks the same level of integration and customization options available with AWS EC2 instances, potentially requiring additional configuration efforts. [20]

Google Cloud Platform's Compute Engine also supports the deployment of Mosquitto through custom images, yet it does not match AWS's extensive global infrastructure, which guarantees superior low-latency performance and high availability across multiple regions. [21] Additionally, AWS EC2 provides a broader selection of instance types and pricing models, enabling more precise scaling and cost management tailored to specific workload demands, whereas GCP and Azure might offer more limited or less flexible options in comparison.

IBM Cloud, while strong in hybrid cloud solutions and enterprise-grade support, does not provide the same level of marketplace convenience and automated deployment features that AWS offers with its Mosquitto AMI. [22] Furthermore, AWS's robust security features, including comprehensive identity and access management, encryption, and compliance certifications, ensure that Mosquitto deployments are secure and meet stringent regulatory standards, which is essential for maintaining the integrity and reliability of MQTT communications.

Moreover, AWS's extensive support ecosystem and detailed documentation facilitate easier troubleshooting and optimization of Mosquitto servers, providing users with the necessary resources to maintain high performance and resilience. In contrast, other cloud providers lack the same depth of support and community resources, making AWS a more reliable and user-friendly choice for deploying Mosquitto via AMI on EC2 instances. [23]

3. Research Methodology

Figure 1 illustrates the research flow for developing an automatic water gate control prototype. The research begins with designing the prototype device, creating a broker hosted on the cloud, implementing the prototype and cloud system, conducting functional testing of the device, collecting data from the tests, and concluding by documenting the research findings in a report.

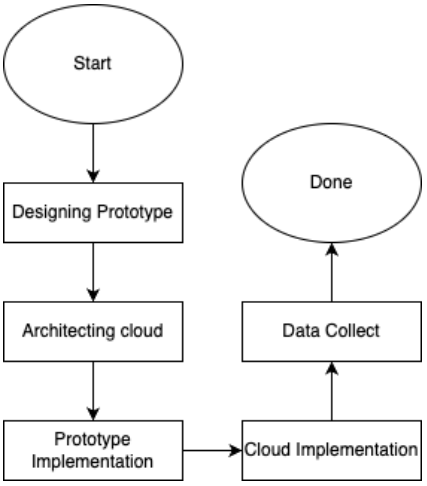

Figure 1. Research Flow Diagram

3.A. Designing the Prototype

Figure 2 illustrates the electronic circuitry of the automatic water gate control prototype. The water level is measured using an ultrasonic sensor SR04, which processes the data via NodeMCU to determine the extent to which the water gate should open. Simultaneously, the data is sent to the cloud. A motor with a driver is used as an actuator to open or close the gate based on the commands from NodeMCU.

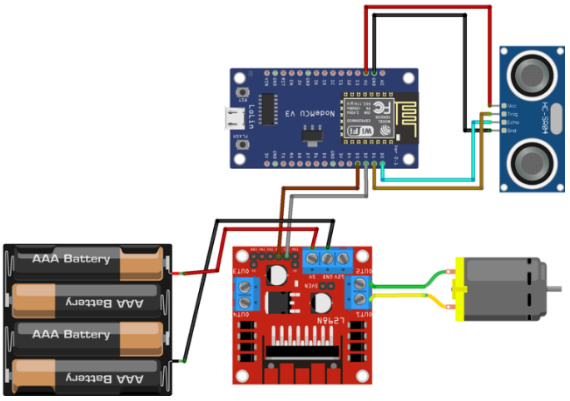

Figure 2. Electronic Circuitry of the Water Gate Prototype

3.B. Designing the Cloud

The MQTT Server architecture on the AWS platform is illustrated in Figure 3. On the AWS cloud, all resources are deployed in the **ap-southeast-3 (Jakarta)** region to ensure that no data leaves Indonesia. To secure the resources, a VPC is created with two public subnets and two private subnets, distributed across two zones (zone A and zone B). An internet gateway connects the VPC to the internet, while a NAT gateway is used to mask the private subnet, ensuring that the IP addresses of resources in the private subnet are not exposed publicly. Mosquitto is run on an EC2 instance located in the public subnet, which has a public IP address accessible directly by the prototype via the internet.

