MATICS: Jurnal Ilmu Komputer dan Teknologi Informasi ISSN : 1978-161X(p); 2477-2550(e)
(*Journal of Computer Science and Information Technology*) *Volume* 15, No. 1(2023), pp 29-42
Received: July 23th 2023; **Accepted**: August 4th 2023; **Avalaible Online**: August 14th 2023

Design of Water Monitoring System in Aquaponics Based on Arduino Nano and Raspberry Pi

Nyoman Wira Prasetya, Arya Rizky Imansyah Harahap, Fadhli Aulady, Inayah Wulandari

Abstract— **The aquaponic system is an agricultural technology that can provide a lot of results in a limited space by recirculating water and nutrients from the symbiosis created between fish farming and hydroponic plant cultivation. To maximize the nutritional needs of water that will be supplied to plants and maintain good water conditions for fish life, it is necessary to check several parameters that can be measured in water periodically in fishponds in an aquaponic system so that farmers can provide appropriate actions in managing the system. aquaponic farming. Based on this background, a system was designed using the Internet of Things concept that can monitor water conditions in aquaponic fishponds, store water parameter data in a database, then data can be monitored via the Website. The system is designed using the Arduino Nano microcontroller board as a data processor which is equipped with the ESP8266-07 wifi module which is connected via a WiFi Local Area Network (LAN) network and connected to the Raspberry Pi 3 Model B as a gateway to the database using the MQ Telemetry Transport Protocol (MQTT). The water parameters that are measured in the designed system are: water temperature, water turbidity, dissolved solids, dissolved oxygen, and water acidity (pH). The results of the designed system show that this system can run as expected so that it can facilitate monitoring of water in aquaponic farming systems.**

Index Terms—**Aquaponic, Water monitoring, Arduino, MQTT, Rasberry.**

I. Introduction

Currently, food security is one of the issue that we face together [1]. With increasing food insecurity, there is a need to develop innovative farming methods to complement current food production techniques. Apart from innovation and agricultural development over the years, until 2021 the prevalence value of insufficient food consumption in Indonesia is still worth 8.49% where this value has increased compared to the previous year which was 8.34% (Central Bureau of Statistics, 2022). This food and agriculture problem has been exacerbated by the impact of the recent Covid-19 pandemic [2]. Apart from being unable to provide reliable food for the world's population, the global food system has also caused environmental degradation. Industrialized agricultural can cause pollution, climate change, public health, and many environmental impacts [3]. One way to overcome this problem is to incorporate sustainable agriculture into urban communities, one of which is in the form of aquaponics.

Aquaponics is an innovative and widely studied urban farming technology that uses the symbiotic relationship between fish and plants to increase food production [4]. The aquaponic system grows fish and plants simultaneously where this system combines fish farming techniques with hydroponic plant cultivation [5]. Aquaponic systems cultivate fish and plants in a closed, sustainable environment. With the application of aquaponics which recirculates water and nutrients, aquaponics becomes a zero waste and water efficient system [6]. Unlike soil-based agriculture, which requires copious amounts of land, aquaponics can produce high yields in confined urban environments because it can be implemented in narrow yards [7]. Besides that, the use of aquaponics carries more with the concept of sustainable agriculture. namely a concept in which agriculture is carried out by prioritizing a balance between human needs with environmental protection and the welfare of its farmers [8].

Although aquaponics has many benefits, this system has its own challenges to implement. This is because the user needs to know the quality of the water used in aquaponics. Several water quality parameters that need to be known are temperature, dissolved oxygen, and degree of acidity [9]. If these parameters are not known, then this system cannot run optimally [10]. Therefore, it is necessary to have a system that can facilitate the implementation of aquaponics in terms of monitoring these water quality parameters.

There have been several studies about aquaponic and its implementation. Alfia and Mailoa designed a smart aquaponic system that can be controlled remotely by utilizing Telegram chatbot [11], [12]. Users can monitor water quality by sending command messages to chatbots. Then the chatbot will provide feedback

Manuscript received July 23, 2023. This work was supported in part by Informatics Engineering Department of Maulana Malik Ibrahim Islamic State University.

NW Prasetya Author is with the Computer Science Department, School of Computer Science, Bina Nusantara University, Indonesia, email nyoman.wira@binus.ac.id

Arya Rizky Imansyah Harahap Author was with Brawijaya University, Malang, Indonesia, email aryarimansyah@gmail.com

Fadhli Aulady Author was with Brawijaya University, Malang, Indonesia, email fadhlideede3@gmail.com

Inayah Wulandari Author was with Brawijaya University, Malang, Indonesia, email inayahwulan2410@gmail.com

containing the parameter values of the water quality at that time. Burlian developed an IoT based aquaponic system using Arduino as its microcontroller [13]. This system will pump water automatically with a time schedule that has been programmed into the system. Iskandar Nasution also developed IoT based system to monitor aquaponic water quality [14]. In his program, the system will receive input in the form of temperature and water pH values. Then the measurement results will be displayed on the mobile application. Besides that, the mobile application can also be used to set schedules for feeding fish and water pumps. Endah has also proven that the use of an aquaponic system can increase crop production. In the experiments conducted, it was proven that the reuse of catfish manure as a nutrient for water spinach plants increased yields by 29.12%. [15].

The problem of developing a water quality monitoring system in aquaponics by measuring 5 parameters namely pH, dissolved oxygen levels, dissolved solids, temperature, and water turbidity is needed. This system uses five types of sensors to measure pH, dissolved oxygen levels, dissolved solids, temperature, and water turbidity. For processing, an Arduino Nano microcontroller equipped with an ESP8266-07 is used for communication devices. Arduino Nano here also acts as a node which will later send data to the base in the form of Raspberry Pi 3 Model B. In Raspberry Pi 3 Model B, sensor data will be uploaded to the database then data from the database will be displayed via the website so that users can monitor water quality conditions. aquaponics at any time remotely.

II. Desain dan Metode

2.1 Basic Design

The sensor node is placed on the side of the pool using Arduino as the main microcontroller. All sensors read by the Arduino are then sent to the raspberry pi3 via the ESP-07 module, which communicates via Wi-Fi to the access point. Raspberry pi3 will send processed data to the cloud server periodically every 1 minute. The box is made with industrial grade quality to make it stronger and waterproof so it can be installed outdoors for a long time. The basic design of this project is depicted in Figure 1 below.

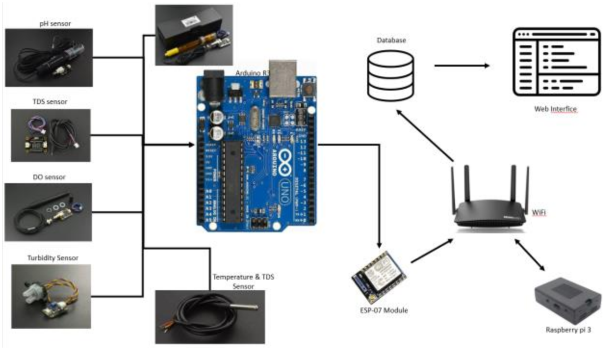


Fig 1 Basic design

2.2 System Design

The system gets input from five sensors to determine the value of water quality which includes temperature, turbidity, dissolved solids, pH, and dissolved oxygen. These five inputs will be processed in the Arduino and Raspberry Pi 3 Model B and then issued through the website. The processes that occur in the Arduino and Raspberry Pi 3 Model B are as follows:

1. Arduino will receive reading data from the five sensors connected to the I/O pin.
2. Arduino is connected to ESP8266-07 via UART serial communication so that Arduino Nano can access wifi to send data to Raspberry Pi 3 Model B.
3. ESP8266-07 will send sensor data from Arduino to Raspberry Pi 3 Model B using the MQTT protocol via LAN (Local Area Network).
4. On the Raspberry Pi 3 Model B, data will be filtered and then uploaded to a database using the internet.

From the database, the data will then be displayed on the website using PHP program code.

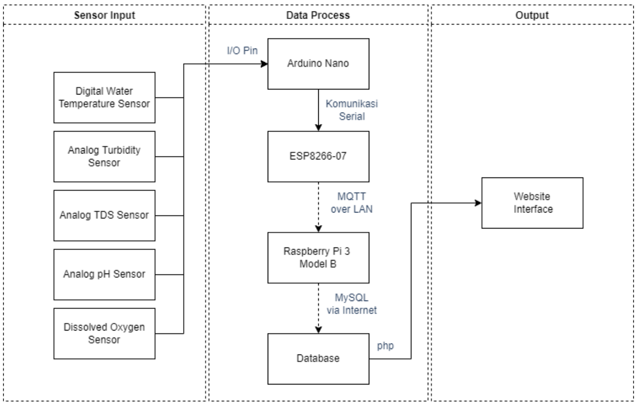


Fig 2 System design block diagram

2.3 Hardware Design

The numbering of equations uses Arabic numerals, written in brackets in the right alignment of the column. The equation is written inward by ± 10 mm. Equations that require space for more than one column, writing is possible across 2 columns.

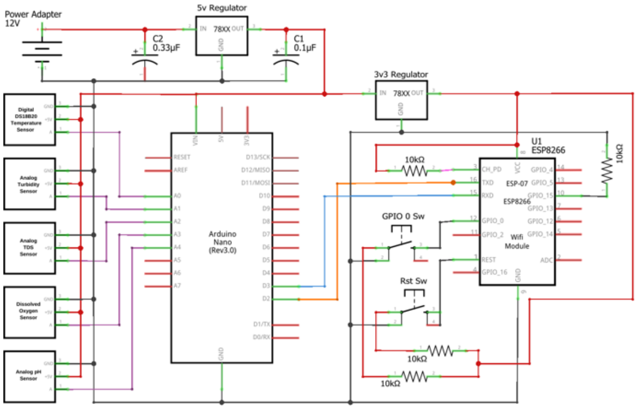


Fig 3 Hardware schematic diagram

From the schematic diagram above, Arduino Nano is used as a microcontroller. The sensors used are as follows:

a. DFROBOT DS18b20 which is connected to analog pin A0 which functions to measure water temperature.
b. The DFROBOT Analog Turbidity sensor is connected to the Arduino Nano analog pin A1 which functions to measure water clarity.
c. DFROBOT Analog TDS Sensor which is connected to the analog pin A2 Arduino Nano which functions to measure TDS (Total Dissolved Solid) or dissolved solids in water.
d. The DFROBOT Dissolved Oxygen Sensor is connected to the Arduino Nano A3 analog pin which functions to measure dissolved oxygen levels in water.
e. DFROBOT Analog pH Sensor which is connected to the analog pin A4 Arduino Nano which functions to measure the degree of acidity in water.

