Project Semester 4

Daniel Lawton - 20084608

Applied Computing - Internet Of Things(IOT)

TABLE OF CONTENTS

Introduction

For this module, the main aim spanning the semester was learning how to implement a sensor that could be used as one of the USV's (Unmanned Surface Vehicles) many features/functionalities.

The sensor must perform its own core function, and also have the ability to be used with a micro:bit. The micro:bit is a small compact, programmable microcontroller, known for its versatility and being user-friendly to introduce new people to the world of coding and electronics.

With this in mind, the goal was to develop a way that allows a sensor to perform, using the microbit as its controller. That is capable of publishing the data from the sensor from the microbit to a MQTT broker, from the broker to be easily accessible by a user by looking at their phone.

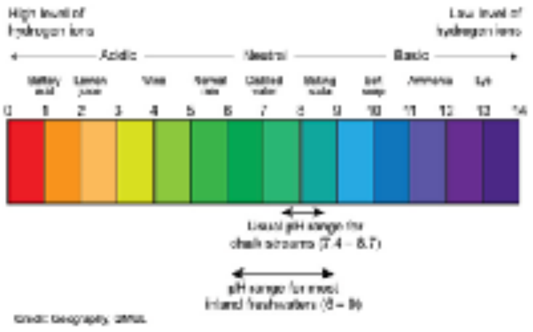
Project Overview

The main aims of my use case were as shown below:
- Have a sensor function using a micro:bit (exclusively).
- Cloud connectivity (ability to send data from sensor to microbit, then to a broker (IoT platform)
- Have the sensors readings be easily accessible for anyone with a phone.
- Brainstorm a way to secure the sensor to the USV, (Container).

Sensor

For my idea of a sensor that could be used as a part of a "network" of sensors, was a PH sensor. The idea stemmed from the overall theme of the USV, as an eco-friendly aqua vessel that can collect data from any type waters to study. I thought a ph sensor would be an appropriate inclusion as it fit the core tone of the USV, to be used for marine research by biologists from Catholic University of Valencia.

A ph sensor is a device that is used to measure the pH of a solution(liquid). pH is a measure of the acidity or alkalinity of a solution. This would be an excellent way, for example to see the effect Acid rain would have on a water source. Acid rains is a type of rain that is more acidic than normal rainwater. It can be caused by the emissions of certain pollutants, which are due to the burning of fossil fuels. These pollutants react with rainwater to form sulphuric acid and or nitric acid. Acid rain makes lakes and rivers more acidic as well as being harmful to aquatic life. I envisioned the sensor as a way to interpret or cement hypothesis on why some sources of water might be showing signs of degradation.



The sensor used was a SEN0161 ph sensor module, this type of ph sensor is designed to be used with microcontrollers(microbit). And consists of a pH electrode and a signal conditioning circuit board(analog-to-digital converter), that measures the output voltage of the module and converts it to a ph Value.

Cloud Connectivity

Cloud connectivity is where the IoT aspect of the use case is most apparent, using a microbit to retrieve data from the sensor send it by radio to another microbit attached to a wifi module, which contains code that publishes the data to an MQTT broker in the cloud.

Making Access to the data easy

Using any MQTT IoT platforms (apps), the topic in which the data has been sent to can be accessed to show the real-time readings of data. The key of this part of the project, though an objective overall, was to make the retrieval of the data easy and accessible to any users of any level of experience. The use case should be universal to anyone wanting to try and understand how it works and implement it themselves.

What was Achieved

Overall in the semester what was achieved were the following:

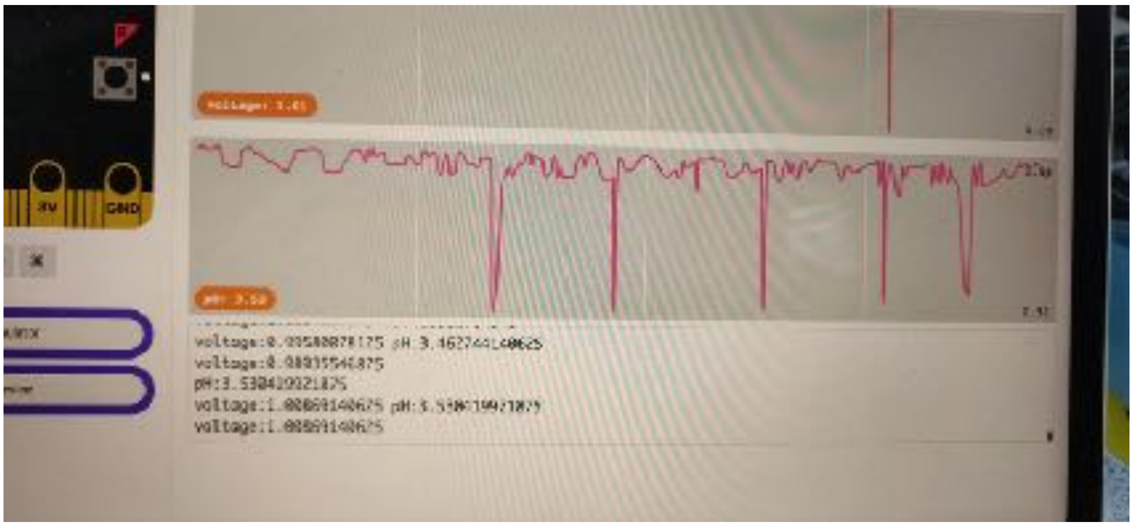
- Configure pH sensor to work with micro:bit.
- Implement MQTT Services with the micro:bit.
- Real-Time Data Recording

Configuring the pH sensor:

The specific ph sensor module used for this use case, the SEN0161, needed to be configured thoroughly to work at an acceptable level.
The first obstacle was supplying the sensor with 5 volts. Getting the sensor to work was simple using a laptop to power both the sensor and the microbit, however it became apparent that the sensor was showing clear inaccuracies in the voltages/Ph values being expressed. Using a multimeter and an oscilloscope the sensor was beeped out showing no signs of issues, besides the voltage being supplied not being enough. The problem was resolved by modding an Arduino to be the power supply for the sensor. A breakout board was also used to connect the sensor to the microbit.

The next task, after solving the power issue, was the micro:bit code, to calibrate the sensor to achieve accurate readings. This was done by having multiple samples of solutions that all had specific ph values pre determined. Along with the solutions another ph sensor was supplied to compare the microbit ph sensor values to the “true” values found by the already configured “accurate” ph sensor . When the real values were found, these values were then took into consideration when determining the offset for the values when creating the code for the micro:bit ph sensor. The goal was to get accurate readings that would be within 0.5 to 0.8 of the accurate sensor ph values. So for example if the Accurate sensor determined a solution to be ph 4, the microbit sensor would still be considered accurate if it was roughly 3.6 or even 3.2

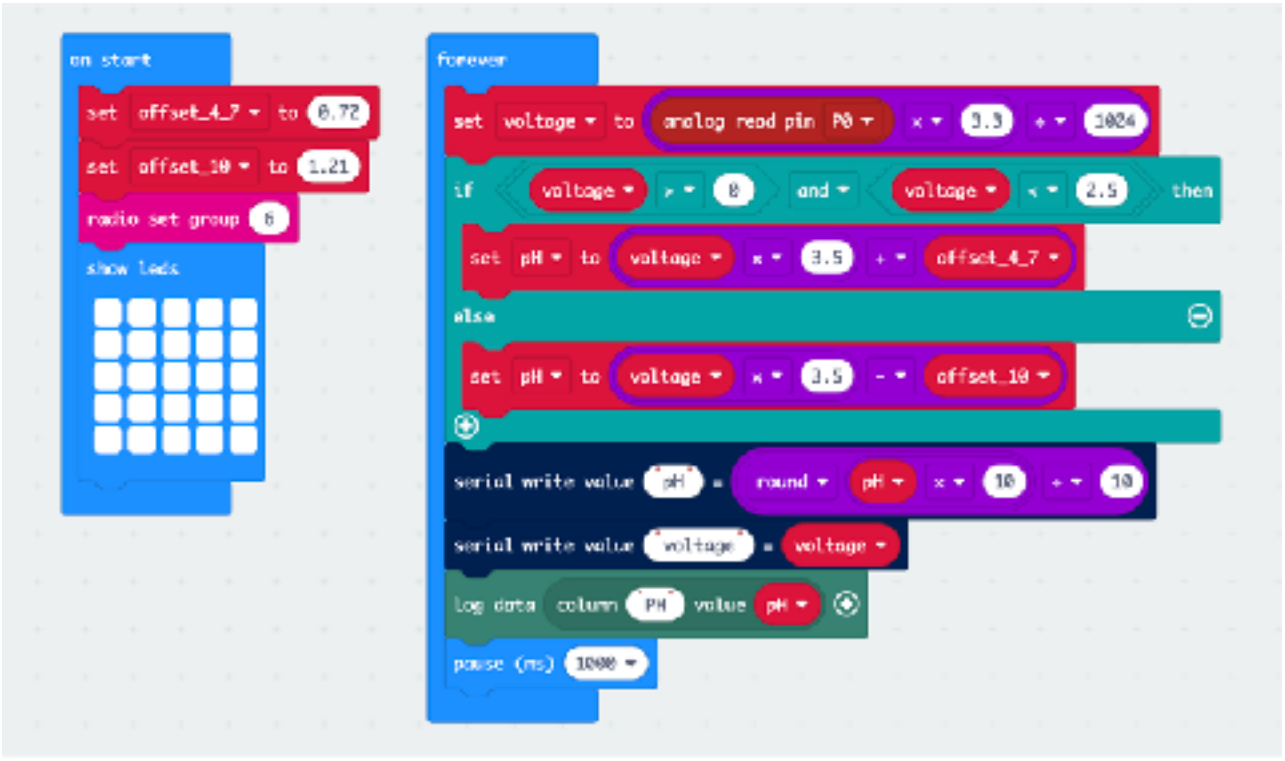
Microbit ph Sensor Values



```
voltage:0.99580078125 pH:3.462744140625
voltage:0.98935546875
pH:3.53042921875
voltage:1.00069140625 pH:3.530719921875
voltage:1.00069140625
```