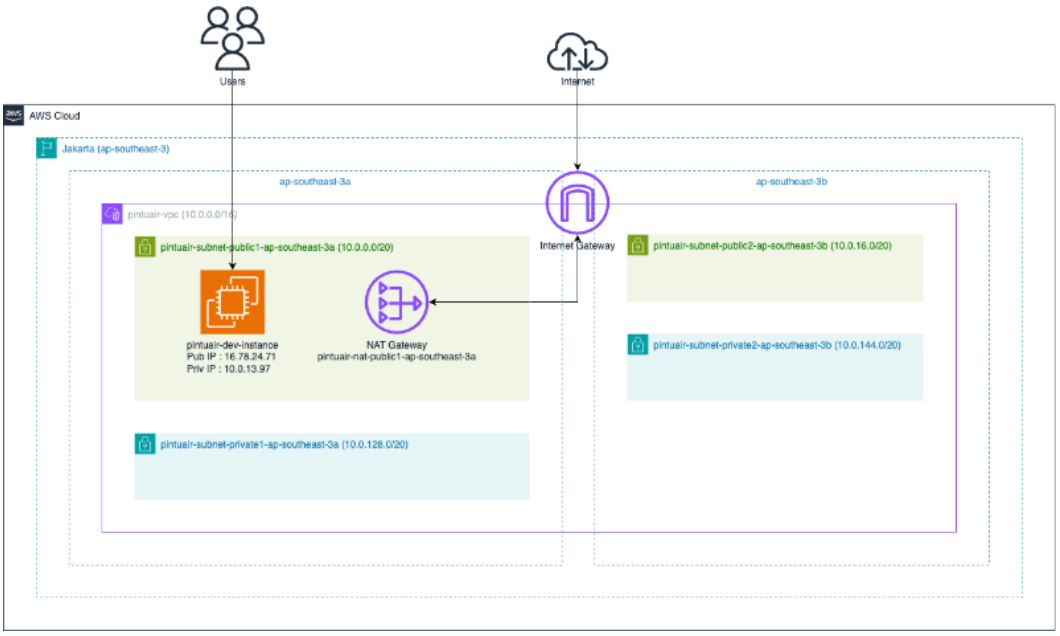

Figure 3. MQTT Server Architecture

3.C. Prototype Implementation

Figure 4 illustrates the physical implementation of the prototype. A jar is used as a water reservoir to simulate a tributary. An ultrasonic sensor is mounted above the jar to measure the water level. The NodeMCU and relay are placed on a breadboard to simplify the wiring process. A small hose is attached to one side of the jar to simulate the water source, while a motorized water flow mechanism is installed on the other side to simulate the water gate that can open and close as needed.

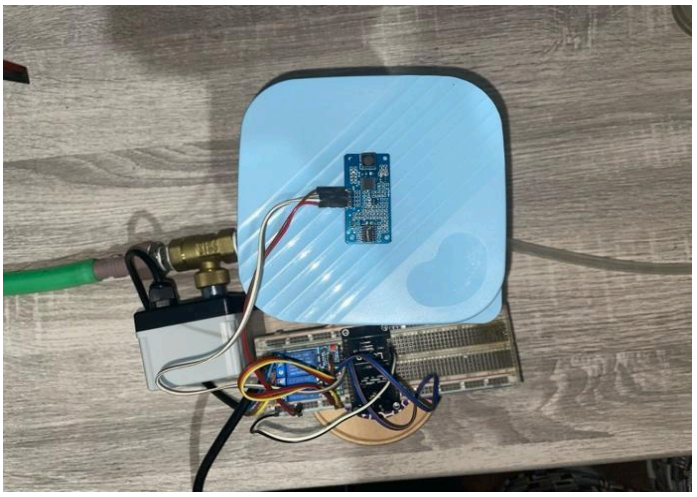

Figure 4. Automatic Water Gate Prototype

3.D. Cloud Implementation

Figure 5 shows the network schema created on the AWS console. The resources include one VPC, four subnets distributed across two different zones, four route tables, and

three network connections. **Figure 6** demonstrates that Mosquitto is running on an EC2 instance and is listening on port 1883.

