Design and Development of a Monitoring and Controlling System for Automatic Watering and Filling in Fungi House's Internet of Things-Based Mushroom Cultivation

Eko Supriyanto[1], Rizkha Ajeng Rochmatika[2], Cantika Cakhya Oktaviani[3], Syauqi Fajar Lukita[4], Abu Hasan[5], Hutama Arif Bramantyo[6], Tri Raharjo Yudantoro[7]

[1,2,3,4,5,6,7] *Electrical Engineering Departement, Politeknik Negeri Semarang, Central Java, 50275, Indonesia*

Abstract -- Temperature and humidity are aspects that need to be considered in cultivating oyster mushrooms. Previously, Fungi House in Genting Village, Semarang, implemented an automatic temperature, humidity, and watering monitoring system, but manually filled the water. This new system's design and development aim to simplify the monitoring and control of temperature, humidity, and water level for managers. Managers determined temperature, humidity, and water level thresholds via the web page. This system used the agile scrum method. The test results showed that the temperature measurement accuracy was 96.85%, humidity 99.35%, and water level 98.99%. With this system, the quality of baglog (mushroom growing medium) increased by 4.62%, while dead baglog decreased by 99.01%. Black box testing demonstrates that all features perform well in web testing. In the load activity test, with low bandwidth (6.71 Mbps), the average load time was 1.32 seconds, and with high bandwidth (37.15 Mbps), it was 0.878 seconds. These two conditions indicate excellent system performance and provide optimal user experience.

Keywords-- Oyster Mushroom, Internet of Things, Monitoring, Controlling, SHT31, JSN-SR04T, Web, TTGO T-CALL ESP32

1. Introduction

The public's demand for data processing that is effortless is on the rise in all sectors of the workforce. Currently, data processing extensively employs information technology due to its numerous benefits, such as increased efficiency, accuracy, and speed compared to manual methods [5]. The web interface is currently one of Indonesia's technologies.

The climate of Indonesia is tropical, with two distinct seasons: dry and rainy. Oyster mushrooms are among the numerous varieties of mushrooms that thrive in this climate. People are fond of mushrooms due to their nutritional value and delectable flavor. Oyster mushrooms are also simple to cultivate, and they are highly nutritious. In the growth process, media growth is a critical factor [7].

Oyster mushrooms thrive in temperatures ranging from 16 to 30 °C and humidity levels of 80 to 95% [11]. To cultivate mushrooms, you need a kumbung, a container designed to store baglog (a mushroom growing medium) and facilitate mushroom growth. The structure must maintain the fungus's temperature and humidity [9].

Previously, a fungus house in Genting Village, Jambu District, Semarang Regency, cultivated oyster mushrooms using a temperature and humidity monitoring system and automatic watering. However, the manual filling of the reservoir necessitates frequent visits by mushroom farmers to the fungus house during watering to replenish it. Based on this, we designed a monitoring and controlling system for automatic watering and filling in the function house using the internet of things. The goal of this system was to assist farmers in monitoring and regulating the temperature and humidity in the fungus house.

2. Literature Review

In this investigation, the author incorporated references from numerous prior studies. Ayu Yuliani and Tomy Afriyanto, telecommunications engineering students at Semarang State Polytechnic, completed the final assignment "Temperature Monitoring System in the Mushroom Baglog Sterilization Process with Android-Based ESP32 at Fungi House, Genting Village, Semarang Regency" on October 14, 2020. This Android system monitors the temperature of the baglog sterilization process [12].

The second reference is the final assignment titled "Analysis of the Temperature and Humidity Monitoring System and Automatic Watering in Mushroom Cultivation with ESP32 at the Fungi House, Semarang Regency" by Wahyu Cahyaningtyas, a student at the Semarang State Polytechnic, on September 7, 2021.. The design of this system includes automatic watering of the fungus house and monitoring of temperature and humidity. The system will implement irrigation if the temperature and humidity levels exceed predetermined thresholds [3].

Gisnaya Faridatul Avisyah, a telecommunications engineering student at Semarang State Polytechnic,

conducted the final assignment, "Analysis of the Temperature and Humidity Monitoring System and Automatic Watering in Android-Based Mushroom Cultivation in Fungi House, Semarang Regency", on September 7, 2021. This system displays the temperature and humidity data in the fungus house through a smartphone application [1].

Based on these references, the author developed a system for automatic watering and filling of water in oyster mushroom cultivation in the fungus house, utilizing the ESP32 TTGO T-CALL microcontroller, the SHT-31 sensor for temperature and humidity detection, and the JSN-sr04t ultrasonic sensor to detect the water level in the water reservoir. This system employs a website as a means of monitoring and controlling.

3. Research Methods

This final assignment employed the agile method, which was adaptable to modifications. Agile underscored the importance of being adaptable to change, as the initial plans may not be suitable for execution. The Scrum method, which integrates agile into project and event development, is one approach to managing this change [10].

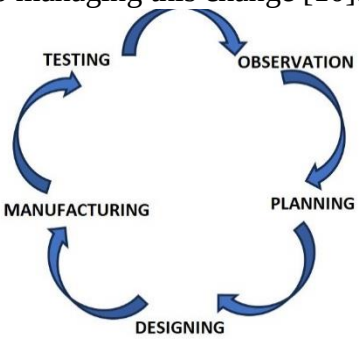

Fig 1. Research Method

Figure 1 presents that this research method comprises five stages: observation, planning, design, manufacturing, and testing. The observations were made at Genting Village's fungus house in Genting Village was the site of the observations.During the planning stage, we compile the necessary components, devices, and software systems for the system's development.