Already calibrated “Accurate” ph Sensor

The Microbit code



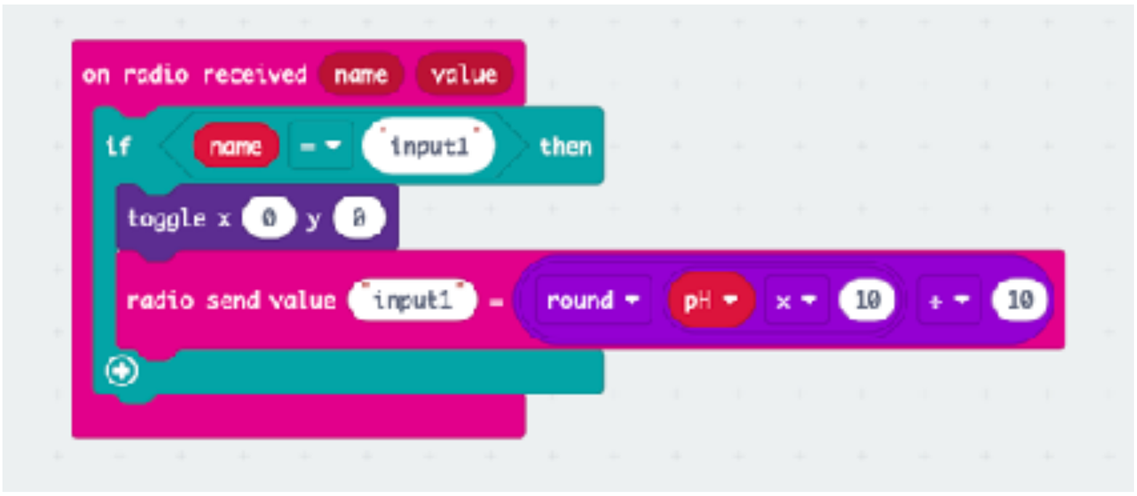
```
on start
    set offset_4_7 to 0.72
    set offset_10 to 1.21
    radio set group 6
    show leds

forever
    set voltage to analog read pin P0 × 3.3 ÷ 1024
    if voltage > 0 and voltage < 2.5 then
        set pH to voltage × 3.5 + offset_4_7
    else
        set pH to voltage × 3.5 - offset_10
    serial write value "pH" = round pH × 10 ÷ 10
    serial write value "voltage" = voltage
    log data column "PH" value pH
    pause (ms) 1000
```

The offset for the ph values were hardcoded values, that were determined by using some simple maths and a graph. The offset for any value under ph Value 7-8 would be added to the ph value.Any ph over 8 would have the offset taken away. The code used simple maths logic to determine whether voltage was in the offset4_7 category or offset_10 category. The ph would be logged into a CSV file using the data logger extension so that if when testing incase the broker could not be accessed that the values could still be retrieved from the micro:bit.

