ELE3305

Design Report

Student Name: Kate Bowater

Student Number: U1019160

Due Date: 8 April 2024

# Executive Summary

Table of Contents

[Executive Summary 2](#_Toc163395366)

[Application Scenario 2](#_Toc163395367)

[Specifications 3](#_Toc163395368)

[Sensors 3](#_Toc163395369)

[Control 3](#_Toc163395370)

[Power Supply 4](#_Toc163395371)

[System Overview 4](#_Toc163395372)

[Network 4](#_Toc163395373)

[Node-Red 5](#_Toc163395374)

[OpenPLC 6](#_Toc163395375)

[HMI 6](#_Toc163395376)

[Data Decoding/Scaling 7](#_Toc163395377)

[Dissolved Oxygen 7](#_Toc163395378)

[pH 7](#_Toc163395379)

[Water Temperature 8](#_Toc163395380)

[ICT security risks 8](#_Toc163395381)

[Risk Mitigation **Error! Bookmark not defined.**](#_Toc163395382)

[Conceptualisation 11](#_Toc163395383)

[Coding 11](#_Toc163395384)

[pH 11](#_Toc163395385)

[Water Temperature 11](#_Toc163395386)

[Dissolved Oxygen 11](#_Toc163395387)

[References 12](#_Toc163395388)

# Application Scenario

Aquaponics is defined as a closed system where fish and plants survive and grow together by providing each other with nutrients. In this system, rather than filtering out waste from the tanks, the waste from fish is consumed by bacteria and converted into nitrates which plants use for nourishment [1] [2]. The thriving plants then release oxygen into the water which circulates back to the fish. In this system, the delicate balance of water pH, temperature and oxygen levels are vital for a happy ecosystem [3], and so requires constant monitoring to ensure fish and plants are obtaining adequate nutrients and not stressed out.

Implementing automated monitoring and maintenance to the aquaponics system using Internet of Things (IoT) will take the strain off the owner of the aquaponics system to ensure all these elements are measured and addressed in a timely manner, reducing the risk of plant and fish illness or death caused by lack of supervision [1] [4].

# Specifications

## Sensors

The details of the sensors required to monitor the aquaponics system include the following:

Table 1: Sensor information

|  |  |  |
| --- | --- | --- |
| **What is Measured** | **Ideal Range** | **Units** |
| Dissolved Oxygen (DO) | 5-8  80-100 | mg/L  % |
| pH | 6.5-8 | - |
| Water Temperature | 18-30 | °C |

It should be noted that the ideal range for DO in water is 5-8 mg/L [1] [2], but this can change depending on water temperature and salinity. Hence an alternative measurement is through percentage with an ideal range of 80-100%. In this case, less than 80% will risk the health of the fish and plants, while greater than 100% is likely to create algae blooms [12].

## Control

To control the aforementioned factors of the tanks, a PLC program can provide automation of required equipment. For dissolved oxygen, an air pump will add more O2 to the water and turn off when the required level is reached.

The water temperature is controlled using a heater to increase the temperature, and when the water temperature is too high then a water pump will activate to increase flow through the tank to cause temperature to decrease.

The pH level is a complex factor closely tied to oxygen and temperature levels. It is assumed in a well-balanced system that the pH level will be self-maintained and should not fall outside the ideal range, but if it does it requires direct intervention from the user due to complex associations to other variables, so only an alert will be sent to the user.

These actions are all outlined in Table 2 below. If any values outside of the ideal range mentioned in Table 1 are detected, the PLC will activate to maintain balance of the aquaponics system. It should be noted that the provided actions are not exclusive (as temperature, DO and pH can directly affect each other), but are the only ones provided to maintain simplicity for the conceptual design of the system.

Table 2: Control of sensor readings

|  |  |  |
| --- | --- | --- |
| **Sensor** | **Problem** | **Required Action** |
| DO | < 80% | Turn on air pump |
| > 100% | Turn off air pump |
| pH | < 6.5 | Notify user |
| > 8 | Notify user |
| Water Temp | < 18 °C | Turn on heating system |
| > 30 °C | Turn on water pump |

## Power Supply

For the system to run over Internet of Things (IoT) the software can be implemented on a small device such as Raspberry Pi. This can be connected to a small power supply either via mains or utilise solar. It could also incorporate battery back up to ensure upkeep of the tanks in the event of power loss. The simplicity of this aquaponics system can then be scaled up to meet larger demands including aquariums and farming [2].