3.1 Observation

We conducted the observation stage at the function house in Genting Village, Semarang Regency. This observation aimed to pinpoint the shortcomings of the previously established system. In addition to that, the observation aimed to identify additional requirements for a fungus house.

3.2 Planning

The planning stage involved the compilation of the system's requirements, including hardware, component requirements, and software system requirements.

3.3 System Design

3.3.1 Tool System Design

The tool's design served as its description.

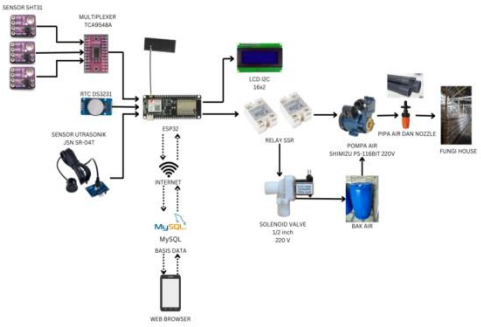

Fig 2. System Design

Figure 2 presents that the ESP32 TTGO microcontroller serves as the control center for this system. The SHT-31 sensor collects temperature and humidity data, the JSN-SR04T sensor collects water level data, and the RTC DS3231 sensor collects real-time data. The microcontroller processes this data. The SIM800 module allows the ESP32 TTGO to access the internet, and the HTTP protocol transmits data from the three sensors to a database. The I2C 16X2 LCD also displays temperature, humidity, and water level measurements for monitoring the fungus house.

In addition, the microcontroller is connected to two relays: Relay 1 regulates the solenoid valve for automatic water filling, while Relay 2 regulates the water pump for automatic watering, based on temperature and humidity data. These two control features compare sensor data with thresholds. The HTTP protocol retrieves thresholds from the database, and the web interface modifies them to control the website. Utilizing the Application Programming Interface (API), it is feasible to access the database.

The system's implementation is the next step. This encompasses the installation of tools, pipes, and the establishment of the overall layout. We install the water pump and solenoid valve near a water source, and position the system near a power source. A pipe connects the sprayer nozzle to the pump, positioning it on the right, middle, and left sides of

the two mushroom baglog racks. Figure 3 presents the system layout design.

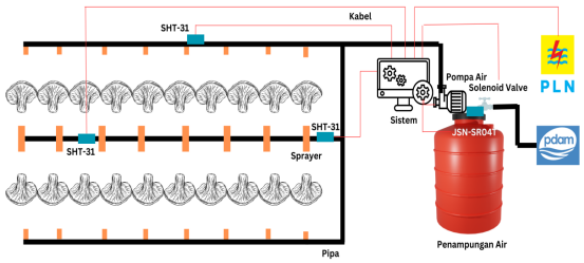

Fig 3. System Location

3.3.2 Web System Design

We completed this stage by identifying the generated web image. The microcontroller later sent all data to the database via the internet for display on a web page. We carried out three stages, which included designing use case diagrams, activity diagrams, and database tables. We used use case diagrams to illustrate and represent the interactions between users and the system. We achieved this by identifying the user and the tasks that both the user and the system could perform. Activity diagrams describe the processes that occur in a system. This system's activity diagram consisted of login activity, monitoring activity, and controlling activity. The system stored its data in a database. Tables 1 to 3 display the database table's design.

Table 1 Login Data

Data table "login"	
Fields	**Type**
Id	int(11)
Username	varchar(20)
Passwords	varchar(20)

Table 2 Monitoring Data

"monitoring" data table	
Fields	**Type**
Id	int(100)
Temperature	Float
Humidity	int(10)
High water	Float
send_date	Datetime
received_date	Datetime

Table 3 Controlling Data

"controlling" data table	
Fields	**Type**
Id	int(10)
temperature_limit	int(10)
humidity_limit	int(10)
min_water height	int(10)
max_water height	int(10)
Time	Datetime

There were 3 tables, namely the table for logging in, namely "login," the table for detection results by sensors, namely "monitoring," and the table for controlling the threshold, namely "controlling." Each table's ID field contained a primary key with an integer data type and an auto-increment with a value that will change automatically if additional rows occur. We used the login table as a database for user accounts, enabling them to log in to web pages. The web will display data from the monitoring and control tables. Additionally, the values in the data table within the controlling table may be modified on the web page.

3.4 System Creation

The creation stage contained the steps for creating the system. Making this system included the stages of designing the user interface, website, database, and tools.

1. Creating a user interface design involved detailing the website's appearance for users to access later.
2. The Visual Studio Code application created website pages using the HTML, CSS, and PHP programming languages. The program was then hosted on the Niagahoster web server.
3. To create a database and to create a table in PHPMyAdmin, we used this database to store system data.
4. We created the tool by assembling all the components used in the system. Next, we constructed a box to safeguard the system circuit.

3.5 Test Design

The testing stage involved analyzing the system's performance. Several tests are carried out in this stage, namely as follows:

1. *Black Box Testing* as a testing method for analyzing the functionality of software or applications. This black box testing tests the appearance of an application from the user's perspective without knowing the code structure [6].
2. *Load Activity Testing* as the testing method. We conducted this test to ascertain the duration required for the browser to display all website pages in response to user requests. We conducted this test using two different internet

speeds. Internet speed was known through speedtests. We tested the system by utilizing the website's inspect network feature. Based on the comparison of each internet speed and network inspection results, load time per activity values were obtained [8].

