JURNAL INFORMATIKA DAN REKAYASA PERANGKAT LUNAK (JATIKA)
Volume 6, Number 2, June 2025, Page 196-207
E-ISSN 2797-2011 P-ISSN 2797-3492
https://publikasi.teknokrat.ac.id/index.php/jatika/index
DOI: https://doi.org/10.33365/jatika.v6i2.349

Implementation of An Internet of Things (IoT)-Based Water Level Early Warning System with Telegram Notification

Agnes Gratia Lengkong[1], Deiby Tineke Salaki [2], Eric Alfonsius [3*]
[1,2,3] Sistem Informasi, Universitas Sam Ratulangi, Indonesia
[1]agnesgratia1016@gmail.com, [2]deibyts.mat@unsrat.ac.id, [3*]ericalfonsius@unsrat.ac.id

Abstract: Water level is a critical parameter in monitoring hydrological dynamics, especially in drainage channels, as significant fluctuations can lead to flooding and disrupt public activities, including those on the campus of the Faculty of Mathematics and Natural Sciences (FMIPA) at Sam Ratulangi University. This study aims to design and implement an Internet of Things (IoT)-based early warning system for water levels that can provide real-time notifications via Telegram. The system integrates an ESP32 microcontroller, HC-SR04 Ultrasonic Sensor, and Raindrop Sensor to monitor rainfall and measure water levels in drainage channels. The system is developed using the Waterfall model and tested using a black-box testing approach. Notifications are automatically sent to Telegram when the water level status reaches the "Alert" or "Overflow" categories and stop when it returns to the "Safe" status. Based on testing results, the system achieved a 100% success rate, indicating that it operated correctly according to the specified design without any errors during testing. This study contributes to local flood mitigation efforts and improves the preparedness of the academic community in response to rising water levels.
Keywords: IoT, Telegram, ESP32, Water Level;

1. INTRODUCING

An early warning system is a structured approach aimed at notifying communities of potential hazards in a timely manner. It comprises several key components: risk knowledge, monitoring and warning services, information dissemination, and response capability. The effectiveness of such systems is closely linked to social, economic, and environmental factors, and generally follows three critical stages: knowledge of past events, forecasting of future hazards, and early alert delivery [1].

One of the most relevant applications of early warning systems is in monitoring water levels in drainage systems or water bodies such as culverts. Significant fluctuations in water levels, particularly during the rainy season, can result in flooding, posing risks to both infrastructure and human safety. The primary cause of water overflow is the increased water volume that exceeds the drainage system's capacity. Therefore, beyond structural improvements, there is a pressing need to develop early warning systems using Internet of Things (IoT) technology to enable real-time flood detection and mitigation [2][3][4].

The Internet of Things (IoT) is a concept in which physical devices are interconnected and communicate via the internet. A typical IoT system consists of sensors, microcontrollers, networking modules, user interfaces, power sources, and software, all of which work autonomously to collect and transmit data. This allows for real-time environmental monitoring and responsive decision-making with minimal human intervention [5]

Numerous studies have demonstrated the effectiveness of IoT in disaster mitigation. For instance, [6] designed a real-time flood monitoring system using ultrasonic sensors and a Raspberry Pi with Twitter-based notifications. Meanwhile, [7] developed a rain detection system using a raindrop sensor and Arduino to trigger visual and sound alerts. Additionally, [8] implemented a vibration detection system using ESP32, automatically sending earthquake alerts via Telegram. These studies highlight the flexibility of IoT in environmental applications and underscore the importance of integrating multiple sensors and communication channels to enhance system accuracy and responsiveness.

Eric Alfonsius: *Corresponding Author

JURNAL INFORMATIKA DAN REKAYASA PERANGKAT LUNAK (JATIKA)
Volume 6, Number 2, June 2025, Page 196-207
E-ISSN 2797-2011 P-ISSN 2797-3492
https://publikasi.teknokrat.ac.id/index.php/jatika/index
DOI: https://doi.org/10.33365/jatika.v6i2.349

The NodeMCU ESP32 is a next-generation microcontroller with built-in WiFi and Bluetooth, offering greater memory capacity, faster processing, and more GPIO and analog input pins than its predecessor, the ESP8266. These features make it highly suitable for IoT-based systems without requiring additional hardware [9]

The HC-SR04 ultrasonic sensor is utilized to measure water levels by calculating the time taken for sound waves to bounce back from a surface. Complementarily, the raindrop sensor detects the presence of rainfall by identifying electrical conductivity changes on its surface. Both sensors are integrated with the ESP32 and programmed using the Arduino IDE, a development environment based on the C programming language [10][11].