# System Overview

## Network

The information obtained from the sensors in Table 1 are sent via LoRaWAN (long range wide area network) to The Things Network (TTN). As LoRaWAN networks use ALOHA based protocols to avoid collision, the sensors don’t need specific gateways and messages are sent through all gateways within range. In this way, the sensor information can be sent to a message broker via the network server [13].

The message broker, in this case Mosquitto, uses MQTT protocol to take these messages as subscriptions. MQTT is appropriate for this situation because of the simple, small amounts of data being sent from the tank’s sensors which do not require large headers or bandwidth. HTTP is another protocol that could have been used for this scenario, but MQTT is low cost, reliable in unstable connections, and provides scalability unlike HTTP which requires more resources to run [6]. The request-response model of HTTP is also a disadvantage compared to MQTT which can send messages without direct authentication to the nodes.

Using TTN, the message broker then sends the message through a local network to the subscribed nodes: Node-Red, OpenPLC, and a web-based human machine interface (HMI). Figure 1 shows an overview of the network and connected devices.

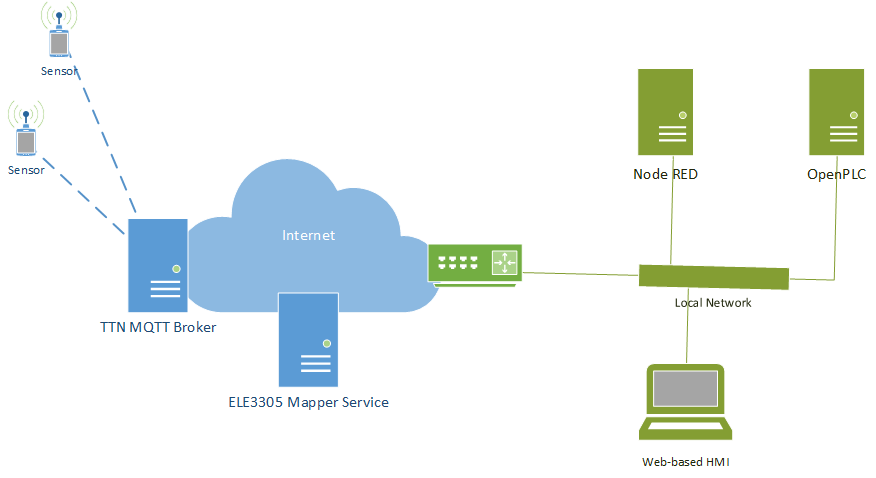


Figure 1: Network layout for aquaponics system

The protocol to connect the sensors and devices to OpenPLC is Modbus TCP/IP. Using TCP/IP allows the data to be carried over ethernet as it is already a widely used protocol and provides compatibility with preexisting ethernet infrastructure [10]. As the data being transferred is only small, Modbus is also suitable for this application due to its limited packet transfer size of 260 bytes [11].

For the design of this system, the nodes need to have IP addresses to communicate to others on the local network. For simulation of the system using a virtual operating system, the loopback address of 127.0.0.1 can be used with different ports to access the different nodes.

In setting up the physical system for the aquaponics setting, the local area network (LAN) requires dedicated IP addresses and subnet. A subnet of 255.255.255.0 will allow up to 254 usable hosts which is sufficient for the needs of the aquaponics setup at present. This means that the available IP addresses are from 192.0.0.0 to 223.255.255.255 [14] [15]. The nodes could then have any IP address in this range, and the table below provides the designations for the LAN discussed:

Table 3: Designated node IP addresses

|  |  |
| --- | --- |
| **Node** | **IP address** |
| OpenPLC | 192.168.0.2 |
| Node-Red | 192.168.0.3 |
| HMI | 192.168.0.4 |

## Node-Red

Node-Red is used to implement the IoT network. This creates flows for the decoding and scaling of the received information before forwarding to OpenPLC and HMI. This also provides the interface for the user to interact with the system where they can view alerts and controls, as mentioned below under HMI.

Use of Node-Red is appropriate as it is easy to set up with visual nodes for flow creation. This means other technicians can easily access and interpret the layout to make changes for the user if needed. Connection to other nodes on the IoT network is also easy to manage using the flows, and the built-in UI dashboard is a necessary feature for the user and is conveniently provided in Node-Red’s features.