3. *QoS (Quality of Service)*. The method measured the quality of service or device performance. Some of the parameters used are delay and jitter [2].
4. *Delay*. It was the time required for data to get from origin to destination. The following is the formula for calculating delay.

$$\text{Delay} = \text{Time Arrival} - \text{Time Departure}$$

$$\text{Average Delay} = \frac{\text{Total Delay}}{\text{Total Packets Received}}$$

Table 4 Delay Categories [4]

Latency Category	Large Delay (ms)
Very good	<150
Good	150-300
Currently	300-450
Bad	>450

5. Jitter. It was a variation of delay in data transmission. The following is the formula for Jitter calculations.

$$\text{Total Variance of Delay} = \text{Delay} - (\text{Average delay})$$

$$\text{Jitter} = \frac{\text{Total Variance of Delay}}{\text{Total Packets Received}}$$

6. Comparative testing between sensors and measuring instruments. This test was based on a comparison between temperature, humidity, water level sensors and measuring instruments. This comparison produced a sensor accuracy value. The following is the formula for calculating sensor accuracy.

$$\text{Difference} = \text{measured value from the intrument} - \text{measured value from the sensor}$$

$$\text{Percentage of Error} = \frac{(\text{measured value from the intrumen} - \text{measured value from the sensor})}{\text{measured value from the intrumen}} \times 100\%$$

$$\text{Percentage of Accuracy} = 100 - \text{Percentage of Error}$$

7. *Comparative testing* of the condition of the baglog. A comparison of the condition of the baglog was carried out before and after the tool was applied to the mushroom kumbung. There were 2 states for baglog, namely baglog in good condition and dead. Figure 4 presents a baglog in good condition. Meanwhile, Figure 5 presents a baglog with a dead condition.

Fig 4. Baglog condition is good

Fig 5. Dead Baglog Condition

Figure 4 presents that good baglog conditions are growing white oyster mushrooms. Meanwhile, the condition of the baglog is dead as in Figure 5, if the baglog does not grow oyster mushrooms and the baglog is black. The formula used to determine the percentage of baglog conditions is as follows.

$$\text{Percentage of Baglog Condition} = \frac{\text{Number of Baglog Condition}}{\text{Total Baglog Samples}} x\ 100\ \%$$

$$\text{Decrease} = \frac{\text{Initial} - \text{Final}}{\text{Initial}} x\ 100\ \%$$

$$\text{Increase} = \frac{\text{Final} - \text{Initial}}{\text{Initiall}} x\ 100\ \%$$

4. Results and Discussion

4.1 Tool Design Results

Figures 6 and 7 show the results of designing and assembling this tool as a system node.

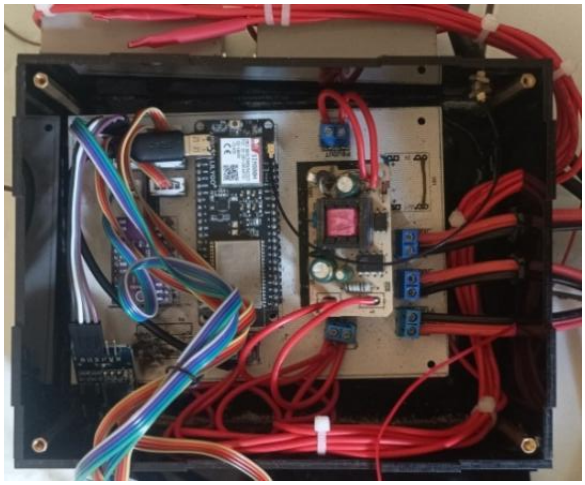

Fig 6. System Node 1

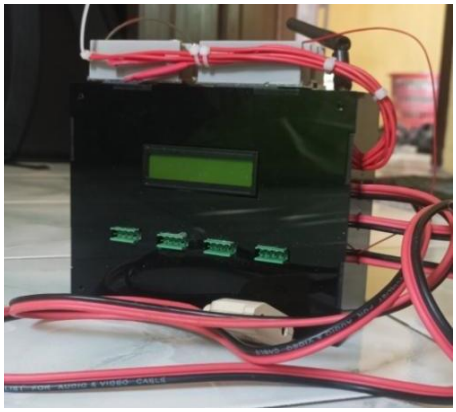

Fig 7. System Node 2

Figure 8 presents a picture of the water reservoir, water pump, ultrasonic sensor, and solenoid valve.

Fig 8. Location of Solenoid Valve and Water Pump

Figure 9 presents a picture of the location of the pipe and sprayer nozzle.

Fig 9. Location of Pipe and Sprayer Nozzle

Figures 10 and 11 present images of the location of the temperature and humidity sensors, as well as the water level sensor.

Fig 10. Location of Temperature and Humidity Sensors

Fig 11. Location of Temperature and Humidity Sensors

Figure 12 presents a picture of the system location.

Fig 12. System Location

4.2 Web Page Implementation

The interface is the display that appears on the user's smartphone web browser after entering the website. The purpose of creating this interface is to

serve as a monitoring and control medium for mushroom cultivation. The following are the results of implementing the monitoring and controlling system web interface for oyster mushroom cultivation.

The login page is the first page that will appear when a user enters the monitoring and controlling website for mushroom cultivation. The user must fill in a form on this page with their username and password. This aims to enable users to enter the dashboard page after pressing the submit button. Figure 13 presents the web implementation of the login page interface.

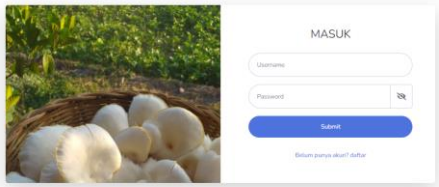

Fig 13. Login Page

The registration page is a page that appears after the user presses the register button on the login page. The user needs to fill out a form on this page with their username and password. The user must press the submit button after filling out the form to register their username and password. This aims to create an account so that users can log in. Figure 14 presents the web-based implementation of the register page interface.