For alert delivery, this system employs Telegram, a cloud-based instant messaging platform that supports Bot APIs, enabling automatic communication between IoT devices and end users. Alerts are pushed to relevant stakeholders when critical water levels are detected, facilitating prompt action [12]

System evaluation is conducted using the blackbox testing method, which assesses functional behavior from a user perspective without considering internal code structure. This ensures the system meets its design specifications [13]

At the Faculty of Mathematics and Natural Sciences, Sam Ratulangi University, localized flooding is a recurring problem during heavy rainfall. Water accumulation on major roads disrupts mobility, interrupts academic activities, and poses safety risks to the university community. The issue stems from undersized drainage channels and clogged discharge outlets. With a drainage dimension of 80 cm in height and 120 cm in width, the system reaches its capacity during prolonged rainfall, leading to overflow.

Additionally, water level information is often inaccessible to campus users, especially those indoors or en route. This lack of timely information increases vulnerability and reduces preparedness. Hence, a real-time water level monitoring and early warning system is essential.

This study proposes the development of an IoT-based early warning system integrating the ESP32 microcontroller, HC-SR04 ultrasonic sensor, and a raindrop sensor. The system monitors water levels and weather conditions, and sends real-time alerts via Telegram to facilitate timely responses by campus authorities and users. The proposed solution aims to improve flood preparedness and resilience at the local level, supporting effective disaster risk mitigation.

2. RESEARCH METHODOLOGY

This research uses the Waterfall method. The Waterfall model is one of the methods in the SDLC (Software Development Life Cycle) which has a characteristic where each phase must be completed first before proceeding to the next phase [14]:

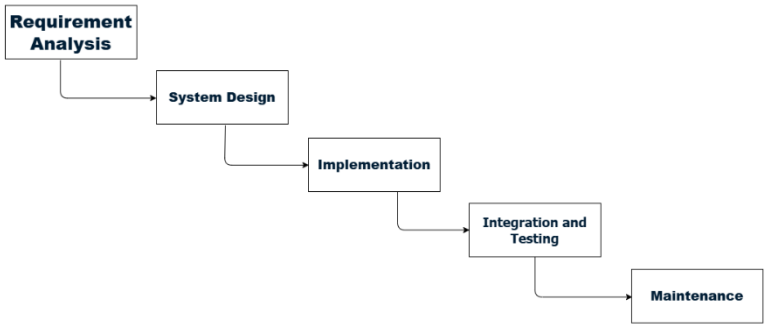

Figure 1. Research Methodology

2.1. Needs Analysis

In the development of an IoT-based early warning system, all system needs must be identified in depth during the initial phase. These needs are obtained through various methods such as interviews, surveys, and discussions with stakeholders. Once this information is collected, it is analyzed to produce system requirement documentation that will be used in the next phase of development. The system is designed to provide alerts related to water levels with automatically classified risk levels.

Eric Alfonsius: *Corresponding Author

JURNAL INFORMATIKA DAN REKAYASA PERANGKAT LUNAK (JATIKA)
Volume 6, Number 2, June 2025, Page 196-207
E-ISSN 2797-2011 P-ISSN 2797-3492
https://publikasi.teknokrat.ac.id/index.php/jatika/index
DOI: https://doi.org/10.33365/jatika.v6i2.349

2.2. System Design

This stage is carried out before the system implementation process. The purpose of this stage is to provide an overview of what to do and how the system runs. At this stage the model, system, and hardware design is carried out. This stage helps in determining *hardware* and system requirements and defining the overall system architecture.

2.3. Implementation

At this stage, the assembly process of the device consisting of NodeMCU ESP32, *Raindrop sensor* and HC-SR04 ultrasonic sensor is carried out. In addition, the program code is written on the Arduino IDE to integrate the hardware with the software. The system will be connected to a *database* to monitor the water level in *real-time*. In addition, the system will also be designed to send automatic notifications to the Telegram application when there is a change in water level status according to a predetermined risk classification.

2.4. Integration and Testing

At this stage, the tools and systems that have been developed are combined and tested. This test aims to ensure that the system built is in accordance with the predetermined design and to identify whether there are still errors or problems that need to be corrected. Testing is done with a *blackbox* approach.

2.5. Maintenance

This is the last stage in the Waterfall model. At this stage, the completed system is put into operation and maintenance is performed. Maintenance includes fixing errors that may not have been detected in previous steps, improving system implementation, and upgrading system services according to new needs.