The code uses a continuous loop to read the analogue voltage value from pin P0, which is connected to the pH sensor. The voltage is converted into the pH value by applying specific formulas based on the range of values expected from the sensor. When the voltage falls between 0 and 2.5 bolts, it is converted to pH using the equation pH = voltage * 3.5 + offset_4_7'. However if the voltage is greater than that range it will be converted using this ph equation pH = voltage * 3.5 - offset_10'. The ph value is also rounded to one decimal before being displayed through a serial interface

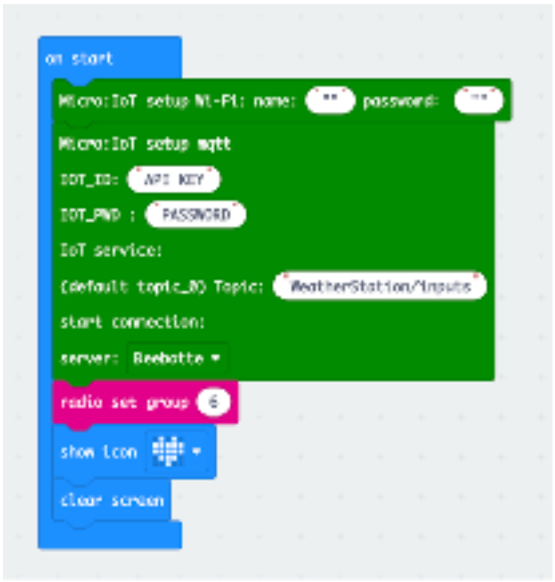
Implementing MQTT services with the sensor:

Being an IoT student this part of project was arguably the most important part to complete. Using the micro:bit ph sensor it was now time to create a way to send the data to another microbit that is connected to the internet that would publish the data to an MQTT broker. **Create flow chart.** The first step shown, sending the data to the microbit connected to the cloud was done by using radio. This would send the data from the ph sensor using this code:

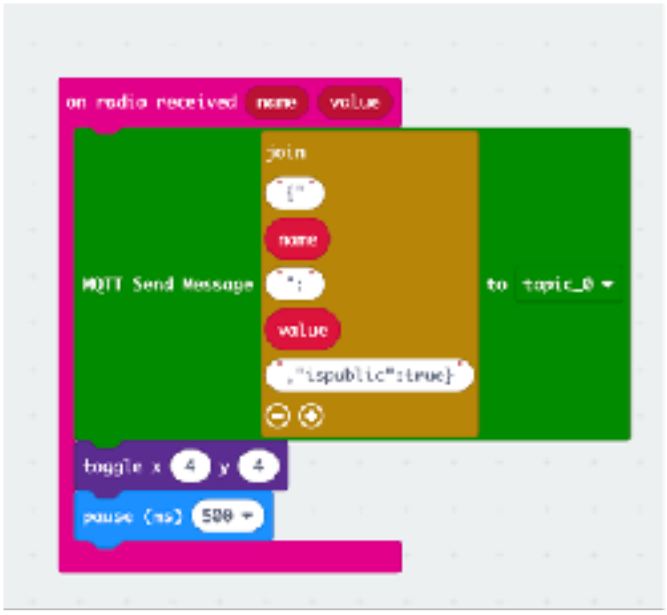


```
on radio received name value
    if name = "input1" then
        toggle x 0 y 0
        radio send value "input1" = round pH × 10 + 10
```

This code looks for the name "input1" in any code from any other microbit that is using the same radio group, the radio group is "6" in this case. It will send the ph value to this name only. The ph will then be picked up through radio by the cloud microbit. This cloud microbit contains code that connects to a an MQTT broker and sends the data up to said broker. The broker used for this was called beebotte. The data was sent in the form of JSON String. This string is published to the same topic in which the broker app is subscribed to.



```
on start
    Micro:IoT setup Wi-Fi: name: "" password: ""
    Micro:IoT setup mqtt
    IOT_ID: "API KEY"
    IOT_PWD : "PASSWORD"
    IoT service:
    (default topic_0) Topic: "WeatherStation/inputs"
    start connection:
    server: Beebotte
    radio set group 6
    show icon
    clear screen
```