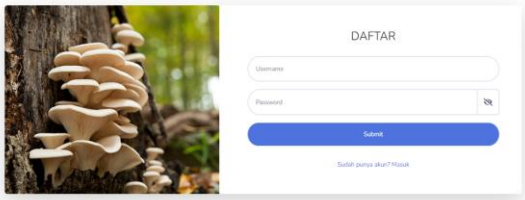

Fig 14. Register page

The dashboard page appears once the user successfully logs in. This page contains real-time values for temperature, humidity, and water level. Aside from that, it also contains graphs of temperature, humidity, and water level values within 1 day according to the user's chosen time. By pressing the select date section, you can determine the time for the graph to appear. Then, choose the date, month, and year. After that, press the select button. Figures 15 and 16 present the web implementation of the dashboard page interface.

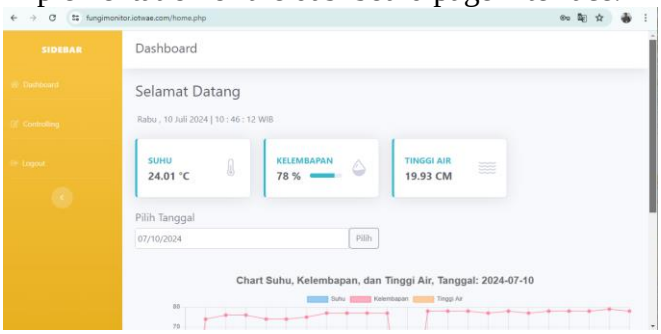

Fig 15. Dashboard page

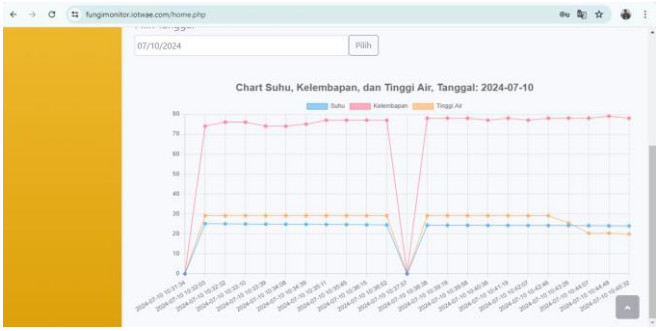

Fig 16. Graphic display

The controlling page is the page that will appear if the user presses the controlling menu in the sidebar, the controlling page will appear. This page contains threshold tables for temperature limits, humidity limits, minimum water height, and maximum water height. The value of the threshold can be changed by pressing the pencil icon in the table. Figure 17 presents the web-based implementation of the controlling page interface.

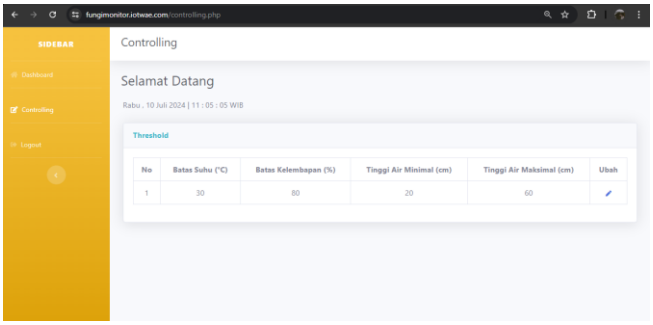

Fig 17. Controlling page

The change threshold page is a page that will appear after the user presses the change button on the table on the controlling page. This page contains a form for changing the threshold values in the table, such as temperature, humidity, and minimum and maximum water height limits. Then, to save the changes, press the change button. Figure 18 presents the implementation of the web page interface for changing the threshold.

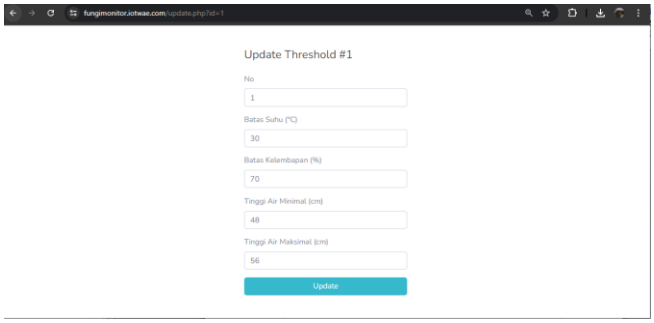

Fig 18. Threshold Update Page

4.3 Black Box Testing Results

Black box testing analyzes software functionality without knowledge of the software's code structure. We carry out testing by determining appropriate input and output. According to the test results, the website functions as specified, and the test was successful.

4.4 Load Activity Testing Results

We use load activity testing to determine the time required to move from one activity to another. We conducted this test using two types of bandwidth: low and high bandwidth. The bandwidth value can be found via speedtest.net, pictures 19 and 20 show the bandwidth measurement results.

Fig 19. Low Bandwidth

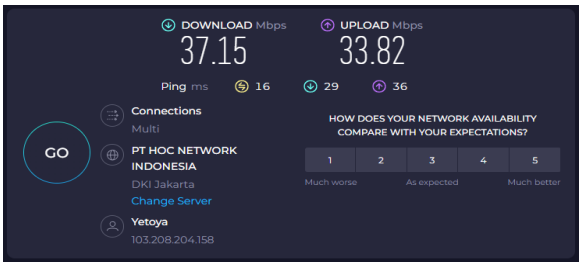

Fig 20. High Bandwidth

The measurement results show two different bandwidths. The low bandwidth used is 6.74 Mbps, and the high bandwidth is 37.15 Mbps. Table 5 carries out load activity testing based on these two bandwidths.