3. RESULT AND DISCUSSIONS

3.1. Needs Analysis Results

In the needs analysis stage, identification and formulation of system requirements based on research objectives are carried out. This research aims to design and develop a microcontroller-based system that can detect rainfall and measure the distance of water levels to provide notifications to users via Telegram.

a. Hardware Requirements

System design requires key components to ensure optimal performance and control as required. These components function in data acquisition, information processing, and control of connected devices. The designed system requires several key components:

Table 1. Hardware Components

Hardware	Description
NodeMCU ESP32	As the main microcontroller responsible for sensor data processing and communication with the server
HC - SR04 Ultrasonic Sensor	Used to measure the water level to determine if it exceeds a predetermined threshold.
Raindrop Sensor	Serves to detect rainy weather conditions that will activate the HC - SR04 Ultrasonic sensor
Blinds	As a power source for the ESP32 and the sensors used.

b. Software Requirements

Software plays an important role in running the system, processing data from sensors, and controlling connected hardware. The following are the software requirements used in this system:

Table 2. Software Components

Software	Description
Arduino IDE	Used to program the ESP32 with the C++ programming language.

Eric Alfonsius: *Corresponding Author

JURNAL INFORMATIKA DAN REKAYASA PERANGKAT LUNAK (JATIKA)
Volume 6, Number 2, June 2025, Page 196-207
E-ISSN 2797-2011 P-ISSN 2797-3492
https://publikasi.teknokrat.ac.id/index.php/jatika/index
DOI: https://doi.org/10.33365/jatika.v6i2.349

Localhost Database (phpMyAdmin)	Used to store sensor reading data.
Telegram API	Used to send notifications to the user when the water level exceeds the threshold.
Web Server (Localhost/phpMyAdmin)	Serves as a place to display data sent by ESP32

c. Water Level Classification Criteria

Field observations and interviews with FMIPA field officer Muhammat Subeti indicate that the drainage system on campus begins to overflow when water levels exceed 80 cm, disrupting activities and access. Building on [15] who categorized water levels into Safe (<10 cm), Alert (10–60 cm), and Caution (≥60 cm), this study refines the classification to reflect local conditions as follows:

- Safe: ≤20 cm
- Alert: >20–60 cm
- Caution: >60–80 cm
- Danger (Overflow): >80 cm

3.2. System Design

In system design, stages are carried out that include designing system architecture, *hardware* and *software* to ensure the system can function as needed.

a. Hardware

Hardware design includes the selection of hardware specifications such as the ESP32 microcontroller, HC-SR04 Ultrasonic Sensor, Raindrop Sensor, Battery and network devices in accordance with system requirements. The hardware design can be seen in Figure 2.

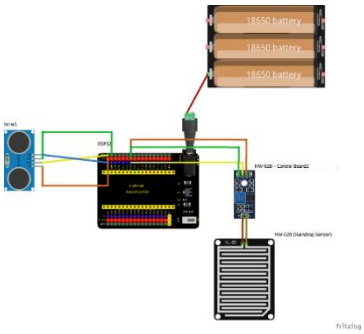

Figure 2. Hardware Design

b. Software

In this system, software development uses Arduino IDE for microcontroller programming, while data communication is done through Telegram as a notification platform or user interface. In addition, the system utilizes a local database (localhost) to store and manage data, and uses local storage as storage media.

c. System Workflow

The process begins with the device detecting data using a rain sensor. If rain is detected, the HC-SR04 ultrasonic sensor will be activated to measure the water level. If no rain is detected, the rain sensor will continue to standby. When the HC-SR04 ultrasonic sensor is active, the sensor reads the water level and then the data is sent to the data storage and then the system automatically evaluates whether the value obtained from the sensor has exceeded the predetermined threshold, then the system will send an alert via Telegram, which is then forwarded to the designated group. If the value is still within normal limits, the system will continue to monitor the sensor data periodically. The System Workflow can be seen in Figure 3.

Eric Alfonsius: *Corresponding Author

JURNAL INFORMATIKA DAN REKAYASA PERANGKAT LUNAK (JATIKA)
Volume 6, Number 2, June 2025, Page 196-207
E-ISSN 2797-2011 P-ISSN 2797-3492
https://publikasi.teknokrat.ac.id/index.php/jatika/index
DOI: https://doi.org/10.33365/jatika.v6i2.349