## OpenPLC

OpenPLC is used to implement the logic that will provide automated control to the aquaponics system, as laid out in Table 2 under specifications. The circuit is created and uploaded to the OpenPLC software which is connected to Node-Red to be implemented in the flows.

The actions in Table 2 can be described using pseudocode below. For DO and water temperature, the pumps will activate automatically but the design will also allow the user to turn on or off the air pump and water pump as needed. The water heating does not have the same functionality because if the heater is left on and reaches the high temperature threshold then it is simply wasting power and risking the health of the fish and plants.

Pseudocode for aquaponics system:

|  |
| --- |
| If dissolved\_oxygen is low OR airpump\_button\_on:   * Turn on air pump   Else If dissolved\_oxygen is high OR airpump\_button\_off:   * Turn off air pump   If pH\_level is high OR low:   * Turn on alert   If water\_temp is low:   * Turn on heater   Else turn off heater  If water\_temp is high OR waterpump\_button\_on:   * Turn on water pump   Else if water\_temp is not high OR water\_temp is low OR waterpump\_button\_off:   * Turn off water pump |

## HMI

The human-machine interface (HMI) is the final node which the user can access using a browser in kiosk mode. This will only allow access to Node-Red’s user interface (UI) and will display information on the pH, DO and water temperature of the tanks. This can be implemented on any tablet device for convenience so the user can easily transport it while checking the tanks of the aquaponics system.

# Data Decoding/Scaling

The information sent by the sensors must be decoded and scaled before being used in the system. Each data packet contains 14 bytes. The raw information contained in each packet is encoded in base 64 and can be decoded in Node-Red using a code snippet written in Javascript as follows:

// Separating data in base64 to individual bytes  (hexadecimal)

var hexbytes = Buffer.from(msg.payload.uplink\_message.frm\_payload,'base64');

msg.payload = hexbytes; //overwriting the payload with the hex values

return msg;

This takes all the information from the data packet and converts it to hexadecimal values that correspond to the bytes. After this, the values can be isolated and scaled for the requirements of the system.

## Dissolved Oxygen

The value for DO is contained in bytes 0 and 1. It is understood that the range in these bytes is 0-5000. While the ideal range for DO is from 80-100%, a maximum range to120% will be sufficient for determining excessively high values. It is then a simple matter of scaling through the following equation:

The following code obtains the transmitted value and scales it to the required 0-120% which can then be used in the system.

const bytes = msg.payload;  // holding all the bytes of the payload

let DO\_data = ((bytes[0]<<8)+bytes[1]);

// the byte needs to be shifted because low byte of transmission is the highest byte of value

// the range is 0-5000

let DO\_value = DO\_data / (5000/120); //scaling to the DO range 0-120%

msg.payload = DO\_value;

return msg;

## pH

The pH value is obtained from bytes 6 and 7. It is understood that this value has a range of 0-100 and was multiplied by 10 for transmission, so first it is divided by 10 before scaling into the pH range 0-14 using the following equation:

The following code determines the transmitted value and scales it to the pH level using the equation above.

const bytes = msg.payload;   // holding the bytes of the payload

const ph\_data = ((bytes[6]<<8)+bytes[7]);  //bytes 6 and 7 hold the pH value

let ph\_value = ph\_data /10; // the transmission was multiplied by 10, so need to divide by 10 for true value

let ph\_info = ph\_value / (100/14); // scaling the data to the range 0-14 for pH level

msg.payload = ph\_info;

return msg;

## Water Temperature

Water temperature is contained in bytes 8 and 9. It is noted that the value was multiplied by 100 for transmission. After dividing by 100, the temperature value is in Kelvin. This is then converted to Celsius by the following equation:

The following code was used to decode and evaluate the temperature from the bytes:

const bytes = msg.payload;  // holding all bytes from the payload

const temperature = ((bytes[8]<<8)+bytes[9]);  //bytes 8 and 9 hold the temperature value

let temp\_kelvin = temperature/100;  // temperature is transmitted in kelvin\*100. So needs to be divided by 100

let temp\_C = temp\_kelvin - 273.15; // converting from kelvin to celcius

msg.payload = temp\_C; // overwrite the msg to display temp in celcius

return msg;