Table 5 Load Activity Testing Results

No	*Bandwidth*	Objek	Waktu (s)
1	6.74 Mbps	*Login*	1,69
		Register	1,02
		Register akun	3,27
		Dashboard	1,53
		Controlling	1.11
		Form threshold	0,563
		Update threshold	0,822
		Logout	0,807
		Rata-rata *load* per *activity*	**1,35**
2	37.15 Mbps	*Login*	0,723
		Register	0,874
		Register akun	2,13
		Dashboard	0,743
		Controlling	0,568
		Form threshold	0,559
		Update threshold	0,639
		Logout	0,754
		Rata-rata *load time* per *activity*	**0,878**

Table 5 presents the test results. When the bandwidth is 6.74 Mbps, the average load time per activity is 1.35 s. Meanwhile, with a bandwidth of 37.15 Mbps, the average load time per activity is 0.878 s. The activity with the lowest or best load time is in the threshold form, namely 0.559 s. Meanwhile, the activity with the highest or worst load time, namely 3.27 S, is in the account register. Based on ETSI standards, both conditions are in the excellent category. Thus, the test results show that the overall user experience remains good, both at low and high bandwidth, in accordance with ETSI standards.

4.5 QoS Testing Results

4.5.1 *Delay*

Delay testing is carried out by looking for the difference between the data sending time on the microcontroller, which is known using RTC time data and the database table receiving time.The following is a table of delay calculation results with data sampling time on July 10 2024 in table 6.

Table 6 Delay Calculation Results

No	Delivery Time	Time Received	*Delay*(ms)
1	11:04:35	11:06:08	93,000
2	11:05:18	11:07:03	105,000
3	11:06:12	11:07:54	102,000
4	11:07:04	11:08:38	94,000
5	11:07:48	11:09:16	88,000
6	11:08:26	11:10:03	97,000
7	11:09:12	11:10:45	93,000
8	11:09:54	11:11:25	91,000

No	Delivery Time	Time Received	*Delay*(ms)
9	11:10:34	11:12:17	103,000
10	11:11:27	11:13:11	104,000
11	11:12:21	11:14:02	101,000
12	11:13:11	11:14:44	93,000
13	11:13:55	11:15:30	95,000
14	11:14:41	11:16:11	90,000
15	11:15:22	11:17:03	101,000
16	11:16:13	11:17:46	93,000
17	11:16:56	11:18:41	105,000
18	11:17:51	11:19:24	93,000
19	11:18:34	11:20:04	90,000
20	11:19:14	11:20:47	93,000
21	11:19:58	11:21:27	89,000
22	11:20:37	11:22:11	94,000
23	11:21:20	11:22:58	98,000
24	11:22:08	11:23:40	92,000
25	11:22:50	11:25:20	150,000
26	11:24:29	11:26:07	98,000
27	11:25:17	11:27:01	104,000
28	11:26:11	11:27:39	88,000
29	11:26:49	11:28:17	88,000
30	11:27:26	11:28:58	92,000
31	11:28:08	11:29:39	91,000
32	11:28:48	11:30:26	98,000
33	11:29:35	11:31:09	94,000
34	11:30:18	11:31:51	93,000
35	11:31:01	11:32:30	89,000
36	11:31:40	11:33:14	94,000
37	11:32:24	11:34:04	100,000
38	11:33:13	11:34:54	101,000
39	11:34:03	11:35:42	99,000
40	11:34:51	11:36:18	87,000
41	11:35:27	11:36:56	89,000
42	11:36:05	11:37:37	92,000
43	11:36:46	11:38:20	94,000
44	11:37:29	11:38:56	87,000
45	11:38:05	11:39:34	89,000
46	11:38:43	11:40:12	89,000
47	11:39:22	11:40:49	87,000
48	11:39:58	11:41:31	93,000
49	11:40:41	11:42:13	92,000
50	11:41:23	11:42:54	92,000
Average			**95,540**

We obtained an average delay of 95,540 ms from the 50 data samples used. The utilization of the GPRS network contributes to this unfavorable result. However, the delay value does not really affect system performance.

4.5.2 *Jitter*

Jitter is a variation of delay in data transmission.Table 7 is the result of jitter calculations based on the data in table 6.

Table 7 Jitter Calculation Results

No	Total Delay Variation	Total Packages Received	*Jitter*(ms)
1	2,540	50	50.8
2	9,460	50	189.2
3	6460	50	129.2
4	1,540	50	30.8
5	7,540	50	150.8
6	1,460	50	29.2
7	2,540	50	50.8
8	4,540	50	90.8
9	7,460	50	149.2
10	8,460	50	169.2
11	5,460	50	109.2
12	2,540	50	50.8
13	540	50	10.8
14	5,540	50	110.8
15	5,460	50	109.,2
16	2,540	50	50.8
17	9,460	50	189.2
18	2,540	50	50.8
19	5,540	50	110.8
20	2,540	50	50.8
21	6,540	50	130.8
22	1,540	50	30.8
23	2,460	50	49.2
24	3,540	50	70.8
25	54,460	50	1,089.2
26	2,460	50	49.2
27	8,460	50	169.2
28	7,540	50	150.8
29	7,540	50	150.8
30	3,540	50	70.8
31	4,540	50	90.8
32	2,460	50	49.2
33	1,540	50	30.8
34	2,540	50	50.8
35	6,540	50	130.8
36	1,540	50	30.8
37	4,460	50	89.2
38	5,460	50	109.2
39	3,460	50	69.2
40	8,540	50	170.8
41	6,540	50	130.8
42	3,540	50	70.8
43	1,540	50	30.8

No	Total Delay Variation	Total Packages Received	*Jitter*(ms)
44	8,540	50	170.8
45	6,540	50	130.8
46	6,540	50	130.8
47	8,540	50	170.8
48	2,540	50	50.8
49	3,540	50	70.8
50	3,540	50	70.8
Average			**113,888**

We obtained an average jitter of 113,888 ms from the 50 data samples used. This value is in the medium category; this happens because the delay value of the 50 sample data points is not too far from the average delay value. This indicates that the delay in data transmission tends to be stable.