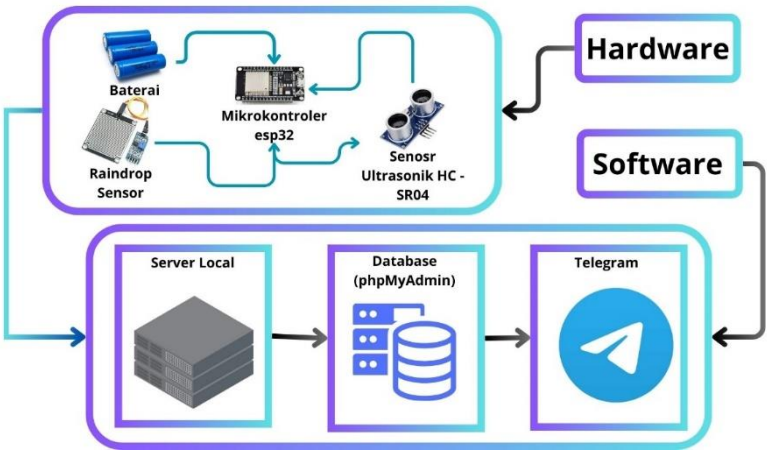

Figure 3. System Workflow

d. The process begins with the device detecting data using a rain sensor. If rain is detected, the HC-SR04 ultrasonic sensor will be activated to measure the water level. If no rain is detected, the rain sensor will continue to standby. When the HC-SR04 ultrasonic sensor is active, the sensor reads the water level and then the data is sent to the data storage and then the system automatically evaluates whether the value obtained from the sensor has exceeded the predetermined threshold, then the system will send an alert via Telegram, which is then forwarded to the designated group. If the value is still within normal limits, the system will continue to monitor the sensor data periodically. The System Workflow can be seen in Figure 4.

Figure 4. Hardware

3.3. Implementation

After the system design, the system is implemented by installing and configuring the hardware and software so that it can function according to the specifications that have been set. This process includes component installation, connectivity testing, and sensor calibration to ensure the accuracy and reliability of the system in operation.

3.3.1. Hardware Implementation

Hardware installation begins with assembling the components that have been designed in the previous stages, which consist of the HC-SR04 Ultrasonic Sensor, Raindrop Sensor, ESP32 Microcontroller, and Battery as a power source. In Figure 4, shows the components that have been assembled into one.

Table 3. Configuration of Ultrasonic Sensor to ESP32

Cable Color	Function	ESP32
Green	Echo	IO22
Yellow	Trig	IO23

Eric Alfonsius: *Corresponding Author

JURNAL INFORMATIKA DAN REKAYASA PERANGKAT LUNAK (JATIKA)
Volume 6, Number 2, June 2025, Page 196-207
E-ISSN 2797-2011 P-ISSN 2797-3492
https://publikasi.teknokrat.ac.id/index.php/jatika/index
DOI: https://doi.org/10.33365/jatika.v6i2.349

Orange	VCC	3.3 Volts
Blue	GND	GND

Table 4. Rain Sensor to ESP32 Configuration

Cable Color	Function	ESP32
Green	Echo	IO22
Yellow	Trig	IO23
Orange	VCC	3.3 Volts

3.3.2. Software Implementation

Configuration between hardware and software is done to ensure the system can operate optimally according to the specifications that have been designed. This process includes writing program code on the Arduino IDE, designing and implementing databases, and integration with the telegram API so that information can be conveyed.

a. Integration with Arduino IDE

The program code written in the Arduino IDE generally consists of several main parts that form the basic structure of microcontroller programming, such as the initialization function (setup) and the main loop function (loop) which is executed repeatedly.

b. Localhost Database (phpMyAdmin)

In the system implementation, the sensor_db database created on localhost will be used to store the sensor reading data from the ESP32. This database has one main table, namely distance, which serves to store information about the water level measured by the ultrasonic sensor as well as the status of the water condition. The structure of the distance table is shown in Table 5.

Table 5: Database Table Structure

Column	Function
id	The primary key column with int type is set as AUTO_INCREMENT, so that every incoming data will get a unique ID automatically.
distance	A column that stores the converted water level value from the ultrasonic sensor reading.
timestamp	A column of type timestamp, which automatically stores the time when the data was entered into the database.
criteria	A column that stores the status of the water condition based on the measured height, such as "Safe", "Increasing", "Alert", or "Overflowing".

c. Localhost Server

In the implemented system, the ESP32 microcontroller reads data from the rain sensor and ultrasonic sensor, then sends information in the form of water level values and their status to the server. The received data is stored in a database and displayed through a web-based monitoring dashboard which is run on a local web server (localhost). This interface was developed using the Hypertext Preprocessor (PHP) programming language.

d. Telegram API

In this system, the notification mechanism to Telegram is developed using Python. The Python code periodically retrieves the latest data from the MySQL database, then compares it with the previous data that has been processed. If there is a change in the status of the water level, the system will send a notification to Telegram using the Telegram API, so that users can get early warnings about potential hazards.