In addition, the LM7805 5VDC Voltage Regulator IC module is used as a power source for the use of Arduino Nano and all sensor modules. Then, the AMS1117 3.3VDC Voltage Regulator module is used as a power source for the use of ESP8266-07. Then the ESP8266-07 is given a push button on GPIO-0 and RESET which is used to upload the program code to the ESP8266-07. This ESP8266-07 functions as a wifi module and sends sensor data that has been collected on the Arduino Nano to the Raspberry pi via the MQTT protocol.

$$R(x,y) = \begin{cases} 1 & \text{if } f(x,y) >= T \\ 0 & \text{else} \end{cases} \qquad (1)$$

III. Result and Discussion

3.1 System planning

The design of a water quality monitoring system for aquaponics based on Arduino Nano and Raspberry Pi 3 Model B is depicted in the diagram in Figure 2. The system gets input from five sensors to determine water quality values which include temperature, turbidity, dissolved solids, pH, and dissolved oxygen. These five inputs will be processed in the Arduino Nano and Raspberry Pi 3 Model B and then issued through the website. The processes that occur in the Arduino Nano and Raspberry Pi 3 Model B are as follows:

1. Arduino Nano will receive reading data from the five sensors connected to the I/O pin.
2. Arduino Nano is connected to ESP8266-07 via UART serial communication so that Arduino Nano can access wifi to send data to Raspberry Pi 3 Model B.
3. ESP8266-07 will send sensor data from Arduino Nano to Raspberry Pi 3 Model B using the MQTT protocol via LAN (Local Area Network).
4. On the Raspberry Pi 3 Model B, data will be filtered and then uploaded to a database using the internet.
5. From the database, the data will then be displayed on the website using PHP program code.

3.2 Hardware Design

At this stage, a schematic is made for connecting the components used, namely in the form of connecting sensors and ESP8266-07 to the Arduino Nano. The schematic and pin configuration can be seen in Figure 3. From the schematic diagram, the Arduino Nano is used as the microcontroller. The sensors used are as follows:

a. DFROBOT DS18b20 which is connected to analog pin A0 which functions to measure water temperature.
b. The DFROBOT Analog Turbidity sensor is connected to the Arduino Nano analog pin A1 which functions to measure water clarity.
c. DFROBOT Analog TDS Sensor which is connected to the analog pin A2 Arduino Nano which functions to measure TDS (Total Dissolved Solid) or dissolved solids in water.
d. The DFROBOT Dissolved Oxygen Sensor is connected to the Arduino Nano A3 analog pin which functions to measure dissolved oxygen levels in water.
e. DFROBOT Analog pH Sensor which is connected to the analog pin A4 Arduino Nano which functions to measure the degree of acidity in water.

In addition, the LM7805 5VDC Voltage Regulator IC module is used as a power source for the use of Arduino Nano and all sensor modules. Then, the AMS1117 3.3VDC Voltage Regulator module is used as a power source for the use of ESP8266-07. Then the ESP8266-07 is given a push button on GPIO-0 and RESET which is used to upload the program code to the ESP8266-07. This ESP8266-07 functions as a wifi module and sends sensor data that has been collected on the Arduino Nano to the Raspberry pi via the MQTT protocol. A more detailed explanation of the hardware used is in Table 1 below.

Table 1 Hardware Module Pinout Table

Module	Module Pin	Power Source	Arduino Nano Pin	Function / Description
DFROBOT DS18B20 Digital Waterproof Temperature Sensor	VCC	5V		Input power 5V LM7805
	GND	GND		Ground
	Data		A0	Sending water temperature sensor data from the module to Arduino Nano
DFROBOT Analog Turbidity Sensor	VCC	5v		Input power 5V LM7805
	GND	GND		Ground

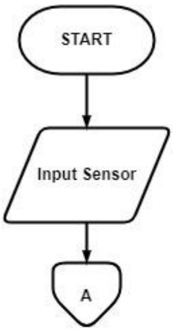
	Data		A1	Sending turbidity sensor data from the module to the Arduino Nano
DFROBOT Analog TDS Sensor	VCC	5v		Input power 5V LM7805
	GND	GND		Ground
	Data		A2	Sending TDS sensor data from the module to Arduino Nano
DFROBOT Dissolved Oxygen Sensor	VCC	5V		Input power 5V LM7805
	GND	GND		Ground
	Data		A3	Sending dissolved oxygen sensor data from the module to Arduino Nano
DFROBOT Analog pH Sensor	VCC	5V		Input power 5V LM7805
	GND	GND		Ground
	Data		A4	Sending pH sensor data from the module to Arduino Nano
ESP8266-07	VCC	3.3V		Input power 3.3V AMS1117
	GND	GND		Ground
	Rx		D2	Receive serial communication data with Arduino Nano
	Tx		D3	Sending serial communication data with Arduino Nano

The output of this system is data or values from each sensor displayed on the website. So that the sensor data that has been collected on the Arduino Nano will be sent to the Raspberry Pi 3B via the ESP8266 module and the MQTT protocol. After that the Raspberry Pi 3B will send the data to the database and display the data on the website.

3.3 Software Design

Before making software program code that will be used in the system, it is necessary to design the flow of the program first so that it facilitates the software implementation process later. The flow diagram of the system as a whole can be seen in Figure 3. The system begins by entering the sensors that have been connected to the Arduino Nano, then the Arduino Nano will read data from all sensors. To send data to ESP8266-07, Arduino Nano must wait up to 1 minute beforehand. If the data has been sent to the ESP8266-07, the ESP will send the data via the internet network and the MQTT protocol to the raspberry pi. Raspberry pi will later collect the data that has been received into an array and do filtering, if 5 minutes have been running then the last data will be sent by raspberry pi to the database. After the data is entered into the database, the data will be displayed on the website.

In the main system flow diagram in Figure 4 below, there is a subsystem for sending data from Arduino Nano to ESP8266-07. The flow chart for this subsystem can be seen in Figure 5. The system starts from the initialization of the serial library and software called espSerial via pins 2 and 3. After that it defines the pins used for sensor data input and also declares the required variables such as number node, location, ip mqtt server, wifi SSID, and wifi password of type string. Next, create a scheduler called waterMonitoring and after that create several methods. The first method called sendData functions to send Arduino data to the esp and creates a sendtoEsp task on the waterMonitoring scheduler to send water parameter data to the esp serial every 1 minute. The second method called sensorReadLoop functions to read from all sensors and creates a periodicSensorRead task on the scheduler to update sensor values every 22 seconds. The last method called checkCom functions to display incoming serial communication from the esp to arduino and creates the espComCheck task on the waterMonitoring scheduler to display if there is serial input from the ESP every 10ms. If so, the system will start serial communication and the ESP serial software and the system will wait for data requests from the esp. If the esp has requested data then the data will be sent to the ESP. If the esp has not requested the data, the system will wait until the esp requests the data.



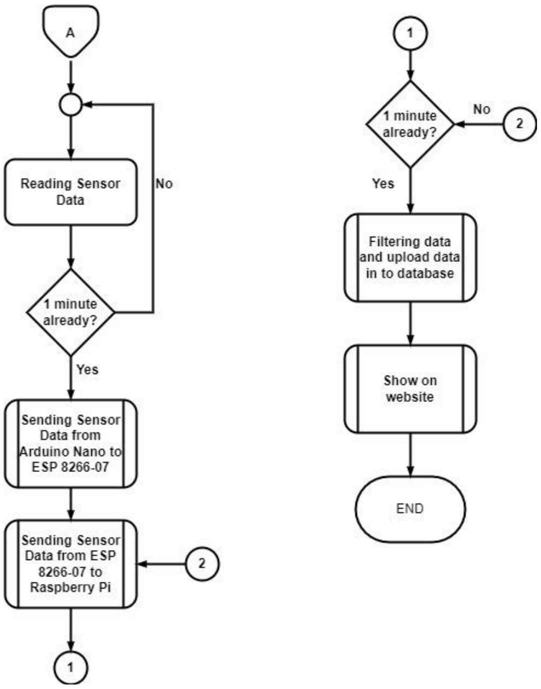

A
Reading Sensor Data
No
1 minute already?
Yes
Sending Sensor Data from Arduino Nano to ESP 8266-07
Sending Sensor Data from ESP 8266-07 to Raspberry Pi
2
1
1
1 minute already?
No
2
Yes
Filtering data and upload data in to database
Show on website
END

Fig 4 Main System flowchart

After the data is sent to the esp, the system will check if the serial input from the esp has the value "ESP Com Ready!" or not. If not, then esp will ask for data again, whereas if it is true then the system will activate the espComCheck task. The system will then initialize and activate all sensor modules and also the periodicSensorRead task. After that the system will execute the Water Monitoring Scheduler and check whether the Arduino Nano is off or not. If not, the system will re-execute the water Monitoring scheduler, whereas if the Arduino Nano dies, the system will be terminated.