# ICT Security Risks

With regards to MQTT and the IoT network that the aquaponics system utilizes, security threats typically derive from the following three aspects: authentication, authorization, and package encryption [7]. Attackers can take advantage of weaknesses in these aspects to cause problems in the system, steal data, or perform DDOS attacks [6] [7] [8]. Modbus TCP is also vulnerable to DDoS attacks as it was originally designed for serial connections which had no security features [9]. While the sensors of the aquaponics system do not transmit sensitive data (i.e. there is no information that can be used for identity theft or fraud), interference with the data packets can be used to obtain login information which would then allow an unauthorized user to tamper with the information from the nodes. Damage to the tanks’ environment can occur as a result of changing inputs and outputs on the PLC, sending misleading data to the user, or blocking the system from performing its intended routines.

## Authentication

Without appropriate authentication, information from the nodes and sensors can be intercepted and stolen when being sent across the network and through the internet. Recommended mitigation strategies include using secure socket layer (SSL) or transport layer security (TLS) certificates for authentication [7]. Documentation for TTN states that it already supports authentication using API keys, OAuth access tokens and session cookies [16]. The Node-Red editor on the other hand is not secured by default and requires configuration to incorporate security settings. It uses HTTP but can be changed to use HTTPS certificates within the settings file. It also supports API authentication via username and password credentialing for admin access only, as well as setting user permissions to restrict access to the flows [17].

Regarding OpenPLC, the threat involves access to the program to upload unwanted programs or tampering with the state of the controls. As with Node-Red, OpenPLC uses only HTTP and it was found that the login credentials are sent in plaintext which exhibits an easy exploit [18]. The software does not provide any inherent security features so mitigation strategies could include setting up TLS certificates, changing the default username and password to something more secure, or isolation of the software to reduce the risk of attack via other nodes.

## Authorization

TTN also supports authorization using Rights which are assigned to the API keys used in authentication [16]. As mentioned above, OpenPLC requires log in credentials which must be changed from the default of ‘openplc’ for higher security.

Unauthorized access to physical devices, such as the devices containing the HMI or the nodes, can be mitigated by using strong passwords for logins, revoking privileges to certain parts of the device, or, in the case of the HMI for the user, setting up biometric authentication such as fingerprint or facial recognition, if the device supports it.

As the system is not anticipated to be used for web browsing at all, the risk of downloading malware is minimal, however the risk still exists should someone else gain access to the logins by packet sniffers or an injection attack to do so [7] [18]. An antivirus software is not recommended due to limitations in processing capabilities of the small devices used for the IoT network [6] [8]. In lieu of this, it is recommended to ensure that authorization for each node is sufficiently secure so that if a node is compromised then malicious operations cannot be easily transferred to other nodes [6].

## Encryption

Encryption will help prevent stolen packets from being used or tampered with. The data packets sent via LoRaWAN are typically encrypted using advanced encryption standard, AES-128 symmetric keys [13]. Hashing passwords further protects authentication methods and makes them more difficult to decipher [7], and this is already included in Node-Red software [17]. Securing messages through MQTT via Mosquitto can be achieved by configuring SSL/TLS end-to-end AES encryption [19].

## Patching and Other Recommendations

One final method of security is to ensure software is up to date and firmware patches are regularly applied. Out of date software can exhibit vulnerabilities that have been previously exploited, and applying updates can mitigate this risk as well as eliminating bugs that may cause issues in the system. For devices using IoT, connectivity may not always be assured so downloading the required updates can prove difficult and time consuming [8]. One suggestion is to provide an IT maintenance schedule for the user’s system to check for issues and apply updates as necessary. Another suggestion is to have an additional device part of the network that will download the software updates separately, then push them to the devices once completed. Educating the user on cyber security and signs of potential interference can also help address the consequences of attacks.

# Conceptualisation

Conceptualisation and documentation of the system and code to achieve the functionality.

This includes the local network configuration. This needs to include details such as the assumptions you have made, function definitions and shortcomings of the implementation and potential improvements. This section should be supported by block diagrams, flow charts etc as appropriate.

\*\*\*

## Coding

The raw information is sent in base 64 and is decoded using a code snippet written in Javascript as follows:

// Converting data in base64 to bytes

var bytes = Buffer.from(msg.payload.uplink\_message.frm\_payload,'base64');

msg.payload = bytes; //overwriting the payload with contained bytes

return msg;

After this, the values can be obtained from the bytes contained in the payload, then processed and scaled for the requirements of the system.