The system works by retrieving the latest data from the distance table, then comparing the ID and water status (criteria) with the last processed data. If there is a change in status or new

Eric Alfonsius: *Corresponding Author

JURNAL INFORMATIKA DAN REKAYASA PERANGKAT LUNAK (JATIKA)
Volume 6, Number 2, June 2025, Page 196-207
E-ISSN 2797-2011 P-ISSN 2797-3492
https://publikasi.teknokrat.ac.id/index.php/jatika/index
DOI: https://doi.org/10.33365/jatika.v6i2.349

data, the system will send a notification via Telegram bot using the Telegram API. This system runs automatically using a *schedule* that checks data every 1 minute, so that it can provide *real-time* warnings about water level conditions.

3.4. Integration and Testing

Integration and testing are carried out to ensure that the system can function according to the specifications that have been designed. Testing is carried out in several stages, namely individual testing of each sensor, communication testing between sensors and microcontrollers, and testing the entire system.

3.4.1. Sensor Readings

The system uses two main types of sensors, namely Ultrasonic Sensor and Raindrop Sensor HW-028, which have their respective functions in supporting the performance of the system.

1. Raindrop Sensor Testing

The Raindrop Sensor will activate when exposed to water. The serial monitor will show information when there is rain or not. Can be seen in Figure 5.

Figure 5. Raindrop Sensor Testing

2. HC-SR04 Ultrasonic Sensor Testing

Device testing was carried out using a trapezoid-shaped container as a test medium. To increase the effectiveness of sensor reflection, cork material is used as a coating. Tests are carried out according to the criteria that have been classified.

In Figure 6, the test was conducted with a water level of 68.04 cm, which corresponds to the "Alert" category based on the water level classification in the system. This value was obtained from the measurement results using an ultrasonic sensor that recorded the original distance of 16.42 cm. The measurement data was then successfully sent and stored into the database storage via an HTTP connection, which shows that the data transmission process from the device to the server runs well without any problems.

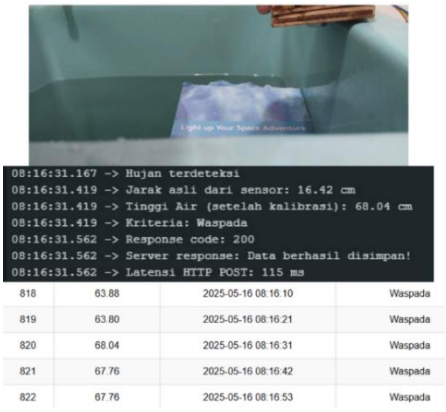

Figure 6. Ultrasonic Sensor Testing

Eric Alfonsius: *Corresponding Author

JURNAL INFORMATIKA DAN REKAYASA PERANGKAT LUNAK (JATIKA)
Volume 6, Number 2, June 2025, Page 196-207
E-ISSN 2797-2011 P-ISSN 2797-3492
https://publikasi.teknokrat.ac.id/index.php/jatika/index
DOI: https://doi.org/10.33365/jatika.v6i2.349

3.4.2. Network Mechanism

This test was conducted to evaluate the network mechanism involving the local server WiFi connection, and the Telegram platform. The system is tested to ensure each network component functions properly in sending and receiving data.

If the device is successfully connected to a WiFi network, the system will display an indicator that the connection has been established, as shown in Figure 7A. Furthermore, if the data obtained from the sensor is successfully sent to the local server, the system will provide a description that the data transmission process has been successful, as shown in Figure 7B. In addition, when the system successfully sends an alert notification to Telegram, a confirmation message will appear stating that the notification has been sent successfully as shown in Figure 7C.

Figure 7. Network Mecasime Testing

The system will send notifications to Telegram only under certain conditions to avoid sending repetitive or unnecessary data. Notifications will be sent when the water level criteria are at the "Alert" or "Overflow" level, indicating a potential risk. In addition, the system will also send a final notification when the status returns to "Safe", but only if the previous status was "Alert" or "Overflow", to indicate that conditions have returned to normal. If the latest data has the same ID and criteria as the previous data, the system will not send a notification to avoid unnecessary repetition. In this way, the system ensures that only significant changes in water status will be communicated through notifications. Figure 8 shows the alert notification sent to Telegram.

Figure 8. Alert Notification to Telegram

Eric Alfonsius: *Corresponding Author

JURNAL INFORMATIKA DAN REKAYASA PERANGKAT LUNAK (JATIKA)
Volume 6, Number 2, June 2025, Page 196-207
E-ISSN 2797-2011 P-ISSN 2797-3492
https://publikasi.teknokrat.ac.id/index.php/jatika/index
DOI: https://doi.org/10.33365/jatika.v6i2.349

Table 5. shows the test results of all components functioning as expected. The NodeMCU ESP32 successfully connects to a WiFi network with a stable connection and is detected by the computer, allowing communication through the serial port to read and display sensor data on the Serial Monitor. The ultrasonic sensor works well in detecting and measuring the water level accurately, while the Raindrop Sensor is able to detect rainfall.