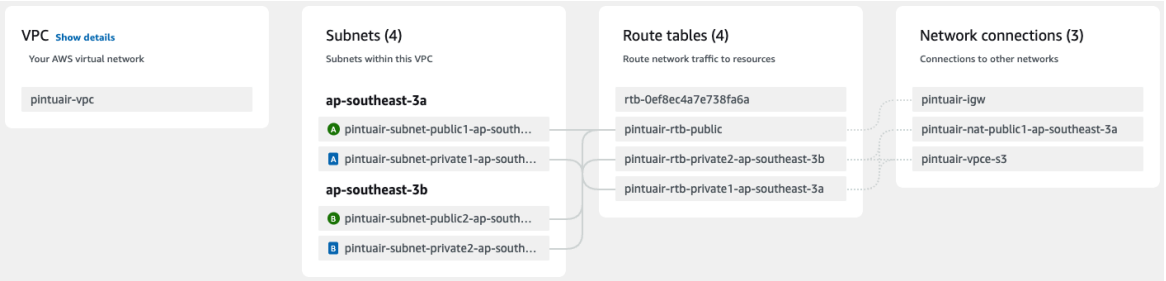

Figure 5. Network Diagram on AWS Cloud

```
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN      959/sshd
tcp        0      0 0.0.0.0:1883            0.0.0.0:*               LISTEN      960/mosquitto
tcp        0      0 127.0.0.53:53           0.0.0.0:*               LISTEN      776/systemd-resolve
tcp6       0      0 :::22                   :::*                    LISTEN      959/sshd
udp        0      0 127.0.0.53:53           0.0.0.0:*                           776/systemd-resolve
udp        0      0 10.0.11.143:68          0.0.0.0:*                           751/systemd-network
```

Figure 6. Port Mapping for Mosquitto EC2 Instance

3.E. Data Collection

During the data collection process, water levels were measured using an ultrasonic sensor. Two scenarios were tested to evaluate the prototype's performance. In the first scenario, the reservoir was initially empty and continuously filled for 30 seconds while the prototype was turned on, allowing the system's response to be observed. In the second scenario, the prototype was initially turned off with the water gate fully closed. The reservoir was filled with water up to a height of 8 cm, after which the prototype was turned on for 30 seconds while the water source continued to flow.

4. Results and Discussion

The results of water level readings from the Scenario 1 experiment are presented in **Table 1**, while the results from Scenario 2 are displayed in **Table 2**.

Table 1. Water Level Test Results for Scenario 1

Time (second)	Water Level (cm)	Time (second)	Water Level (cm)
00:01	0	00:16	3
00:02	1	00:17	4
00:03	1	00:18	4
00:04	1	00:19	4
00:05	1	00:20	4
00:06	1	00:21	3
00:07	2	00:22	3
00:08	2	00:23	3
00:09	2	00:24	4
00:10	2	00:25	4
00:11	2	00:26	4
00:12	3	00:27	4
00:13	3	00:28	3

00:14	**3**	00:29	3
00:15	**3**	00:30	3

Table 2. Water Level Test Results for Scenario 2

Time (Second)	Water Level (cm)	Time (Second)	Water Level (cm)
00:01	8	00:16	5
00:02	8	00:17	5
00:03	8	00:18	5
00:04	8	00:19	5
00:05	7	00:20	4
00:06	7	00:21	4
00:07	7	00:22	4
00:08	7	00:23	4
00:09	7	00:24	4
00:10	6	00:25	4
00:11	6	00:26	4
00:12	6	00:27	4
00:13	6	00:28	3
00:14	6	00:29	3
00:15	5	00:30	3

In this study, Fuzzy memberships are categorized into three levels: (1) *Low:* The membership value is 1 for water levels ranging from 0 to 3, and it decreases linearly from 1 to 0 as the water level increases from 3 to 4. (2) *Medium:* The membership value is 0 for water levels ranging from 0 to 3. It increases linearly from 0 to 1 as the water level rises from 3 to 4 and then decreases linearly from 1 to 0 as the water level increases from 4 to 5. (3) *High:* The membership value is 1 for water levels of 5 or more. It starts increasing from 0 as the water level rises from 4 to 5. A graph representing these membership values is shown in Figure 7.