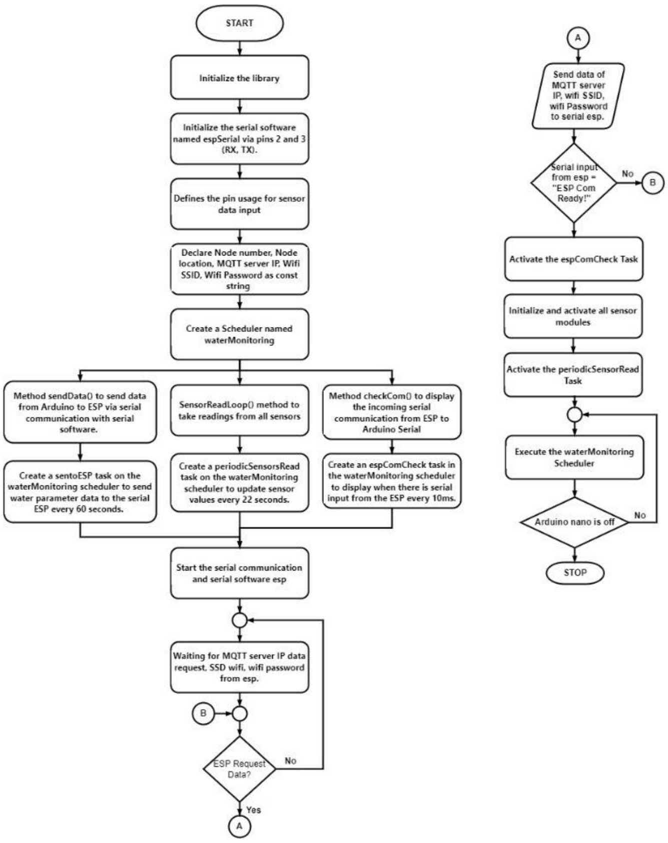

START
Initialize the library
Initialize the serial software named espSerial via pins 2 and 3 (RX, TX).
Defines the pin usage for sensor data input
Declare Node number, Node location, MQTT server IP, Wifi SSID, Wifi Password as const string
Create a Scheduler named waterMonitoring
Method sendData() to send data from Arduino to ESP via serial communication with serial software.
SensorReadLoop() method to take readings from all sensors
Method checkCom() to display the incoming serial communication from ESP to Arduino Serial
Create a sentoESP task on the waterMonitoring scheduler to send water parameter data to the serial ESP every 60 seconds.
Create a periodicSensorsRead task on the waterMonitoring scheduler to update sensor values every 22 seconds.
Create an espComCheck task in the waterMonitoring scheduler to display when there is serial input from the ESP every 10ms.
Start the serial communication and serial software esp
Waiting for MQTT server IP data request, SSD wifi, wifi password from esp.
B
ESP Request Data?
No
Yes
A
A
Send data of MQTT server IP, wifi SSID, wifi Password to serial esp.
Serial input from esp = "ESP Com Ready!"
No
B
Activate the espComCheck Task
Initialize and activate all sensor modules
Activate the periodicSensorRead Task
Execute the waterMonitoring Scheduler
Arduino nano is off
No
STOP

Fig 5 Data Transfer Subsystem Flowchart from Arduino Nano to ESP8266-07

In the main system flow diagram in Figure 4, it can be seen that there is a subsystem for sending sensor data from ESP8266-07 to Raspberry Pi. The flow chart for this subsystem can be seen in Figure 6. The system starts by declaring the use of the libraries needed for using wifi and also the MQTT protocol. After that, prepare a string variable to store the wifi ssid, wifi password, MQTT server ip, node number, and topic number. After that create a class object on the wifi and MQTT module and ESP will start serial communication then ask for the MQTT server ip, wifi SSID, and wifi password then stored in each variable. Then the ESP will run the function to start a connection with the wifi network and set the MQTT server on the specified IP with port 1883. If the ESP is not connected to a wifi connection then the ESP will try to connect it to the wifi network, if the ESP is already connected, then the ESP will check whether he is already connected with MQTT broker or not. If the ESP is not connected, it will try to connect to the MQTT broker. If it is already connected, the ESP will check the command line for incoming data to serial communication. If the command line contains ardData, ESP will send topics and messages in the form of data received from arduino nano to the MQTT broker. If not the ESP will be delayed for 500 ms and if the ESP is still on, then the system will restart starting from checking the wifi connection. Meanwhile, if the ESP does not turn on the system will be terminated.

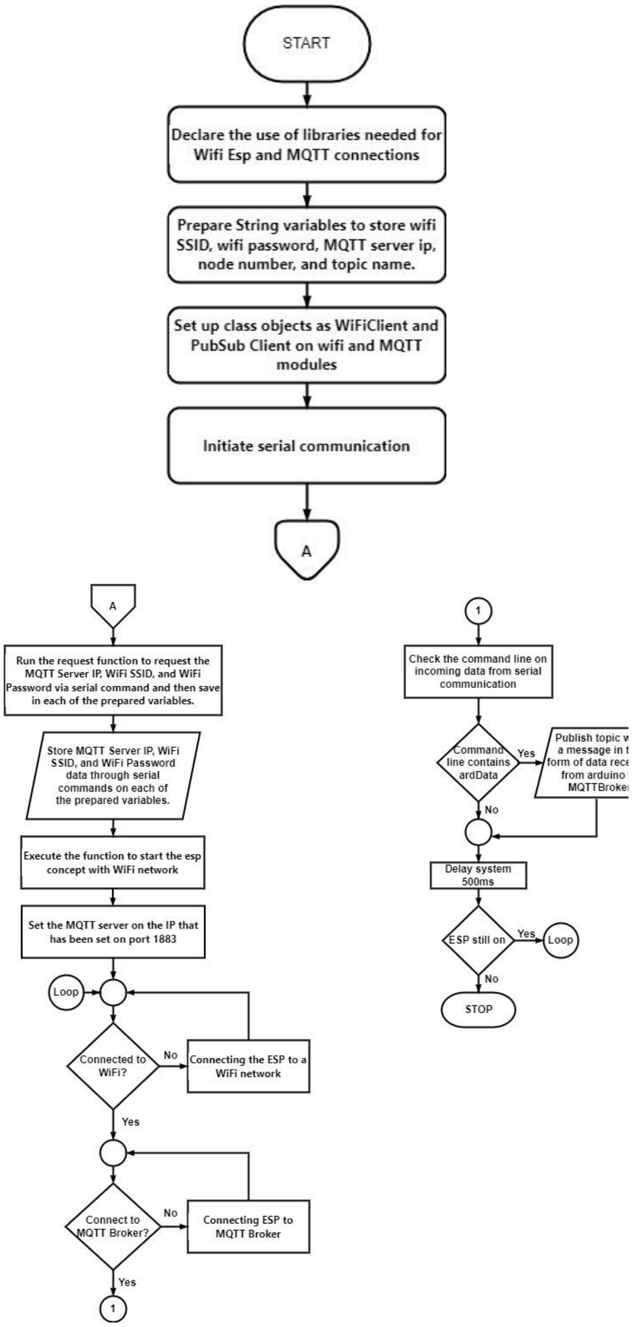

START
Declare the use of libraries needed for Wifi Esp and MQTT connections
Prepare String variables to store wifi SSID, wifi password, MQTT server ip, node number, and topic name.
Set up class objects as WiFiClient and PubSub Client on wifi and MQTT modules
Initiate serial communication
A
A
Run the request function to request the MQTT Server IP, WiFi SSID, and WiFi Password via serial command and then save in each of the prepared variables.
Store MQTT Server IP, WiFi SSID, and WiFi Password data through serial commands on each of the prepared variables.
Execute the function to start the esp concept with WiFi network
Set the MQTT server on the IP that has been set on port 1883
Loop
Connected to WiFi?
No
Connecting the ESP to a WiFi network
Yes
Connect to MQTT Broker?
No
Connecting ESP to MQTT Broker
Yes
1
1
Check the command line on incoming data from serial communication
Command line contains ardData
Yes
Publish topic w a message in t form of data rece from arduino MQTTBroker
No
Delay system 500ms
ESP still on
Yes
Loop
No
STOP

Fig 6 Data Transfer Subsystem Flowchart from ESP8266-07 to Raspberry Pi

In the main system flow diagram in Figure 4, it can be seen that there is a subsystem for filtering data and sending data to the database via the Raspberry Pi. The flow chart for this subsystem can be seen in Figure 7. The system starts from initializing the libraries needed in the system and creates an array for each sensor data. After that, the sensor data that has been sent from the ESP8266-07 to the raspberry pi, the raspberry pi will separate the data in 1 string into several arrays and include them in each sensor array.

The system will check if the length of the array is more than 1 or not, if not, the system will add it if there is data entered. If so, the system will compare and reduce the 2 values in the array from the 0th index to the last index and the results of the reduction will be entered into a temporary variable. The results of this reduction will later be compared with the maximum offset value that has been given to each sensor. If it exceeds the offset value, the data will be replaced with the data in the previous index array. If the length of the array is five, then the last array index will be sent to the database. After being sent to the database, the array will be emptied again and if the Raspberry Pi dies, the system will repeat the program from sensor readings that enter the Raspberry Pi, whereas if the Raspberry Pi does not turn on then the system will be terminated.