In addition, the NodeMCU ESP32 can connect with the XAMPP Server, send data to the web server, and ensure the data is stored properly in the database. The test results also show that the system can display output in accordance with the program logic, so it can be used to monitor environmental conditions in real-time.

Table 5. System Testing Results

No.	Device	Test Scenario	Expectation of Results	Results	Status	Description
1	NodeMCU ESP32	Powering on the ESP32 and connecting it to a WiFi network	ESP32 successfully connects to WiFi network	☑	Pass	Stable WiFi connection
		Connecting the ESP32 to a computer via USB	The computer recognizes the ESP32 as a serial device	☑	Pass	Device recognized
		Accessing data via Serial Monitor	Data displayed as sent from ESP32	☑	Pass	Data read on serial monitor
		Connect the ultrasonic sensor to the ESP32 and read the output	Sensor provides distance data in cm	☑	Pass	Sensors work normally
		Set up a connection to the local server (XAMPP)	ESP32 successfully connects to the server	☑	Pass	Stable server connection
		Running the main program	Serial Monitor displays data according to program logic	☑	Pass	Output as per program
		Send the sensor reading data to the web server	Data goes to MySQL database in XAMPP	☑	Pass	Data is stored properly
2	Raindrop Sensor	Dripping water on the sensor surface	Sensor detects rain and sends signal to ESP32	☑	Pass	Sensors work normally
3	Ultrasonic Sensor	Places the object at a certain distance and reads the results on the Serial Monitor	Distance values are accurate and consistent with the object position	☑	Pass	Accurate measurement
4	XAMPP Server	Accessing a web application from a local browser	The web application opens and	☑	Pass	Successful connection

Eric Alfonsius: *Corresponding Author

JURNAL INFORMATIKA DAN REKAYASA PERANGKAT LUNAK (JATIKA)
Volume 6, Number 2, June 2025, Page 196-207
E-ISSN 2797-2011 P-ISSN 2797-3492
https://publikasi.teknokrat.ac.id/index.php/jatika/index
DOI: https://doi.org/10.33365/jatika.v6i2.349

		displays the main page			
	Send data from ESP32 to server and check database contents	The latest data appears in the database table	☑	Pass	Data is stored in the database

Based on the test results that have been carried out, Table 6. shows consistent and accurate performance in classifying water levels according to predetermined limits. In the "Safe" condition, the sensor detects the water level below or equal to 20 cm, and the system only sends data to the web with a delay of 10 seconds without activating Telegram notifications. This shows that the system is able to maintain data communication efficiency without overloading the notification network when conditions are still classified as safe.

Furthermore, in the "Alert" state, the detected water level is within the range of 21.39 cm to 56.09 cm. In this state, the system still only sends data to the web every 10 seconds and does not send Telegram notifications, in accordance with the system design that considers the efficient use of messaging services only for more critical conditions.

At "Alert" status, the sensor successfully detected water levels between 63.88 cm and 74.94 cm. In accordance with the system requirements, Telegram notifications were activated with a 60-second delay in sending. The activation of notifications in this condition shows that the system is able to anticipate potential dangers by giving early warning to users even though the water level has not reached the maximum limit.

Finally, in the "Overflowing" condition, the sensor detected a water level above 80 cm, 83.69 cm to be precise. The system again showed an appropriate response by continuing to send data periodically to the web and sending notifications via Telegram as a form of critical warning. This shows that the system is not only able to recognize emergency conditions, but also able to trigger warning mechanisms as expected.

Overall, the test results show that the system works reliably and in accordance with the designed functional specifications. Each status category is successfully recognized, and the system provides outputs in accordance with predefined rules, both in terms of data submission delays and sending alert notifications.

Table 6. Sensor Testing Results and Data Delivery

No.	Test Scenario	Expected Output	Test Results
1	Sensor detects water level of 3.87 cm	Status: Secure, Delay to Web: 10 seconds, No Telegram notifications	☑ Successful
2	Sensor detects water level of 10.33 cm	Status: Secure, Delay to Web: 10 seconds, No Telegram notifications	☑ Successful
3	Sensor detects water level of 19.31 cm	Status: Secure, Delay to Web: 10 seconds, No Telegram notifications	☑ Successful
4	Sensor detects water level of 32.90 cm	Status: Standby, Delay to Web: 10 seconds, No Telegram notifications	☑ Successful
5	Sensor detects water level of 21.39 cm	Status: Standby, Delay to Web: 10 seconds, No Telegram notifications	☑ Successful
6	Sensor detects water level of 56.09 cm	Status: Standby, Delay to Web: 10 seconds, No Telegram notifications	☑ Successful
7	Sensor detected water level 68.04 cm	Status: Alert, Delay to Web: 10 seconds, Telegram Notification after 60 seconds	☑ Successful