4.5.3 Temperature Detection Test Results

This is a comparison of the SHT-31 sensor's temperature detection results with measuring instruments. Table 8 presents a table of temperature measurement results over a period of 3 days using sensors and thermometers.

Table 8 Temperature Measurement Results

No	Tanggal	Waktu	Sensor (°C)	Meter (°C)	Selisih (°C))	Error (%)	Akurasi (%)
1	03/07/2024	11:56	25,43	26,40	0,97	3,67	96,33
		14:09	25,01	25,70	0,69	2,68	97,32
		16:43	24,97	25,60	0,63	2,46	97,54
2	07/07/ 2024	11:24	22,31	23,20	0,89	3,84	96,16
		12:41	22,42	23,40	0,98	4,19	95,81
		13:36	22,39	23,20	0,81	3,49	96,51
3	09/07/2024	11:46	24,27	24,70	0,43	1,74	98,26
		13:56	23,76	23,00	0,76	3,30	96,70
		15:15	23,76	24,50	0,74	3,02	96,98
Rata-rata					**0,77**	**3,15**	**96,85**

Table 8 presents that the average percentage error from sensor readings is 3.15%, while the accuracy is 96.85%. Figure 21 is a graph of the average error and accuracy of the tool.

Average Error and Accuracy of Tools

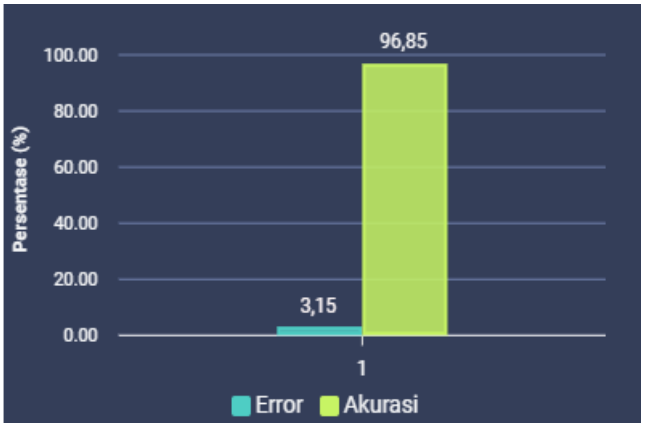

Fig 21. Average Temperature Sensor Error and Accuracy

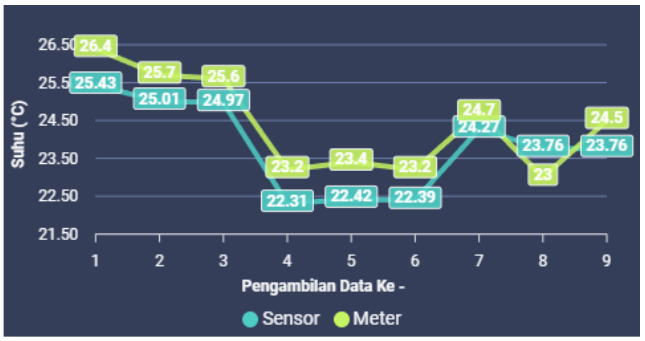

Fig 22. Comparison Results of Sensors and Thermometers

Figure 22 presents that the graphs of sensor readings and measuring instrument readings tend to coincide. This means the sensor and measuring instrument generally give the same temperature reading.

4.5.4 Moisture Detection Test Results

This is a comparison of the humidity readings obtained by the SHT-31 sensor and measuring instruments. Table 9 presents a table of humidity measurement results obtained over a three-day period using sensors and measuring instruments.

Table 9 Humidity Measurement Results

No	Tanggal	Waktu	Sensor (%)	Meter (%)	Selisih (%)	Error (%)	Akurasi (%)
1	03/07/ 2024	11:56	82,57	83	0,43	0,52	99,48
		14:09	84,89	84	0,84	1	99
		16:43	84,40	85	0,60	0,71	99,29
2	07/07/2024	11:24	92,40	93	0,60	0,64	99,36
		12:41	93,13	93	0,13	0,14	99,86
		13:36	93,56	93	0,56	0,60	99,39
3	09/07/2024	11:46	79,55	79	0,55	0,70	99,30
		13:56	87,13	88	0,87	0,99	99,01
		15:15	86,88	87	0,12	0,13	99,86
Rata-rata					**0,52**	**0,60**	**99,39**

Table 9 presents that the average percentage error from sensor readings is 0.60%, while the accuracy

is 99.39%. Figure 23 presents a graph of the average error and accuracy of the tool.