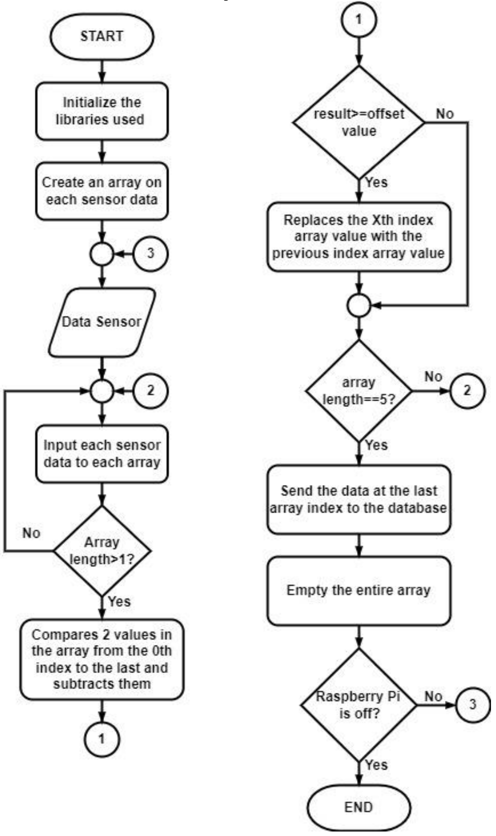

START
Initialize the libraries used
Create an array on each sensor data
3
Data Sensor
2
Input each sensor data to each array
No
Array length>1?
Yes
Compares 2 values in the array from the 0th index to the last and subtracts them
1
1
result>=offset value
No
Yes
Replaces the Xth index array value with the previous index array value
array length==5?
No
2
Yes
Send the data at the last array index to the database
Empty the entire array
Raspberry Pi is off?
No
3
Yes
END

Fig 7 Flowchart of Data Filtering Subsystem and Sending Data to Database

In the main system flow diagram in Figure 4, it can be seen that there are subsystems for displaying data on the website. The flow chart for this subsystem can be seen in Figure 8. The system starts by making a connection to the database and making queries to display all data from the database. Next, write a link that points to the bootstrap, ajax, and jquery libraries so that these libraries can be used on the website that is created. After that, write the page title and the title on the page. Then an input form is created so that users can enter a start date and end date to search for certain data. Then, there is a selection of conditions if the "Filter" button is pressed and not. If the button is not pressed, then all data in the database will be displayed in graphical form and tables will match the initial query made. But if the button is pressed, a new query will be created where the data displayed is only data that is between the start date and end date according to user input. To display data in graphical form, it is necessary to define the name of the graph and also determine the data to be used as the x-axis and y-axis. After that, all the required values are fetched from the database. Data retrieval uses a while loop so that as long as there is data in the database, the data will continue to be issued. Then to display data in tabular form, it is necessary to make the headings of each column first. After that, data is retrieved from the database and entered the appropriate column. Just like graphic data, the loop here also uses while to make sure all data has been fetched. The system will stop after all data is retrieved from the database.

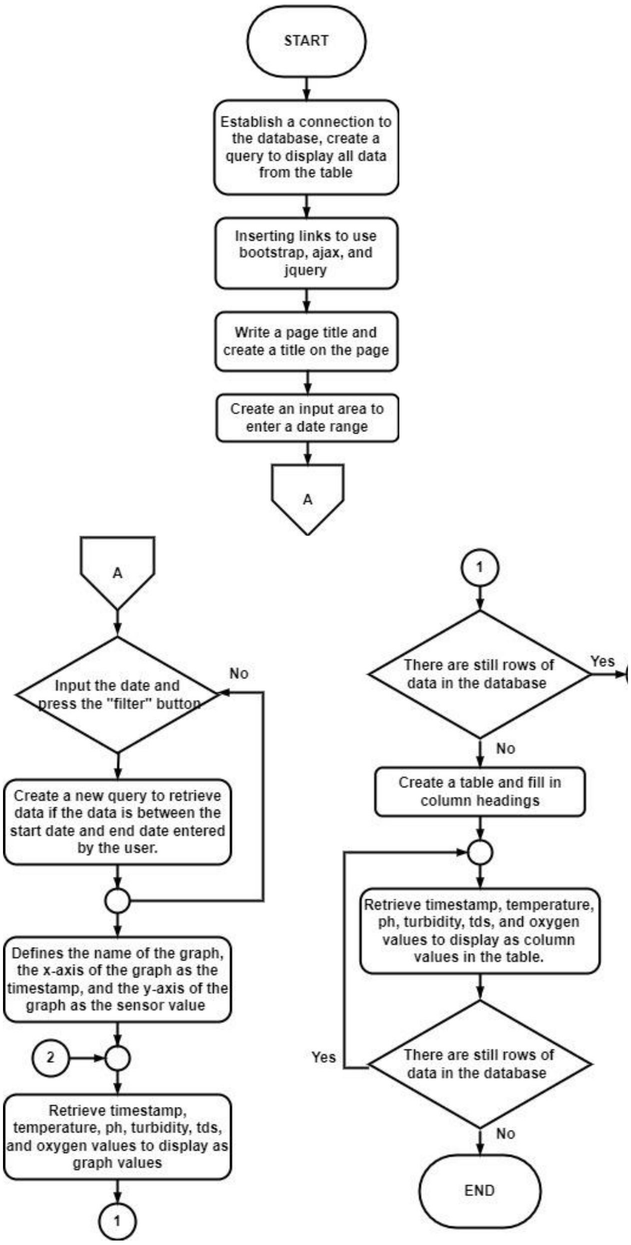

START
Establish a connection to the database, create a query to display all data from the table
Inserting links to use bootstrap, ajax, and jquery
Write a page title and create a title on the page
Create an input area to enter a date range
A
A
Input the date and press the "filter" button
No
Create a new query to retrieve data if the data is between the start date and end date entered by the user.
Defines the name of the graph, the x-axis of the graph as the timestamp, and the y-axis of the graph as the sensor value
2
Retrieve timestamp, temperature, ph, turbidity, tds, and oxygen values to display as graph values
1
1
There are still rows of data in the database
Yes
No
Create a table and fill in column headings
Retrieve timestamp, temperature, ph, turbidity, tds, and oxygen values to display as column values in the table.
Yes
There are still rows of data in the database
No
END

Fig 8 Subsystem Flowchart Show on Website

3.4 Hardware Implementation

In this section it is shown and demonstrated hardware implementation in accordance with the hardware design that has been prepared previously. The following is a picture of the device implementation that has been made :

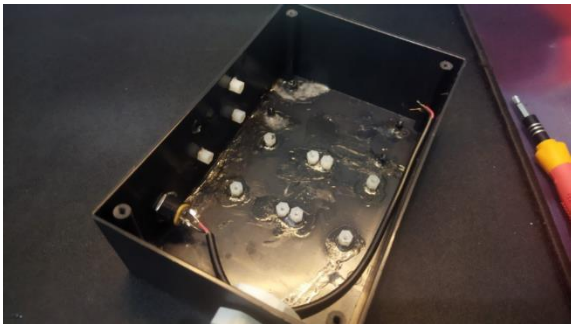


Fig 9 Project Box with the PCB stand that has been installed and the power cable from the DC jack

In Figure 9, the project box will be mounted on a nylon PCB stand using glue which is useful for placing sensor modules.

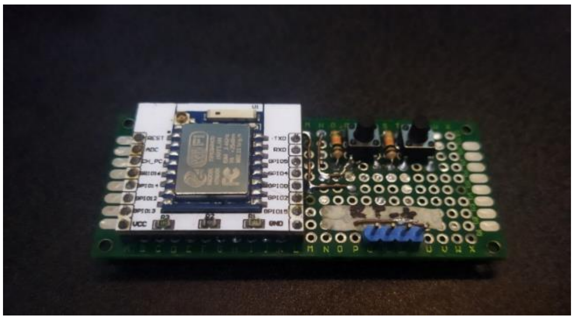


Fig 10 Implementation of the circuit in the ESP8266-07 section

In figure 10, ESP8266-07 is placed on the PCB matrix and given a push button on GPIO-0 and RESET to upload the program to ESP8266-07.

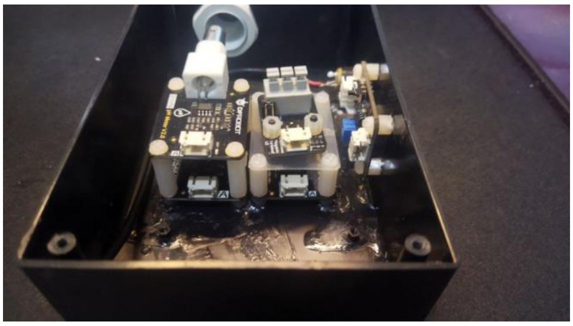


Fig 11 Implementation of sensor module installation in the Project Box

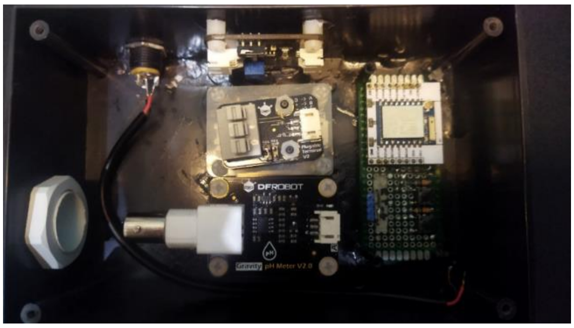


Fig 12 Placement of ESP8266-07 and sensor module in Project Box

All sensor modules will be placed in the project box as shown in Figure 10 and Figure 11 using nylon bolts attached to the nylon stand PCB.

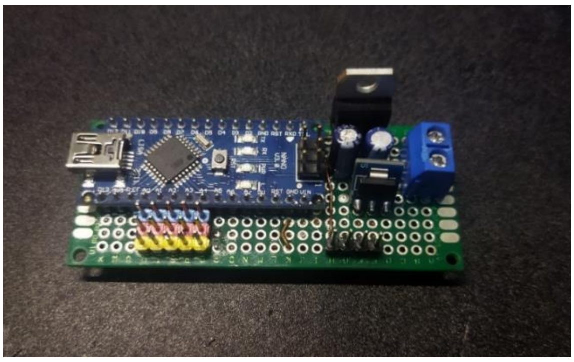


Fig 13 Implementation of the Arduino Nano circuit along with the voltage regulator

In Figure 13, the Arduino Nano is placed on the PCB matrix with the LM7805 5Vdc IC and also the AMS1117 3.3Vdc module.