Eric Alfonsius: *Corresponding Author

JURNAL INFORMATIKA DAN REKAYASA PERANGKAT LUNAK (JATIKA)
Volume 6, Number 2, June 2025, Page 196-207
E-ISSN 2797-2011 P-ISSN 2797-3492
https://publikasi.teknokrat.ac.id/index.php/jatika/index
DOI: https://doi.org/10.33365/jatika.v6i2.349

8	Sensor detected water level of 74.94 cm	Status: Alert, Delay to Web: 10 seconds, Telegram Notification after 60 seconds	☑ Successful
9	Sensor detected water level 63.88 cm	Status: Alert, Delay to Web: 10 seconds, Telegram Notification after 60 seconds	☑ Successful
10	Sensor detected water level 82.12 cm	Status: Overflow, Delay to Web: 10 seconds, Telegram Notification after 60 seconds	☑ Successful
11	Sensor detects water level of 80.04 cm	Status: Overflow, Delay to Web: 10 seconds, Telegram Notification after 60 seconds	☑ Successful
12	Sensor detects water level of 83.69 cm	Status: Overflow, Delay to Web: 10 seconds, Telegram Notification after 60 seconds	☑ Successful

Based on the test results obtained:

$$Success\ Rate = \left(\frac{Number\ of\ Successful\ Tests}{Total\ Number\ Of\ Test}\right) \times 100\%$$
$$= \left(\frac{12}{16}\right) \times 100\% = 100\%$$

Based on the test results, a success percentage of 100% was obtained, which indicates that the system has functioned properly according to the designed specifications without any errors found in the test.

3.5. Maintenance

The system that has been built requires regular maintenance to ensure that its performance remains optimal and all its functions can be utilized effectively. The following maintenance can be done on the system:

a) Sensor Inspection: Ensure that the rain sensor is functioning properly and that there is no dirt or corrosion on its surface and clean the ultrasonic sensor from dust or dirt that may interfere with the water surface distance reading.
b) ESP32 check: Checking the connection between the ESP32 and the sensor to ensure that there are no loose or disconnected cables and performing a reset and firmware update of the ESP32 if needed to improve system stability.
c) Power Source: Recharges the battery if it has run out of power.
d) Program Code Update: Evaluate and update the program code on the ESP32 to keep it optimized. For example, the URL for storing data needs to be updated as it may change as well as ensuring communication between the ESP32 and the server (localhost database) runs smoothly.
e) Network Connection Check: Ensure that the WiFi network is stable so that the ESP32 can send data to the database and Telegram without interruption and restart the router or access point in case of connectivity interruption.
f) Database Management: Cleaning up old data in the localhost database to avoid overloading the storage system and performing regular data backups to prevent loss of important information.

Telegram Notification Monitoring: Ensure that Telegram bots remain active and able to send alerts correctly and test the notification feature regularly to ensure the system's response to sending early warnings remains optimal.

4. CONCLUSION

This research successfully designed and developed a prototype of an Internet of Things (IoT)-based water level early warning system capable of monitoring water levels in real-time using an ESP32 microcontroller, HC-SR04 Ultrasonic Sensor, and Raindrop Sensor. The system can detect water conditions based on classification levels ("Safe," "Rising," "Alert," and "Overflowing") and automatically send notifications via Telegram when the water level reaches the "Alert" and "Overflowing" statuses, with a final notification once the status returns to "Safe." System testing was conducted using a black-

Eric Alfonsius: *Corresponding Author

JURNAL INFORMATIKA DAN REKAYASA PERANGKAT LUNAK (JATIKA)
Volume 6, Number 2, June 2025, Page 196-207
E-ISSN 2797-2011 P-ISSN 2797-3492
https://publikasi.teknokrat.ac.id/index.php/jatika/index
DOI: https://doi.org/10.33365/jatika.v6i2.349

box method and demonstrated a 100% success rate, indicating that all functions operated correctly according to the system specifications with no errors encountered during the testing process. The system also exhibited reliable accuracy, fast response times in sending notifications, and effective integration with a local server for data logging. The implementation of this system offers tangible benefits for the academic community at FMIPA Sam Ratulangi University by enhancing flood preparedness and minimizing the potential disruption of campus activities caused by rising water levels in the drainage system.

5. REFERENCES

[1] I. N. Saputro, F. A. Wakid, and S. S. Dewi, "Pembuatan sistem peringatan dini angin puting beliung di Desa Demakijo, Kecamatan Karangnongko, Kabupaten Klaten," *BEMAS: Jurnal Bermasyarakat*, vol. 4, no. 2, pp. 351–356, 2024.