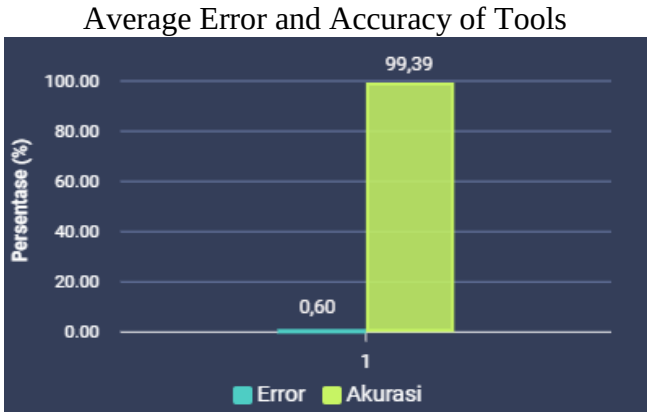

Fig 23. Average Humidity Sensor Error and Accuracy

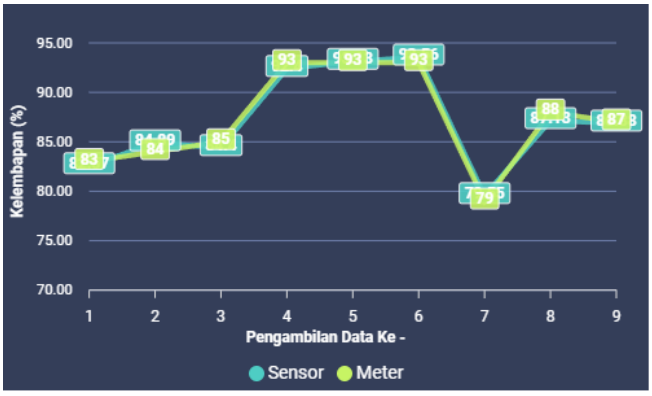

Fig 24. Comparison Results of Humidity Sensor and Hygrometer

Figure 24 presents that the graph of sensor readings and the graph of measuring instrument readings tend to coincide. This indicates that the humidity reading between the sensor and measuring instrument has no significant difference or tends to be the same.

4.5.5 Water Level Detection Test Results

This is a comparison of the JSN-sr04t sensor's water level readings with measuring instruments. Table 10 presents a draft table of measurement results from water level sensors and measuring instruments.

Table 10 Water Level Detection Results

No	Sensor (cm)	Meter (cm)	Selisih (cm)	Error (%)	Akurasi (%)
1	44,13	45	0,87	1,93	98,07
2	39,80	40	0,20	0,50	99,50
3	47,36	48	0,64	1,33	98,67
4	30,29	30	0,29	0,97	99,03
5	34,82	35	0,18	0,51	99,49
6	25,00	25	0,00	0,00	100
7	42,67	42	0,67	1,59	98,41
8	49,36	50	0,64	1,28	98,72
Rata-Rata			**0,44**	**1,01**	**98,99**

Table 10 presents that the average percentage error from sensor readings is 1.01%, while the accuracy is 98.99%. Figure 25 presents a graph of the average error and accuracy of the tool.

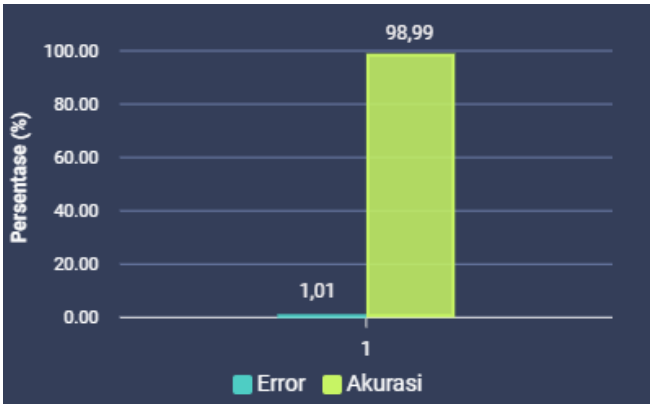

Fig 25. Average Error and Accuracy of the Water Height Sensor

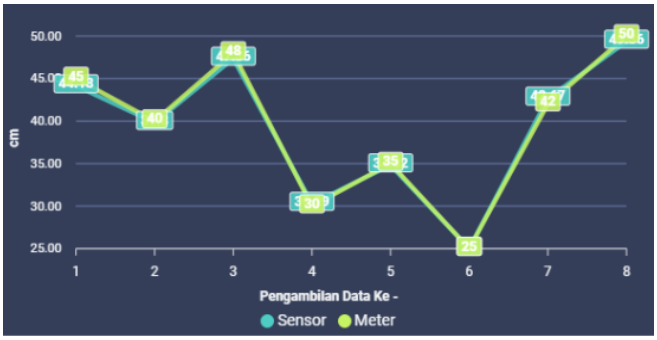

Fig 26. Comparison of Water Level Sensor and Meter

Figure 26 presents that the graphs of sensor readings and the graphs of measuring instrument readings tend to coincide. This indicates that the water level reading between the sensor and measuring instrument does not have a significant difference or tends to be the same.

4.5.6 Results of Comparison of Baglog Conditions

In this test, we compared the baglog conditions before and after the system's application. We tested two baglog conditions: the good condition and the

dead condition. Table 11 presents the comparison results.

Table 11 Comparison of Baglog Conditions

No	Kondisi Baglog	Sebelum Pemasangan Sistem	Setelah Pemasangan Sistem
1	Baik	2.078	2.174
2	Mati	136	2

Based on table 11, the graphs in figures 27 and 28 are obtained.