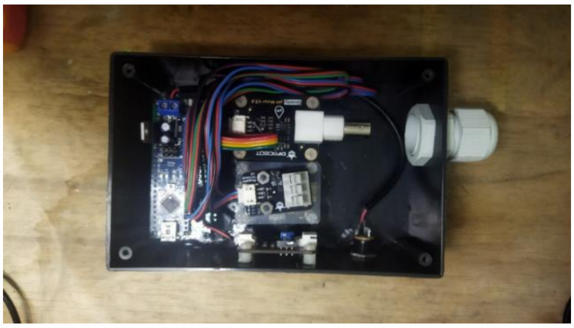


Fig 14 Placement of the entire series of modules and cabling on the project box

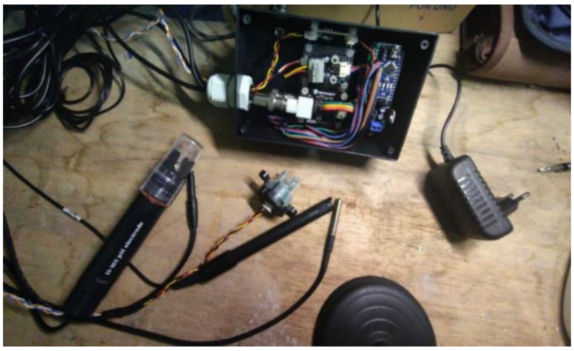


Fig 15 Installation of all sensor cables to each sensor module

In Figure 14, the Arduino Nano is placed on top of the ESP8266-07, then the cables for the sensors will be removed via Cable (Figure 15).

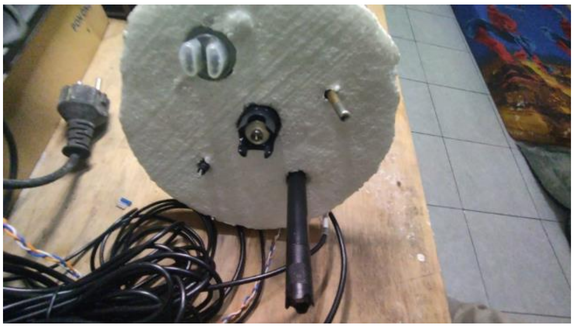


Fig 16. Placement of sensors on styrofoam buoys

So that the five sensors can float on the surface of the water, a cork float is provided at the top of the sensor as shown in Figure 16.

3.4 Software Implementation

This section will explain the implementation of the software in accordance with the design in the point about software design. In this implementation there is some program code needed. The first is the program code used on the Arduino Nano to read sensor values and send them to ESP8266-07 via UART. Second, there is the program code used on the ESP8266-07 to send data to the Raspberry Pi using the MQTT protocol. Third, there is program code that is implemented on the Raspberry Pi 3 Model B to filter data and upload it to the database. Finally, there is program code for building websites that can display data from a database and has a data search feature in a certain date range. Each program code will be accompanied by an explanation of the code so that the program code can be more easily understood.

3.5 System Testing

To ensure the system can run according to design, it is necessary to test each process in the system. Therefore, it is necessary to have a testing process for the aquaponic system that has been made. The goal of this process is to ensure that the system is running well starting from taking parameters using sensors to displaying the data on the website. The flow of the system testing are :

1. Test planning

At this stage, a system test design is made. Seven tests are designed here, namely testing sensor data reading, testing sensor error values, testing sending data from Arduino Nano to ESP8266-07, testing sending data from ESP8266-07 to Raspberry Pi 3 Model B, testing uploading data to the database, testing data display on the website, and testing data search features on the website.

2. Test setup

From the test plan that has been made, preparations begin to be carried out. All the equipment or devices needed will be prepared. Hardware, software, and various other configurations will be prepared. So that the next stage of testing can be carried out properly. Properties like buckets, clean water, turbid water, and others are also prepared.

3. Test execution

In this stage, the actual testing is executed. The seven tests are executed. A more detailed explanation for each test is further explained in the points below.

3.5.1 Sensor reading testing

In the first test, a test was carried out to find out whether the sensor managed to read the data or not. The test was carried out by trying sensor measurements in two types of water, namely cloudy water and clear water. From these two values, it can be seen whether the sensor displays two different values of the two types of water or not. The process of testing sensor data readings can be seen in Figures 17 and 18 below. Figure 17 shows two buckets of water, each containing turbid water (black bucket) and clear water (gray bucket). Figure 18 shows the Arduino Nano serial monitor where the sensor reading data is marked with a blue box. The test results can be seen more clearly in Table 2 and Table 3.

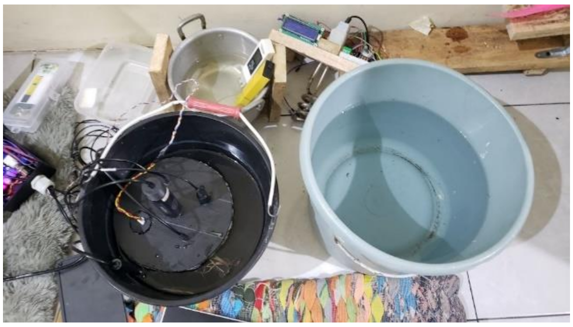
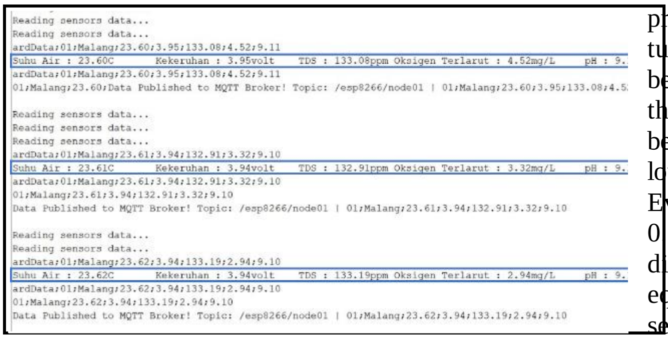


Fig 17 Placement of sensors in turbid water



```
Reading sensors data...
Reading sensors data...
ardData;01;Malang;23.60;3.95;133.08;4.52;9.11
Suhu Air : 23.60C       Kekeruhan : 3.95volt    TDS : 133.08ppm Oksigen Terlarut : 4.52mg/L     pH : 9.
ardData;01;Malang;23.60;3.95;133.08;4.52;9.11
01;Malang;23.60;Data Published to MQTT Broker! Topic: /esp8266/node01 | 01;Malang;23.60;3.95;133.08;4.5

Reading sensors data...
Reading sensors data...
Reading sensors data...
ardData;01;Malang;23.61;3.94;132.91;3.32;9.10
Suhu Air : 23.61C       Kekeruhan : 3.94volt    TDS : 132.91ppm Oksigen Terlarut : 3.32mg/L     pH : 9.
ardData;01;Malang;23.61;3.94;132.91;3.32;9.10
01;Malang;23.61;3.94;132.91;3.32;9.10
Data Published to MQTT Broker! Topic: /esp8266/node01 | 01;Malang;23.61;3.94;132.91;3.32;9.10

Reading sensors data...
Reading sensors data...
ardData;01;Malang;23.62;3.94;133.19;2.94;9.10
Suhu Air : 23.62C       Kekeruhan : 3.94volt    TDS : 133.19ppm Oksigen Terlarut : 2.94mg/L     pH : 9.
ardData;01;Malang;23.62;3.94;133.19;2.94;9.10
01;Malang;23.62;3.94;133.19;2.94;9.10
Data Published to MQTT Broker! Topic: /esp8266/node01 | 01;Malang;23.62;3.94;133.19;2.94;9.10
```

Fig 18 Sensor reading results on the Arduino Nano Serial Monitor

Table 2 Test results on Turbid Water

No	Temperature	Clarity	TDS	Oxygen	pH
1	26.65	4.34	264.95	2.8	6.38
2	26.65	4.33	270.71	3.36	6.43
3	26.66	4.33	267.89	3.57	6.45
4	26.67	4.33	269.6	3.16	6.46
5	26.69	4.34	267.1	3.15	6.41
6	26.71	4.33	265.52	3.03	6.42
7	26.7	4.34	264.83	2.8	6.4
8	26.71	4.33	264.51	1.93	6.37
9	26.71	4.33	262.75	2.3	6.37
10	26.72	4.33	263.06	1.97	6.37

Table 3 Test results on clear water

No	Temperature	Clarity	TDS	Oxygen	pH
1	26.89	4.56	221.86	2.79	7.36
2	26.9	4.56	220.39	4.08	7.35
3	26.9	4.44	217.69	3.63	7.32
4	26.9	4.45	218.63	3.35	7.33
5	26.9	4.44	216.42	3.03	7.33
6	26.9	4.45	217.52	3.25	7.33
7	26.9	4.41	217.76	3.11	7.31
8	26.86	4.43	217.06	2.87	7.3
9	26.86	4.47	216.44	3.26	7.32
10	26.86	4.47	216.06	3.21	7.32

From the two tables it can be seen that the five sensors managed to read water conditions well and provide different reading values in clear water and turbid water. The two values that most differentiate between clear and turbid water are the clarity value and the TDS (dissolved solids) value. From the table it can be seen that the clarity value in table 2 (turbid water) is lower than the clarity value in table 3 (clear water). Even though the distance is only in the range of 0.1 to 0.2, if converted to ntu, the interval will be a large distance because the difference of 0.05 voltage is equivalent to 100 ntu. Then for the TDS value, it can be seen that the two waters have quite a large difference in value, namely in the range of 50 where the TDS value in table 2 (turbid water) is greater than the TDS value in table 3 (clear water). This is appropriate because the dissolved solids content in turbid water will certainly be greater than the dissolved solids value in clear water.