### pH

### Water Temperature

### Dissolved Oxygen

# References

[1] M.M.M. Mahmoud, R. Darwish and A.M. Bassiuny, “Development of an economic smart aquaponic system based on IoT,” *J. Eng. Res.*, Aug 2023, doi: https://doi.org/10.1016/j.jer.2023.08.024.

[2] M. F. Taha et al., “Recent Advances of Smart Systems and Internet of Things (IoT) for Aquaponics Automation: A Comprehensive Overview,” *Chemosensors*, vol. 10, no. 8, Aug 2022, doi: https://doi.org/10.3390/chemosensors10080303

[3] S. Fu, W. Xing, J. Wu, J. Chen and S. Liu, “Research and design of an intelligent fish tank system,” *PLoS ONE*, vol. 18, no. 5, May 2023, doi: <https://doi.org/10.1371/journal.pone.0285105>.

[4] F. A. Z. Shaikh and U. Bhaskarwar, “Smart Aquarium using IoT,” *Int. J. Res. Appl. Sci. Eng. Tech. (IJRASET),* vol. 10, no. 3, Mar 2022. [Online]. Available:<https://papers.ssrn.com/sol3/papers.cfm?abstract_id=4089695>

[5] C.-H. Chiung, Y.-C. W, J.-X. Zhang and Y.-H. Chen, “IoT-Based Fish Farm Water Quality Monitoring System,” *Sensors,* vol. 22, no. 17, Sep 2022, doi: https://doi.org/10.3390/s22176700.

[6] A. Zamifroiu et al. “IoT Communication Security Issues for Companies: Challenges, Protocols and The Web of Data,” *Proc. 14th Int. Conf. Bus. Exc. 2020*, vol. 14, no. 1, pp. 1109-1120, doi: 10.2478/picbe-2020-0104.

[7] B. Russell and D. Van Duren, “Practical Internet of Things Security,” Packt Publishing, 2016.

[8] M. Hunain et al. “Preventing MQTT Vulnerabilities Using IoT-Enabled Intrusion Detection System,” *Sensors*, vol. 22, no. 2, doi: 10.3390/s22020567.

[9] E. Gamess, B. Smith and G. Francia III, “Performance Evaluation Of Modbus TCP In Normal Operation And Under A Distributed Denial Of Service Attack,” *Int. J. Comp. Net. Comm. (IJCNC)*, vol. 12, no. 2, doi: 10.5121/ijcnc.2020.12201.

[10] *Introduction To Modbus TCP/IP*, Acromag, Wixom, MI, USA, 2005.

[11] *Modbus Application Protocol Specification,* v1.1b3, Modbus, Andover, MA, USA, 2012.

[12] Fundamentals of Environmental Measurements. “Dissolved Oxygen.” fondriest.com, https://www.fondriest.com/environmental-measurements/parameters/water-quality/dissolved-oxygen/ (accessed Apr. 2, 2024).

[13] The Things Industries. “LoRaWAN.” thethingsindustries.com, https://www.thethingsindustries.com/docs/getting-started/lorawan-basics/ (accessed Apr. 6, 2024).

[14] Cisco. “Understand Host and Subnet Quantities.” cisco.com, https://www.cisco.com/c/en/us/support/docs/ip/routing-information-protocol-rip/13790-8.html (accessed Apr. 8 2024).

[15] Cisco. “Configure IP Addresses and Unique Subnets for New Users”. cisco.com, https://www.cisco.com/c/en/us/support/docs/ip/routing-information-protocol-rip/13788-3.html (accessed Apr. 8, 2024).

[16] The Things Industries. “Authentication.” thethingsindustries.com, https://www.thethingsindustries.com/docs/api/concepts/auth/ (accessed Apr. 9, 2024).

[17] Node-Red. “Securing Node-RED.” nodered.org, https://nodered.org/docs/user-guide/runtime/securing-node-red (accessed Apr. 10, 2024).

[18] W. Alsabbagh, C. Kim and P. Langendörfer, “Good Night, and Good Luck: A Control Logic Injection Attack on OpenPLC,” doi: 10.13140/RG.2.2.32913.20321.

[19] mosquito. “mosquitto-tls man page.” mosquitto.org, https://mosquitto.org/man/mosquitto-tls-7.html (accessed Apr. 10, 2024).