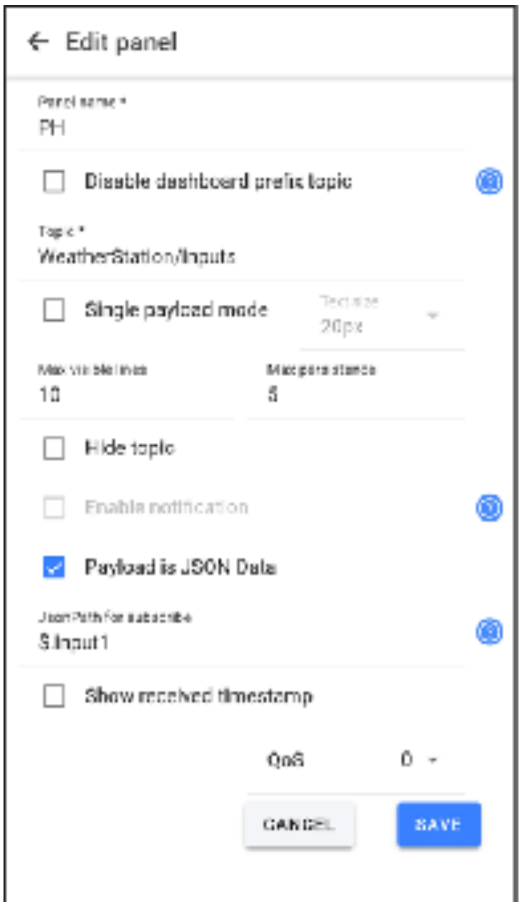


```
on radio received name value
    MQTT Send Message join "{"" name "":" value ","ispublic":true}" to topic_0
    toggle x 4 y 4
    pause (ms) 500
```

Real-Time Data Recording:

With the data now being successfully published to a broker, the next step was to subscribe to the same broker and retrieve the data. Using a MQTT client app such as the one below,

Using this app was straightforward to setup and meant wherever you are as long as theirs internet access, all the data can be accessed through this client app. The most important thing was making sure the app had subscribed to the correct topic, and was updating the data as soon as it was published. For the ph, a simple log table was used at first to retrieve any and all data sent, this needed a small amount of trouble shooting to ensure the correct payload was being sent.