3.5.2 Sensor Error Value Testing

In this test, a test is carried out to determine the error value of each sensor. Testing is carried out by comparing the value of the sensor measurement results with the value obtained from the measurement results using a manual measuring tool. From these two values, the sensor error value will then be obtained using the formula in Figure 18 below.

$$Error = \frac{|Trial\ value - True\ value|}{True\ value} \times 100$$

Fig 19 Error Calculation Formula

Due to the limitations of manual measurement tools, this test was only carried out on temperature, oxygen and TDS sensors. For pH and turbidity sensors, no tests were carried out because there was no manual measurement tool that could be used as a comparison. An example of testing can be seen in Figure 20 below. In the figure it can be seen that the sensor is placed in a container of water along with a manual measuring device. The measurement results are recorded in Table 2.

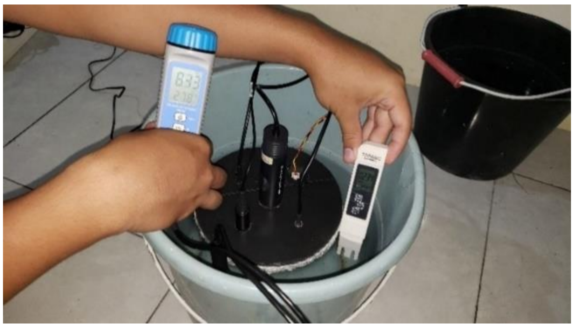


Fig 20 Test Data Retrieval Process

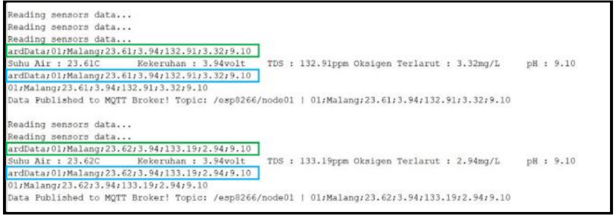
Table 4 Measurement Results of Sensor Values and Manual Measurement Values

No	Manual Measuring Tool			Sensor Reading Results			Error (%)		
	Temperature	TDS	Oxygen	Temperature	TDS	Oxygen	Temperature	TDS	Oxygen
1	23.5	233	5.05	23.36	228.76	4.87	0.60	1.82	3.56
2	23.5	231	4.97	23.38	225.9	4.18	0.51	2.21	15.90
3	23.5	235	5.01	23.37	229.12	4.37	0.55	2.50	12.77
4	23.5	234	4.76	23.35	229.18	4.03	0.64	2.06	15.34
5	23.5	234	5.18	23.38	229.83	5.51	0.51	1.78	6.37
6	23.5	234	5.3	23.36	230.29	4.86	0.60	1.59	8.30
7	23.5	232	5.4	23.38	229.83	5.36	0.51	0.94	0.74
8	23.5	222	5.38	23.38	229.17	4.52	0.51	3.23	15.99
9	23.5	222	5.63	23.4	229.11	4.25	0.43	3.20	24.51
10	23.5	225	5.61	23.36	229.17	4.89	0.60	1.85	12.83
AVERAGE							**0.54**	**2.12**	**11.63**

From table 4 it can be seen that the readings from the three sensors compared to the three manual measuring instruments produce an average error value of 0.54% for the temperature sensor, 2.12% for the TDS sensor, and 11.63% for the dissolved oxygen sensor. The three values are small enough so that the sensor is considered to be able to make the appropriate measurements.

3.5.3. Data Transfer Test from Arduino Nano to ESP8266-07

In this test, a test was carried out for sending data from Arduino Nano to ESP8266-07 using the UART communication protocol. Testing is carried out through a serial monitor by monitoring whether sensor reading data is successfully sent to ESP8266-07. An example of testing can be seen in Figure 21 below. In the figure, sensor reading data from Arduino is marked with a green box and data received by ESP8266-07 is marked with a blue box. The test results can be seen more clearly in Table 5.



```
Reading sensors data...
Reading sensors data...
Reading sensors data...
ardData;01;Malang;23.61;3.94;132.91;3.32;9.10
Suhu Air : 23.61C       Kekeruhan : 3.94volt    TDS : 132.91ppm Oksigen Terlarut : 3.32mg/L     pH : 9.10
ardData;01;Malang;23.61;3.94;132.91;3.32;9.10
01;Malang;23.61;3.94;132.91;3.32;9.10
Data Published to MQTT Broker! Topic: /esp8266/node01 | 01;Malang;23.61;3.94;132.91;3.32;9.10

Reading sensors data...
Reading sensors data...
ardData;01;Malang;23.62;3.94;133.19;2.94;9.10
Suhu Air : 23.62C       Kekeruhan : 3.94volt    TDS : 133.19ppm Oksigen Terlarut : 2.94mg/L     pH : 9.10
ardData;01;Malang;23.62;3.94;133.19;2.94;9.10
01;Malang;23.62;3.94;133.19;2.94;9.10
Data Published to MQTT Broker! Topic: /esp8266/node01 | 01;Malang;23.62;3.94;133.19;2.94;9.10
```

Fig 21 Serial communication data received by the ESP8266-07 is sent back to the Arduino Nano Serial Monitor

Table 5 Test Results for Sending Data to ESP8266-07

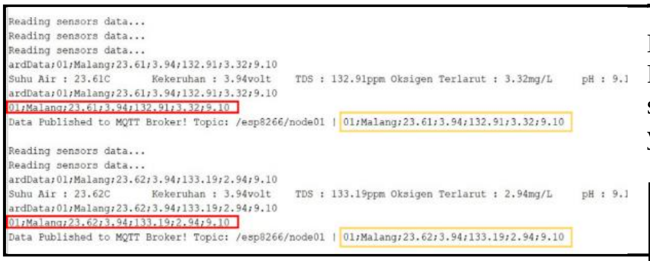
No	Data	Status
1	01;Malang;23.60;3.95;133.08;4.52;9.11	Succeed
2	01;Malang;23.61;3.94;132.91;3.32;9.10	Succeed
3	01;Malang;23.62;3.94;133.19;2.94;9.10	Succeed
4	01;Malang;23.63;3.94;133.57;4.38;9.10	Succeed
5	01;Malang;23.63;3.93;133.68;4.60;9.09	Succeed
6	01;Malang;23.60;3.93;133.74;3.35;9.07	Succeed
7	01;Malang;23.60;3.92;133.53;3.52;9.03	Succeed
8	01;Malang;23.60;3.89;133.02;4.53;8.99	Succeed
9	01;Malang;23.60;3.88;133.53;5.55;8.99	Succeed

10	01;Malang;23.62;3.88;133.68;4.93;8.99	Succeed

From Table 5 it can be seen that out of 10 tests, all sensor data from Arduino Nano was successfully sent to ESP8266-07. This proves that UART communication between Arduino Nano and ESP8266-07 can run smoothly. Because this transmission is done via the Rx Tx pin, data will always be sent as long as the cable connections on the two pins are properly connected.

3.5.4 Data Transfer Test from ESP8266-07 to Raspberry Pi 3 Model B

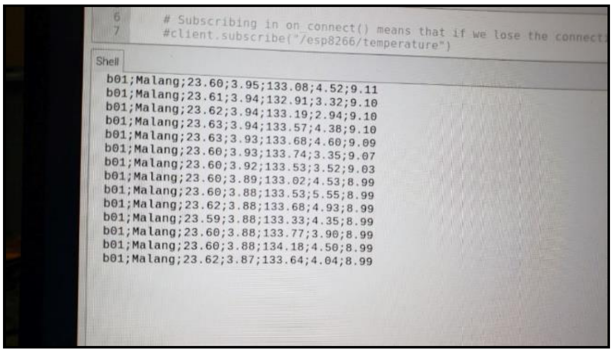
In this test, a test was carried out for sending data from the ESP8266-07 to the Raspberry Pi 3 Model B using the MQTT protocol. Testing is carried out via a serial monitor by monitoring whether data from the ESP8266-07 is successfully sent to the Raspberry Pi or not. In addition, monitoring was also carried out through the Thonny application terminal which was used to run the Python program code on the Raspberry Pi. The screenshots of the Thonny serial monitor and terminal can be seen in Figure 22 and Figure 23 below. In figure 22 data sent from ESP8266-07 is marked with a red box and data sent to the Raspberry Pi is marked with a yellow box. Figure 23 shows the Thonny application terminal displaying data received by the Raspberry Pi. The test results can be seen more clearly in Table 6.



```
Reading sensors data...
Reading sensors data...
Reading sensors data...
ardData;01;Malang;23.61;3.94;132.91;3.32;9.10
Suhu Air : 23.61C       Kekeruhan : 3.94volt    TDS : 132.91ppm Oksigen Terlarut : 3.32mg/L     pH : 9.1
ardData;01;Malang;23.61;3.94;132.91;3.32;9.10
01;Malang;23.61;3.94;132.91;3.32;9.10
Data Published to MQTT Broker! Topic: /esp8266/node01 | 01;Malang;23.61;3.94;132.91;3.32;9.10

Reading sensors data...
Reading sensors data...
ardData;01;Malang;23.62;3.94;133.19;2.94;9.10
Suhu Air : 23.62C       Kekeruhan : 3.94volt    TDS : 133.19ppm Oksigen Terlarut : 2.94mg/L     pH : 9.1
ardData;01;Malang;23.62;3.94;133.19;2.94;9.10
01;Malang;23.62;3.94;133.19;2.94;9.10
Data Published to MQTT Broker! Topic: /esp8266/node01 | 01;Malang;23.62;3.94;133.19;2.94;9.10
```

Fig 22 Data Sending Test to Raspberry Pi



```
        # Subscribing in on connect() means that if we lose the connect
        #client.subscribe("/esp8266/temperature")
```

```
b01;Malang;23.60;3.95;133.08;4.52;9.11
b01;Malang;23.61;3.94;132.91;3.32;9.10
b01;Malang;23.62;3.94;133.19;2.94;9.10
b01;Malang;23.63;3.94;133.57;4.38;9.10
b01;Malang;23.63;3.93;133.68;4.60;9.09
b01;Malang;23.60;3.93;133.74;3.35;9.07
b01;Malang;23.60;3.92;133.53;3.52;9.03
b01;Malang;23.60;3.89;133.02;4.53;8.99
b01;Malang;23.60;3.88;133.53;5.55;8.99
b01;Malang;23.62;3.88;133.68;4.93;8.99
b01;Malang;23.59;3.88;133.33;4.35;8.99
b01;Malang;23.60;3.88;133.77;3.90;8.99
b01;Malang;23.60;3.88;134.18;4.50;8.99
b01;Malang;23.62;3.87;133.64;4.04;8.99
```