[2] H. Purwanto, M. Riyadi, D. W. W. Astuti, and I. W. A. W. Kusuma, "Komparasi sensor ultrasonik HC-SR04 dan JSN-SR04T untuk aplikasi sistem deteksi ketinggian air," *Simetris: Jurnal Teknik Mesin, Elektro dan Ilmu Komputer*, vol. 10, no. 2, pp. 717–724, 2019.

[3] S. Sukisno and M. F. O. Kurniawan, "Prototipe Sistem Pendeteksi Banjir Berbasis IoT (Internet Of Things)," *Unistek: Jurnal Pendidikan dan Aplikasi Industri*, vol. 9, no. 2, pp. 149–156, 2022.

[4] M. B. Ulum and F. Badri, "Sistem Monitoring Cuaca Dan Peringatan Banjir Berbasis IoT Dengan Menggunakan Aplikasi Mit App Inventor," *JITET (Jurnal Informatika dan Teknik Elektro Terapan)*, vol. 11, no. 3, pp. 319–328, 2023.

[5] I. P. Sari, A. Novita, A. K. Al-Khowarizmi, F. Ramadhani, and A. Satria, "Pemanfaatan Internet of Things (IoT) pada Bidang Pertanian Menggunakan Arduino UnoR3," *Blend Sains Jurnal Teknik*, vol. 2, no. 4, pp. 337–343, 2024.

[6] E. P. Tenda, A. V. Lengkong, and K. F. Pinontoan, "Sistem Peringatan Dini Banjir Berbasis IoT dan Twitter," *CogITo Smart Journal*, vol. 7, no. 1, pp. 26–39, 2021.

[7] E. B. Y. Prakoso, D. D. Saputri, D. Joedhistiro, E. Irawan, and R. Susanto, "Rancang Bangun Alet Alarm Peringatan Hujan dengan Menggunakan Sensor Raindrop Untuk Petani," *Uranus: Jurnal Ilmiah Teknik Elektro, Sains dan Informatika*, vol. 2, no. 3, pp. 83–92, 2024.

[8] P. T. Wikantama, M. Bahalwan, and M. A. G. Akmal, "SIGEMPA: Sistem Peringatan Dini Gempa Bumi berbasis IoT dengan ESP32," *Jurnal Teknik Mesin, Elektro dan Ilmu Komputer*, vol. 4, no. 1, pp. 63–70, 2024.

[9] D. S. E. Estu, M. Yantidewi, M. B. Adikuasa, B. M. Rusdi, and M. Khoiro, "Alat Monitoring Ketinggian Air Laut Berbasis IoT dengan NodeMCU ESP32 dan HC-SR04," *Jurnal Kolaboratif Sains*, vol. 6, no. 7, pp. 586–597, 2023.

[10] M. Husein, A. Armanto, and A. Sobri, "Monitoring Sistem Pendeteksi Ketinggian Bencana Banjir dengan Sensor Ultrasonik Berbasis IoT," *JUSIKOM: Jurnal Sistem Komputer Musirawas*, vol. 8, no. 1, pp. 68–79, 2023.

[11] N. Nurlaila, S. Paembonan, and R. Suppa, "Rancang Pendeteksian Kecepatan Kendaraan Berbasis Arduino," *Jurnal Informatika dan Teknik Elektro Terapan*, vol. 12, no. 3, 2024.

[12] A. F. Waluyo and T. R. Putra, "Peringatan Dini Banjir Berbasis Internet Of Things (IoT) dan Telegram," *Infotek: Jurnal Informatika dan Teknologi*, vol. 7, no. 1, pp. 142–150, 2024.

[13] H. S. Millah and S. Hidayatulloh, "Implementasi IoT Pada Rumah Budidaya Jamur Tiram Putih," *eProsiding Teknik Informatika (PROTEKTIF)*, vol. 5, no. 1, pp. 168–176, 2024.

[14] S. D. M. Panjaitan, "Prototype Pengendalian Lampu Jarak Jauh dengan Jaringan Internet Berbasis Internet of Things (IoT) Menggunakan Rasberry Pi 3," *Jurnal Pendidikan Sains dan Komputer*, vol. 2, no. 2, pp. 361–365, 2022.

[15] I. B. M. L. Pradirta, I. N. Piarsa, and I. P. A. Dharmaadi, "Sistem Pendeteksi Banjir dan Badai Angin serta Monitoring Cuaca Berbasis Internet of Things," *Jurnal Teknologi Informasi dan Ilmu Komputer (JTIIK)*, vol. 9, no. 5, 2022.

Eric Alfonsius: *Corresponding Author