Panel name *
PH

☐ Disable dashboard prefix topic

Topic *
WeatherStation/Inputs

☐ Single payload mode — Text size: 20px

Max visible lines: 10
Max persistence: 5

☐ Hide topic

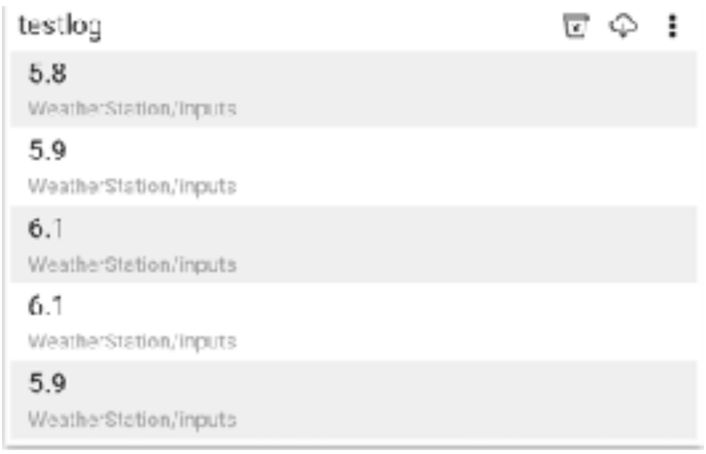
☐ Enable notification

☑ Payload is JSON Data

Json Path for subscribe
$.Input1

☐ Show received timestamp

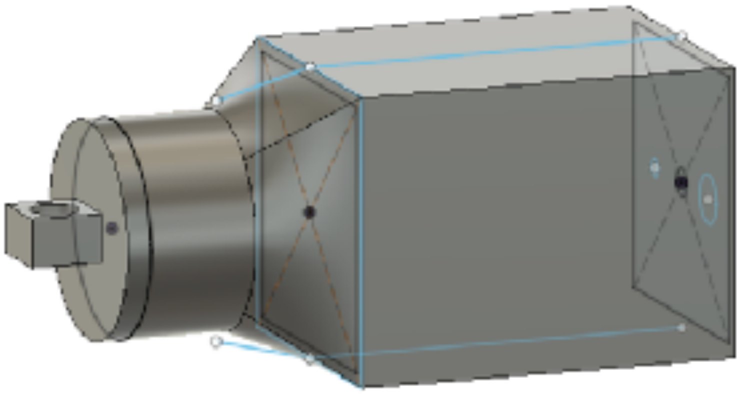
QoS: 0



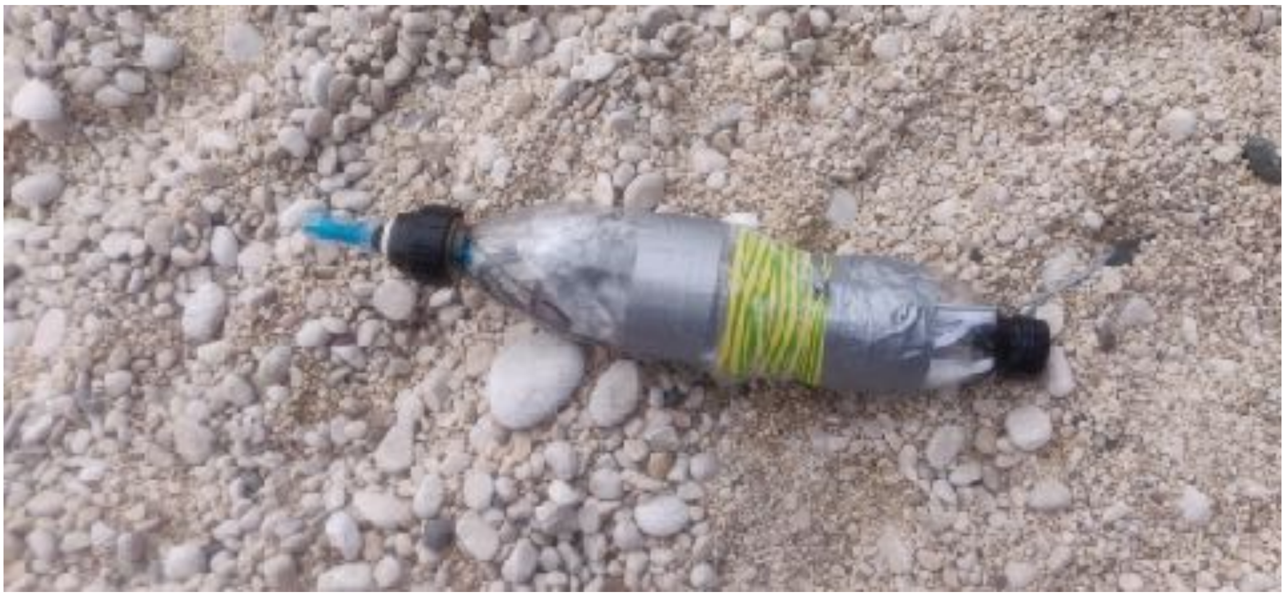
```
5.8
WeatherStation/Inputs
5.9
WeatherStation/Inputs
6.1
WeatherStation/Inputs
6.1
WeatherStation/Inputs
5.9
WeatherStation/Inputs
```

Sensor Pod Case

The final task of the project semester was too design an enclosure that would contain the sensor and could be used on the USV winch system to take readings in open waters. At first I used fusion 360 to draw up a draft of what the model could look like.



Though as time had become limited, instead I turned away from 3d printing an enclosure and thought towards a prototype made out of every day things, one of the senior engineers on the project helped out tremendously with this as realistically we only had a few hours for it to be finished in time. Using a battery pack to replace the plug meant the sensor was fully mobile and could theoretically be used anywhere.



This enclosure was tested for the first time on sight of all the main USV tests coordinated in Valencia. There was little time to actually attach the enclosure to he USV itself, and also the risk of damaging the electronics on the inside was too great for such a big first test. Instead I simply [laced the enclosure into the water by hand picking up any data being sent to the microbit cloud broker. There was some technical difficulties due to the data needed for the cloud not being at the most optimal. Though ph Values did show up, the cloud would fail more than once. However adding the load data function to the code allowed even with no access to the cloud to at least have the data being stored on the microbit itself for analysis later. Th sensor enclosure held up well until water began to leak through the bottle.

Project Conclusion and Future Recommendations

Conclusion:
In conclusion the project was a success, from start to finish. The aim of using solely microbes to control a ph sensor, sending its data over to another micro:bit to be published to a broker, to then be accessible by an MQTT client app was all achieved. Even with he problems of calibration and general troubleshooting errors with the broker, it all was functional by the end. Even the enclosure though rugged and very early in its use case development cycle was a successful first step in how an enclosure could function. The hands on practical nature of the module allowed for you to see some errors and make some more yourself, but then learn to solve them using your own knowledge that was gathered not through technically a lot of lecture type theory but experience from other practical classes through out the semester that prepared for this main task.

Future Recommendations:
My recommendation would be to try create a weather station sensor pod. With a much more refined enclosure, containing multiple sensors that can pick all kinds of data (temp, turbidity, pH, salinity, etc). And to have one broker with multiple topics that accommodate each sensor and its own data. Think of the weather station enclosure as a Swiss army knife of sensors.