Fig 23 Raspberry Pi Delivery Thonny Terminal

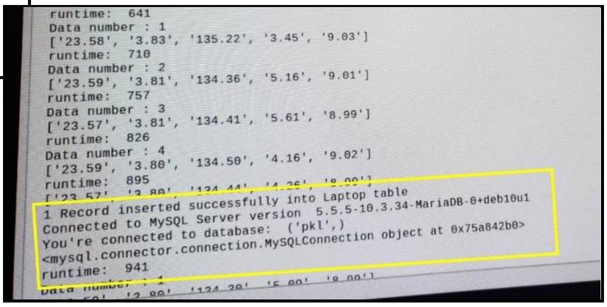
Table 6 Results of Data Sending Test to Raspberry Pi

No	Data	Status
1	01;Malang;23.60;3.95;133.08;4.52;9.11	Succeed
2	01;Malang;23.61;3.94;132.91;3.32;9.10	Succeed
3	01;Malang;23.62;3.94;133.19;2.94;9.10	Succeed
4	01;Malang;23.63;3.94;133.57;4.38;9.10	Succeed
5	01;Malang;23.63;3.93;133.68;4.60;9.09	Succeed
6	01;Malang;23.60;3.93;133.74;3.35;9.07	Succeed
7	01;Malang;23.60;3.92;133.53;3.52;9.03	Succeed
8	01;Malang;23.60;3.89;133.02;4.53;8.99	Succeed
9	01;Malang;23.60;3.88;133.53;5.55;8.99	Succeed
10	01;Malang;23.62;3.88;133.68;4.93;8.99	Succeed

From Table 6 it can be seen that out of 10 tests, all data from the ESP8266-07 was successfully sent to the Raspberry Pi. This proves that MQTT communication between ESP8266-07 and Raspberry Pi can run smoothly. Because sending via the MQTT protocol is done via wifi, data will always be sent as long as the sender and receiver are on the same network.

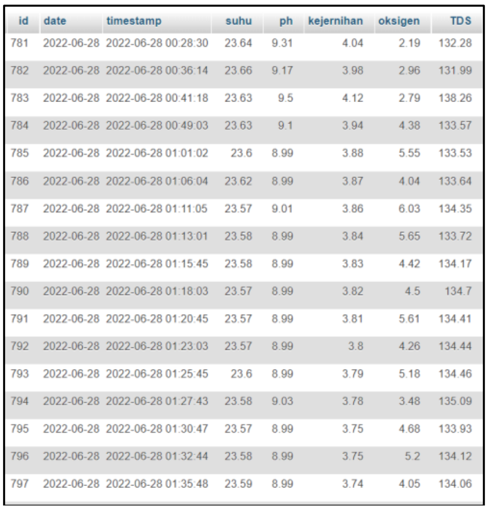
3.5.5 Data Upload Test to Database

In this test, a test was carried out to find out whether the data on the Raspberry Pi was successfully uploaded to the database or not. Testing is carried out by monitoring data transmission via the Thonny application terminal and also phpmyadmin. Test images on the Thonny application terminal and screenshots of the MySQL database on phpMyAdmin can be seen in Figure 24 and Figure 25. In Figure 24 data that was successfully uploaded to the database is marked with a yellow box.



```
runtime:  641
Data number : 1
['23.58', '3.83', '135.22', '3.45', '9.03']
runtime:  710
Data number : 2
['23.59', '3.81', '134.36', '5.16', '9.01']
runtime:  757
Data number : 3
['23.57', '3.81', '134.41', '5.61', '8.99']
runtime:  826
Data number : 4
['23.59', '3.80', '134.50', '4.16', '9.02']
runtime:  895
['23.57', '3.80', '134.44', '4.26', '8.99']
1 Record inserted successfully into Laptop table
Connected to MySQL Server version  5.5.5-10.3.34-MariaDB-0+deb10u1
You're connected to database:  ('pkl',)
<mysql.connector.connection.MySQLConnection object at 0x75a842b0>
runtime:  941
Data number : 1
```

Fig 24 Terminal Thonny Upload Database



id	date	timestamp	suhu	ph	kejernihan	oksigen	TDS
781	2022-06-28	2022-06-28 00:28:30	23.64	9.31	4.04	2.19	132.28
782	2022-06-28	2022-06-28 00:36:14	23.66	9.17	3.98	2.96	131.99
783	2022-06-28	2022-06-28 00:41:18	23.63	9.5	4.12	2.79	138.26
784	2022-06-28	2022-06-28 00:49:03	23.63	9.1	3.94	4.38	133.57
785	2022-06-28	2022-06-28 01:01:02	23.6	8.99	3.88	5.55	133.53
786	2022-06-28	2022-06-28 01:06:04	23.62	8.99	3.87	4.04	133.64
787	2022-06-28	2022-06-28 01:11:05	23.57	9.01	3.86	6.03	134.35
788	2022-06-28	2022-06-28 01:13:01	23.58	8.99	3.84	5.65	133.72
789	2022-06-28	2022-06-28 01:15:45	23.58	8.99	3.83	4.42	134.17
790	2022-06-28	2022-06-28 01:18:03	23.57	8.99	3.82	4.5	134.7
791	2022-06-28	2022-06-28 01:20:45	23.57	8.99	3.81	5.61	134.41
792	2022-06-28	2022-06-28 01:23:03	23.57	8.99	3.8	4.26	134.44
793	2022-06-28	2022-06-28 01:25:45	23.6	8.99	3.79	5.18	134.46
794	2022-06-28	2022-06-28 01:27:43	23.58	9.03	3.78	3.48	135.09
795	2022-06-28	2022-06-28 01:30:47	23.57	8.99	3.75	4.68	133.93
796	2022-06-28	2022-06-28 01:32:44	23.58	8.99	3.75	5.2	134.12
797	2022-06-28	2022-06-28 01:35:48	23.59	8.99	3.74	4.05	134.06

Fig 25 MySQL database on phpMyAdmin

From the test results above, it can be seen that the sensor data on the Raspberry Pi was successfully uploaded to the database. In Figure 24 it can be seen that there are data numbers 1 to 4 that were not uploaded to the database. This is because uploading to the database is only done every 5 minutes while the Raspberry Pi receives data from the ESP8266-07 every 1 minute so there are 4 data that are not uploaded to the database.

3.5.6 Data Output Testing on the Website

In this test, a test is carried out to find out whether the sensor data in the database is successfully displayed on the website or not. Testing is done by looking at the data in the database and the data displayed on the website. If all the data in the database is successfully displayed on the website, then we can be sure that the website can display data properly. A screenshot of the data display in graphical form can be seen in Figure 26, while a screenshot of the data display in tabular form can be seen in Figure 27. The data displayed on this website refers to the database in Figure 25 previously.

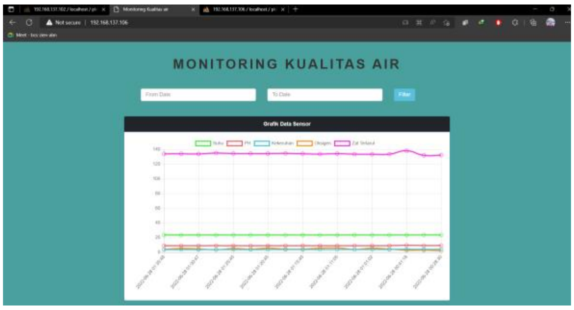
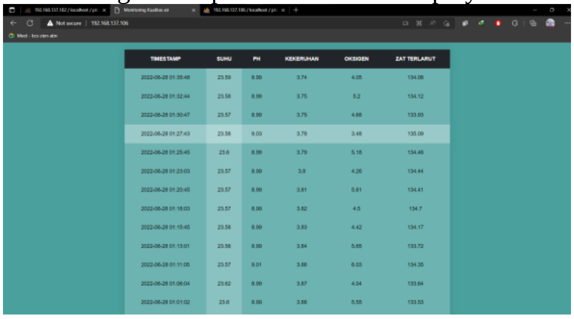


Fig 26 Graph Data Website Display



TIMESTAMP	SUHU	PH	KEKERUHAN	OKSIGEN	ZAT TERLARUT
2022-06-28 01:35:48	23.59	8.99	3.74	4.05	134.06
2022-06-28 01:32:44	23.58	8.99	3.75	5.2	134.12
2022-06-28 01:30:47	23.57	8.99	3.75	4.68	133.93
2022-06-28 01:27:43	23.58	9.03	3.78	3.48	135.09
2022-06-28 01:25:45	23.6	8.99	3.79	5.18	134.46
2022-06-28 01:23:03	23.57	8.99	3.8	4.26	134.44
2022-06-28 01:20:45	23.57	8.99	3.81	5.61	134.41
2022-06-28 01:18:03	23.57	8.99	3.82	4.5	134.7
2022-06-28 01:15:45	23.58	8.99	3.83	4.42	134.17
2022-06-28 01:13:01	23.58	8.99	3.84	5.65	133.72
2022-06-28 01:11:05	23.57	9.01	3.86	6.03	134.35
2022-06-28 01:06:04	23.62	8.99	3.87	4.04	133.64
2022-06-28 01:01:02	23.6	8.99	3.88	5.55	133.53

Fig 27 Table Data Website Display

From the test above, the website has succeeded in displaying data from the database both in graphic and table form. In Figure 24, the last data entered into the database was data at 01.35, this corresponds to the leftmost data in the graphical view and the topmost data in the table view. The appearance of data on the website is done in descending order based on the id in the database. Therefore, the data display on the website can be exactly the same as the data display on phpMyAdmin.