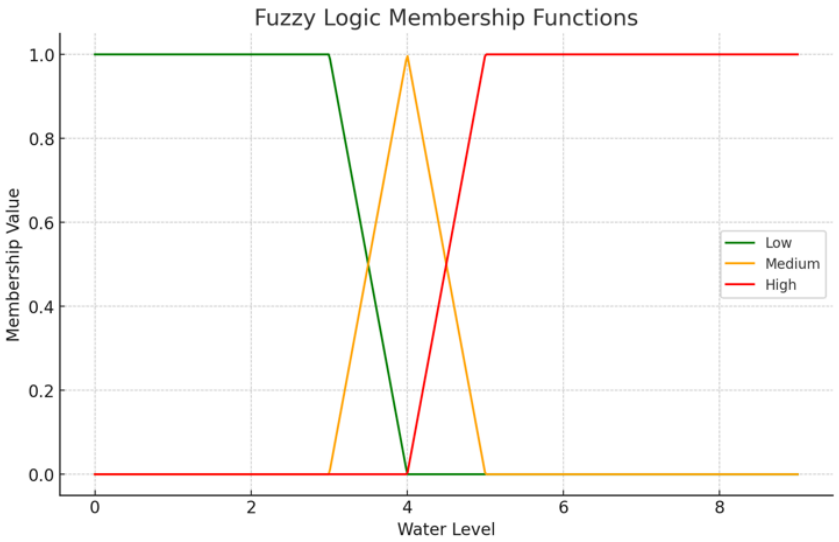

Figure 7. Fuzzy Membership Graph

Mathematically, the membership value can be calculated using equations 1 (low), 2 (medium), and 3 (high). Based on these membership values, the water gate opening value can also be determined using equation 4. Sampling can then be performed using Scenario 1, as shown in Table 3.

$$Low(x) = \{1\ \frac{4-x}{4-2}\ 0 \quad if\ 0 \leq x < 2 \quad if\ 2 \leq x \leq 4\ if\ x > 4 \tag{1}$$

$$Medium(x) = \{0\ \frac{x-4}{6-4}\ \frac{6-x}{6-5}\ 0 \quad if\ x < 4 \qquad if\ 4 \leq x \leq 5\ if\ 5 \leq x \leq 6\ if\ x > 6 \tag{2}$$

$$High(x) = \{1\ \frac{4-x}{4-2}\ 0 \quad if\ 0 \leq x < 2 \quad if\ 2 \leq x \leq 4\ if\ x > 4 \tag{3}$$

$$Gate\ Value = \frac{\sum(M_i . V_i)}{\sum M_i} \tag{4}$$

Where:

- M_i is the membership value (Low, Medium, High)
- V_i is the output value for each fuzzy set (0 for full open, 4 for adjust, 8 for full closed)

Table 3. Sampling for Scenario 1

Time	Water Level	*Fuzzy Value*			Water Gate Opening	
(second)	(cm)	*Low*	*Medium*	*High*	*Value*	*Gate*
00:01	8	1	0	0	10	*Full Close*
00:02	8	1	0	0	10	*Full Close*
00:03	8	1	0	0	10	*Full Close*
00:04	8	1	0	0	10	*Full Close*
00:05	7	1	0	0	10	*Full Close*
00:06	7	1	0	0	10	*Full Close*
00:07	7	1	0	0	10	*Full Close*
00:08	7	1	0	0	10	*Full Close*
00:09	7	1	0	0	10	*Full Close*
00:10	6	1	0	0	10	*Full Close*
00:11	6	1	0	0	10	*Full Close*
00:12	6	1	0	0	10	*Full Close*
00:13	6	1	0	0	10	*Full Close*
00:14	6	1	0	0	10	*Full Close*
00:15	5	1	0	0	10	*Full Close*
00:16	5	1	0	0	10	*Full Close*
00:17	5	1	0	0	10	*Full Close*
00:18	5	1	0	0	10	*Full Close*
00:19	5	1	0	0	10	*Full Close*
00:20	4	0	1	0	5	*Adjust*
00:21	4	0	1	0	5	*Adjust*
00:22	4	0	1	0	5	*Adjust*
00:23	4	0	1	0	5	*Adjust*
00:24	4	0	1	0	5	*Adjust*
00:25	4	1	0	0	10	*Full Close*
00:26	4	1	0	0	10	*Full Close*
00:27	4	1	0	0	10	*Full Close*
00:28	3	0	1	0	5	*Adjust*
00:29	3	0	1	0	5	*Adjust*
00:30	3	0	1	0	5	*Adjust*