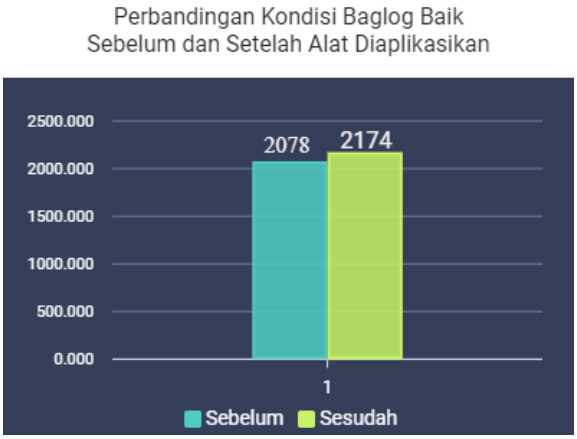

Fig 27. Comparison Chart of Good Baglog Condition

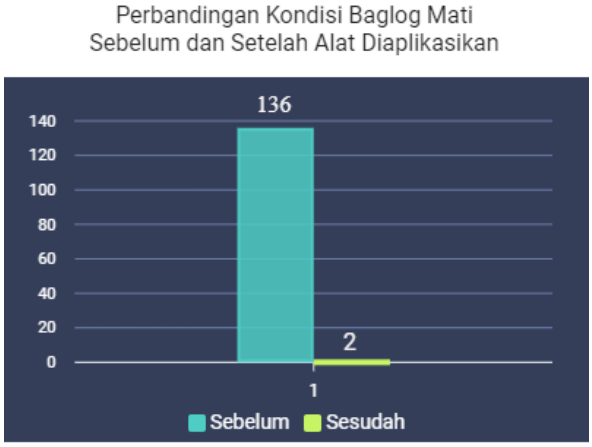

Fig 28. Comparison Chart of Dead Baglog Conditions

Before system installation, of the total 2,214 baglogs, 2,078 were in excellent condition and 136 were dead. After installing the system, out of a total of 2,176 baglogs, 2,174 were in excellent condition and 2 were dead. The difference between a good baglog condition before and after installation is 96 pieces. According to the percentage increase and decrease formula in Chapter 3, the condition of good baglogs rose by 4.62%, while the condition of dead baglogs fell by 99.01%.

5. Conclusion

We draw conclusions on the monitoring and controlling system for automatic watering and filling in internet of things-based mushroom cultivation at the fungus house based on the results of observation, design, manufacture, system testing, and data analysis.

1. By creating a monitoring and controlling system for automatic watering and filling, it makes farmers' work easier in maintaining temperature and humidity in the fungus house, resulting in more efficient mushroom cultivation activities.
2. Black box testing shows the website is functioning as planned. Load activity testing with a bandwidth of 6.71 Mbps has an average load time of 1.32 seconds, while with a bandwidth of 37.15 Mbps, the average load time is 0.878 seconds, which is within the standard. The SHT-31 sensor has an average temperature error and accuracy of 3.15% and 96.85%, as well as a humidity error of 0.60% and 99.39%, indicating good quality. The JSN-SR04T sensor has an average difference of 0.44 cm with the measuring instrument, with an error of 1.01% and an accuracy of 98.99%, so it is considered feasible.
3. After installation of the system, the test results for good baglog conditions increased by 4.62% from 2078 to 2174, while dead baglog conditions decreased by 99.01% from 136 to 2.

References

[1] Avisyah, GF (2021). "Analysis of the Temperature and Humidity Monitoring System and Automatic Watering in Android-Based Mushroom Cultivation at Fungi House, Semarang Regency." Semarang State Polytechnic.

[2] Budiman, A., Duskarnaen, MF, & Ajie, H. (2020). "ANALYSIS OF QUALITY OF SERVICE (QOS) ON THE INTERNET NETWORK OF SMK NEGERI 7 JAKARTA." Pinter Journal, 4.

[3] Cahyaningtyas, W. (2021). "Analysis of the Temperature and Humidity Monitoring System and Automatic Watering in Android-Based Mushroom Cultivation at Fungi House, Semarang Regency." Semarang State Polytechnic.

[4] ETSI. (1999). "Telecommunications and Internet Protocol Harmonization Over Networks (TIPHON); General aspects of

Quality of Service (QoS).” Etsi Tr 101 329 V2.1.1, 1, 1–37.
[5] Hariyanto, A., Sugiharti, E., & Arifudin, R. (2019). "Android Based Labschool Academic Information System Mobile Application at Semarang State University." UNNES Journal of Mathematics, 12.
[6] Isa Alfaris, HB, Anam, C., & Masy'an, A. (2013). "IMPLEMENTATION OF BLACK BOX TESTING ON A WEB-BASED SANTRI REGISTRATION INFORMATION SYSTEM USING PHP AND MYSQL." Journal of Science And Technology, 6, 27–28.
[7] Karangsewu, D. (2020, March 18). “Oyster Mushroom Cultivation.” https://karangsewu-kulonprogo.desa.id/index.php/article/2020/3/18/budidaya-jamur-tiram.
[8] Meier, J.D., Farre, C., Bansode, P., Barber, S., & Rea, D. (2007). “Performance Testing Guidance for Web Applications. Microsoft Corporation.”
[9] Sujoko, A., Lutfi, M., & Purnomo, D. (2015). "Study on Sterilization of Growing Media for White Oyster Mushrooms (Pleurotus Ostreatus (L) Fries) Using a Baglog Steamer." Journal of Tropical Agricultural Engineering and Biosystems, 3, 303.
[10] Umam, MS (2019, March 12). “Agile and Scrum | Introduction." Https://Medium.Com/DotIntern/Agile-Dan-Scrum-Penetalan Ed1bb461a5bc.
[11] Waluyo, S., Wahyono, RE, Lanya, B., & Telaumbanua, M. (2018). "Automatically Controlling Temperature and Humidity in Oyster Mushroom Mushrooms (Pleurotus sp) Based on a Microcontroller." Agritech, 282.
[12] Yuliani, A., & Afriyanto, T. (2020). "Temperature Monitoring System in the Mushroom Baglog Sterilization Process with Android-Based ESP32 at Fungi House, Genting Village, Semarang Regency." Semarang State Polytechnic.