3.5.7 Data Search Testing on the Website

In this test, testing is carried out to find out whether the data search feature can run well on the website that has been made. In addition to displaying data, the created website also has a feature to search data on a certain date. Users can enter a start date and end date in the input locations provided. The data search results for graphical data can be seen in Figure 28 and the data search results for table data can be seen in Figure 29.

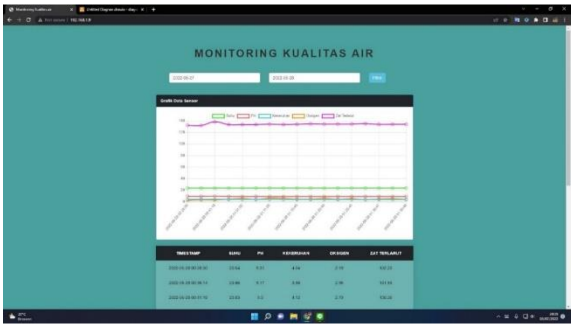
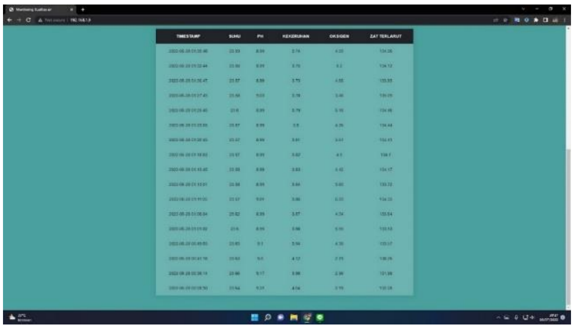


Fig 28 Graph Data Search Display Results



TIMESTAMP	SUHU	PH	KEKERUHAN	OKSIGEN	ZAT TERLARUT

Fig 29 Table Data Search Display Results

From the test above, the data search feature on the website can run well. The data displayed corresponds to the date entered by the user in the input field. For the results of this search data display, data from the database is displayed in ascending order so that the leftmost data on the graph and the topmost on the table is the sensor data at the earliest time of the day.

IV. Conclusion & Suggestions

4.1 Conclusion

Water Monitoring System Design in Arduino Nano and Raspberry Pi 3 Model B Based Aquaponics, there are several conclusions that can be drawn, namely:

1. The system has successfully programmed the Arduino Nano microcontroller to be able to

read sensors by utilizing the sensor library and Arduino IDE software. The sensor needs to be connected to the input output pin on the Arduino Nano so that sensor data can be read.

2. The system has succeeded in sending data from Arduino Nano to Raspberry Pi 3 Model B with the data flow sent to ESP8266-07 first using UART. Then data from the ESP8266-07 is sent to the Raspberry Pi 3 Model B using the MQTT protocol via the wifi network.
3. The system has succeeded in displaying sensor data to the website by uploading sensor data from the Raspberry Pi to the database and then displaying it on the website. In making a website, a connection is made to the database and executing queries so that the sensor data in the database can be displayed on the website.
4. The system has been tested in every stage and gives the appropriate results. Sensor data readings are in accordance with the type of water as evidenced by the difference in sensor values when tested in clear water and turbid water. The sensor has also been proven to be able to make appropriate readings because it has a fairly small error value of 0.54% for the temperature sensor, 2.12% for the TDS sensor, and 11.63% for the dissolved oxygen sensor. Sending data from Arduino Nano to ESP8266-07 via UART and sending data from ESP8266-07 to Raspberry Pi using MQTT also went smoothly because out of 10 data, all data has been successfully sent. Uploading data to the database was also successful and the data in the database was successfully displayed on the website. The data search feature on a certain date on the website can also provide appropriate results. From all these tests, it can be said that the system has succeeded in monitoring water quality and displaying the data on the website.

4.2 Suggestions

Some suggestions that can be applied for further development are:

1. The process of sending data is done offline so that the system can still be used when the internet connection is unstable and can be used in remote areas with a slow internet connection.
2. The system is also implemented to monitor water quality in hydroponic plants which are included in aquaponics.
3. Make website hosting to an external server so that the monitoring process can be done anywhere.
4. Adding a battery or other power source to the system so that the system is not constrained when a power outage occurs.
5. Added power efficiency considerations if the system is supplied with limited resources.

REFERENCES

[1] O. Nindita Priambodo, D. Rachmat, and A. Novrizal, “SISTEM AKUAPONIK CERDAS UNTUK KETAHANAN PANGANMASYARAKAT MANDIRI DI DESA KARANGASIH,” *Prosiding Konferensi Nasional Pengabdian Masyarakat*, vol. 3, pp. 415–424, 2022.

[2] Helin G Yudawisastra *et al.*, “Budikdamber akuaponik sebagai strategi ketahanan pangan dan stimulus kewirausahaan saat pandemi covid-19,” *BEMAS: Jurnal Bermasyarakat*, vol. 3, no. 2, pp. 162–170, Mar. 2023, doi: 10.37373/bemas.v3i2.258.

[3] Y. Liu, X. Ma, L. Shu, G. P. Hancke, and A. M. Abu-Mahfouz, “From Industry 4.0 to Agriculture 4.0: Current Status, Enabling Technologies, and Research Challenges,” *IEEE Trans Industr Inform*, vol. 17, no. 6, pp. 4322–4334, Jun. 2021, doi: 10.1109/TII.2020.3003910.

[4] D. Megawati, K. Masykuroh, D. Kurnianto, K. Kunci, and : Akuaponik, “Rancang Bangun Sistem Monitoring PH dan Suhu Air pada Akuaponik Berbasis Internet of Thing (IoT) Design of Monitoring System for PH and Water Temperature in Aquaponic Base on Internet of Thing (IoT),” *TELKA*, vol. 6, no. 2, pp. 124–137, 2020.

[5] Nawawi, Sriwahidah, and A. Asdar Jaya, “IbKIK BUDIDAYA IKAN NILA SISTEM AKUAPONIK,” *Jurnal Dedikasi Masyarakat*, vol. 2, no. 1, 2018.

[6] M. Zarantoniello *et al.*, “Growth and Welfare Status of Giant Freshwater Prawn (Macrobrachium rosenbergii) Post-Larvae Reared in Aquaponic Systems and Fed Diets including Enriched Black Soldier Fly (Hermetia illucens) Prepupae Meal,” *Animals*, vol. 13, no. 4, Feb. 2023, doi: 10.3390/ani13040715.

[7] E. D. Sulichantini, “AQUAPONIK,” Samarinda, 2021. Accessed: Jul. 26, 2023. [Online]. Available: https://repository.unmul.ac.id/bitstream/handle/123456789/26766/AQUAPONIK.pdf?sequence=1

[8] H. Hasibur Rohman, E. Samuel Pasaribu, P. Firdaus Nuryananda, R. Firdaus Setiawan, and J. Timur, “Menggagas Pertanian Berkelanjutan Melalui Akuaponik: Penerapan dan Dampaknya dalam Pengabdian kepada Masyarakat Kalirungkut Promoting Sustainable Agriculture through Aquaponics: Implementation and its Impact in Community Engagement in Kalirungkut,” *Jurnal Hasil Pengabdian Masyarakat Indonesia*, vol. 2, no. 3, pp. 114–123, 2023, doi: 10.58192/karunia.v2i2.1142.

[9] A. Bhawiyuga and W. Yahya, “SISTEM MONITORING KUALITAS AIR KOLAM BUDIDAYA MENGGUNAKAN JARINGAN SENSOR NIRKABEL BERBASIS PROTOKOL LORA,” *Jurnal Teknologi Informasi dan Ilmu Komputer*, vol. 6, no. 1, pp. 99–106, 2019, doi: 10.25126/jtiik.201961292.

[10] D. Azhari and A. M. Tomasoa, “KAJIAN KUALITAS AIR DAN PERTUMBUHAN IKAN NILA (Oreochromis niloticus) YANG DIBUDIDAYAKAN DENGAN SISTEM AKUAPONIK,” *Jurnal Akuatika Indonesia*, vol. 3, 2018.

[11] J. Mailoa, E. P. Wibowo, and R. Iskandar, “Sistem Kontrol dan Monitoring Kadar PH Air pada Sistem Akuaponik Berbasis NodeMCU ESP8266 Menggunakan Telegram,” *Jurnal Ilmiah Komputasi*, vol. 19, no. 4, Dec. 2020, doi: 10.32409/jikstik.19.4.336.

[12] R. Alfia, A. Widodo, and N. Kholis, “Sistem Monitoring Kualitas Air pada Sistem Akuaponik Berbasis IoT 707 SISTEM MONITORING KUALITAS AIR PADA SISTEM AKUAPONIK BERBASIS IOT,” *Jurnal Teknik Elektro*, vol. 10, 2021.

[13] A. Burlian, Y. Rahmanto, S. Samsugi, and A. Sucipto, “SISTEM KENDALI OTOMATIS PADA AKUAPONIK BERBASIS MIKROKONTROLER ARDUINO UNO R3,” *JTST*, vol. 02, no. 1, pp. 1–6, 2021.

[14] M. Iskandar Nasution, L. Husna Lubis, and W. Utama Akbar Ritonga, “Implementasi Mikrokontroler Berbasis IoT Untuk Optimalisasi Kinerja Sistem Akuaponik,” *Jurnal Phi: Jurnal Pendidikan Fisika dan Fisika Terapan*, vol. 4, no. 1, p. 2022, 2023.

[15] L. Endah Rahmadhani, L. Ilman Widuri, and P. Dewanti, “KUALITAS MUTU SAYUR KASEPAK (KANGKUNG, SELADA, DAN PAKCOY) DENGAN SISTEM BUDIDAYA AKUAPONIK DAN HIDROPONIK,” *Jurnal Agroteknologi*, vol. 14, no. 01, 2020.