5. Conclusion

The automatic water gate control prototype tested in this study demonstrated promising results, successfully managing gate openings based on detected water levels. This prototype holds significant potential for contributing to water resource management across multiple water gates, reducing the risk of undesirable events such as flooding, and improving water distribution in densely populated urban areas. The prototype can further be developed into a mass-produced product through collaboration between researchers and end-users, allowing for user feedback to guide improvements.

Reference

[1] Anwar S, *Teknologi dan Manajemen Pintu Air*. Graha Ilmu, 2019.

[2] I. Rohman and M. Taufiqurrohman, "MONITORING KETINGGIAN AIR PADA BENGAWAN SOLO BERBASIS MIKROKONTROLLER DAN KOMUNIKASI WIFI," in *Seminar Nasional Kelautan XII*, Surabaya: Fakultas Teknik dan Ilmu Kelautan Universitas Hang Tuah, Jul. 2017.

[3] A. K. Rindra, A. Widodo, F. Baskoro, and N. Kholis, "Sistem Monitoring Level Ketinggian Air Pada Tandon Rumah Tangga Berbasis Iot (Internet Of Things)," *JURNAL TEKNIK ELEKTRO*, vol. 11, no. 1, pp. 17–22, Dec. 2021, doi: 10.26740/jte.v11n1.p17-22.

[4] A. Amin, "MONITORING WATER LEVEL CONTROL BERBASIS ARDUINO UNO MENGGUNAKAN LCD LM016L," *EEICT (Electric, Electronic, Instrumentation, Control, Telecommunication)*, vol. 1, no. 1, pp. 41–52, 2018.

[5] D. Michael and D. Gustina, "RANCANG BANGUN PROTOTYPE MONITORING KAPASITAS AIR PADA KOLAM IKAN SECARA OTOMATIS DENGAN MENGGUNAKAN MIKROKONTROLLER ARDUINO," *IKRA-ITH Informatika*, vol. 3, no. 2, pp. 59–66, Jul. 2019.

[6] T. F. Ramadhan and W. Triono, "SISTEM MONITORING KETINGGIAN AIR DAN PENGENDALIAN PINTU AIR BERBASIS MICROCONTROLLER NODECODE MCU ESP8266," *Jurnal Teknologi Informasi dan Komunikasi*, vol. 10, no. 2, Sep. 2021, doi: 10.56244/fiki.v10i2.396.

[7] A. R. Azhar, D. A. Setiawan, N. A. A. Yasmin, T. A. Putri, and G. F. Nama, "SISTEM MONITORING KAPASITAS AIR DAN PENGISIAN OTOMATIS BERBASIS IOT MENGGUNAKAN MODUL ESP8266," *Jurnal Informatika dan Teknik Elektro Terapan*, vol. 12, no. 1, Jan. 2024, doi: 10.23960/jitet.v12i1.3966.

[8] R. Kurniawan, "Rancang Bangun Alat Monitoring Ketinggian Air Pada Reservoir Berbasis Internet Of Things," *Journal ICTEE*, vol. 4, no. 1, p. 23, Mar. 2023, doi: 10.33365/jictee.v4i1.2694.

[9] T. A. Ramadhani, P. M. Gunawan, L. M. M. Fitriani, and A. R. Yusuf, "PROTOTIPE MONITORING KETINGGIAN AIR DAN KONTROL JARAK JAUH PINTU AIR PADA BENDUNGAN," *Jurnal Ilmiah Teknologi dan Rekayasa*, vol. 29, no. 3, pp. 201–213, Dec. 2024, doi: 10.35760/tr.2024.v29i3.11849.

[10] X. B. N. Najoan and M. E. I. Najoan, "Rancang Bangun Aplikasi Monitoring Ketinggian Air Sungai Berbasis Internet of Things Menggunakan Amazon Web Service," *Jurnal Teknik Elektro dan Komputer (JTEK)*, vol. 9, no. 2, pp. 73–80, May 2020.

[11] F. Agustian *et al.*, "SISTEM MONITORING LUAPAN AIR SUNGAI BERBASIS IoT (INTERNET OF THINGS) STUDI KASUS SUNGAI," vol. 5, no. 3, pp. 637–651, 2024, doi: 10.46576/djtechno.

[12] N. Pratama, U. Darusalam, and N. D. Nathasia, "Perancangan Sistem Monitoring Ketinggian Air Sebagai Pendeteksi Banjir Berbasis IoT Menggunakan Sensor Ultrasonik," *JURNAL MEDIA INFORMATIKA BUDIDARMA*, vol. 4, no. 1, p. 117, Jan. 2020, doi: 10.30865/mib.v4i1.1905.

[13] A. F. Andikos and Y. Gusteti, *KOMUNIKASI MANUSIA DENGAN KOMPUTER*. In Media, 2016.

[14] J. Kopke, *The Internet of Things: Connecting Objects*. Tech Press, 2020.

[15] M. Ubaidillah, M. Mustofa, N. I. Maulidiah, and A. Mukhlison, "Kajian Literatur: Internet of Things (IoT) dalam Konteks Industri 4.0," *Jurnal Ilmiah Teknik Elektro (Jitet)*, vol. 14, no. 2, pp. 121–128, 2020.
[16] H. T. Nguyen, C. L. Walker, and E. A. Walker, *A First Course in Fuzzy Logic*, 4th ed. Boca Raton: Chapman & Hall, 2018.
[17] S. Kusumadewi, *Artificial Intelligence (Teknik dan Aplikasinya)*. Yogyakarta: Andi Ofset, 2002.
[18] A. Barr, *Cloud Computing with AWS*. Tech Publishing, 2017.
[19] J. Smith, S. Johnson, and M. Brown, "Automating Cloud Governance: How Organizations Are Streamlining Compliance and Oversight in the Cloud," *ResearchGate*, Nov. 2022.
[20] B. Lee, J. Oh, W. Shon, and J. Moon, "A Literature Review on AWS-Based Cloud Computing: A Case in South Korea," in *2023 IEEE International Conference on Big Data and Smart Computing (BigComp)*, IEEE, Feb. 2023, pp. 403–406. doi: 10.1109/BigComp57234.2023.00099.
[21] N. Villafuerte, S. Manzano, P. Ayala, and M. V. García, "Artificial Intelligence in Virtual Telemedicine Triage: A Respiratory Infection Diagnosis Tool with Electronic Measuring Device," *Future Internet*, vol. 15, no. 7, p. 227, Jun. 2023, doi: 10.3390/fi15070227.
[22] M. A. Sheba, D. A. Mansour, and N. H. Abbasy, "A new low‐cost and low‐power industrial internet of things infrastructure for effective integration of distributed and isolated systems with smart grids," *IET Generation, Transmission & Distribution*, vol. 17, no. 20, pp. 4554–4573, Oct. 2023, doi: 10.1049/gtd2.12951.
[23] R. Praveen, *Data Engineering for Modern Applications*. Bhopal: Addition Publishing House, 2024.

How to cite this article:
Ismail J A, Aditya W F P, Ekawati A, Sari A P, Prasetya D A. Develop IoT-Based Automatic Water Gate Control Prototype with Fuzzy Logic Approach. *Jurnal Teknologi dan Manajemen.* 2025 Januari; 6(1):19-28. DOI: 10.31284/j.jtm.2025.v6i1.6758