S. SURYA PRASAD 18BCE1240
RS HARISH 18BME1018

SMART SHRIMP FARMING USING NODE RED

by

S SURYA PRASAD 18BCE1240
R S HARISH 18BME1018

A project report submitted to

Prof. Pradeep Kumar TS

SCHOOL OF COMPUTING SCIENCE AND ENGINEERING

in partial fulfilment of the requirements for the course of

CSE3009 – Internet of Things

in

B.Tech. COMPUTER SCIENCE AND ENGINEERING

VIT CHENNAI

Vandalur – Kelambakkam Road

Chennai – 600127

NOVEMBER 2020

S. SURYA PRASAD 18BCE1240
RS HARISH 18BME1018

ABSTRACT AND INTRODUCTION

Shrimp Farming is an aquaculture-based business that exists in either a marine or freshwater environment, producing prawns or shrimps. This project is aimed to built an IoT-based Smart Shrimp Farm.

Features of Smart Shrimp Farm:

1. 24-hour smart monitoring through MQTT- based IoT sensors
2. Reduces labour cost for monitoring ponds through efficient User Interface
3. Cloud repository for storing data
4. Ease of use and understanding by farmers through smartphones
5. Perform local analysis in cloud data for reducing costs in shrimp farms

Objectives of Smart Shrimp Farm:

1. Increased production of Shrimps
2. Ease of use for farmers
3. Reduction in costs
4. Produce healthier Shrimps

Optimal Conditions for Shrimp Farming:

1. Dissolved Oxygen: 3.5 - 4 ppm
2. Salinity: 10-25 ppt
3. Water Temperature: 26-32 Celsius
4. pH: 6.8-8.7
5. Total Nitrite Oxygen: 1 ppm
6. Total Ammonia: 1 ppm
7. Biological Oxygen Demand (BOD): 10 ppm
8. Chemical Oxygen Demand (COD): 70 ppm

9. Carbon dioxide: 10 ppm

10. Sulphide: 0.003 ppm

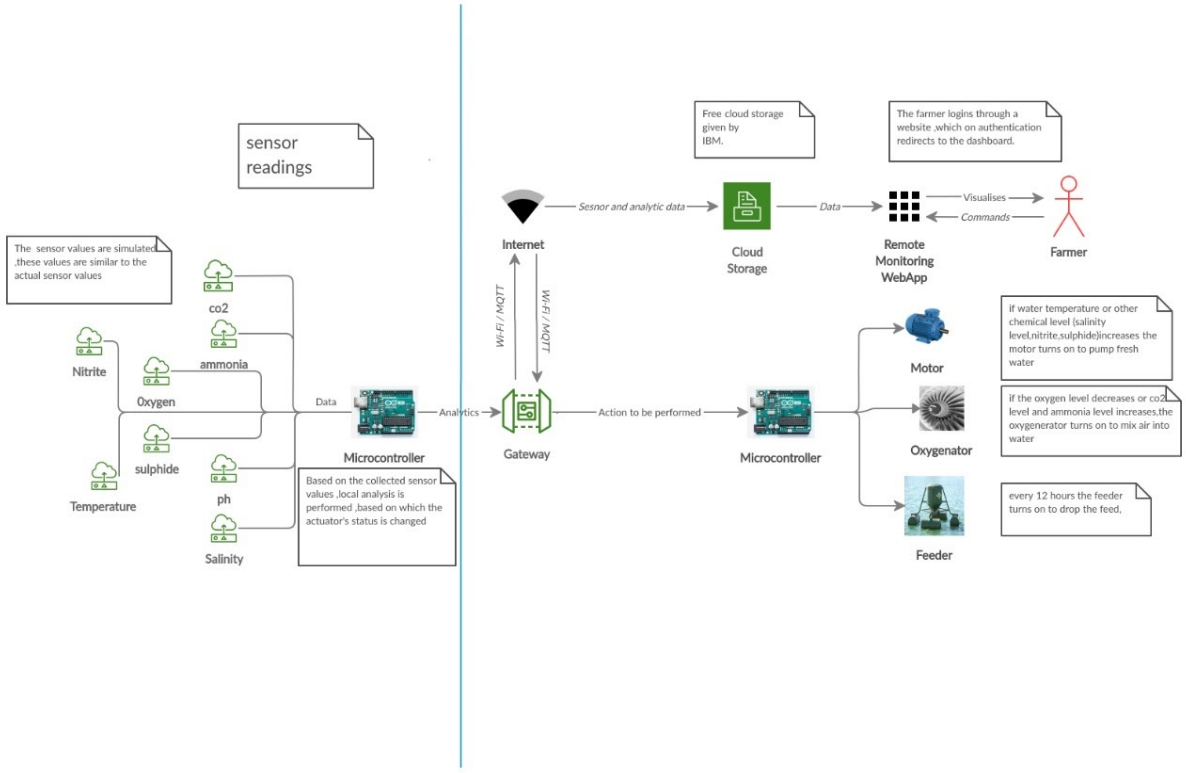
Since, the project is built using Node Red, the sensors values of the above conditions are randomized under specific ranges

SOFTWARE REQUIREMENTS:

1. Node Red: Node-RED is a flow-based development tool for visual programming developed originally by IBM for wiring together hardware devices, APIs and online services as part of the Internet of Things. Node-RED provides a web browser-based flow editor, which can be used to create JavaScript functions

2. IBM Cloud: IBM cloud computing service offered by IBM for storing the sensor values data

FLOW DIAGRAM:



S. SURYA PRASAD 18BCE1240
RS HARISH 18BME1018

WORKING:

1. The sensor values of the optimal conditions required for shrimp farming are randomized based on their ranges mentioned above.

2. All the values generated by the sensor nodes are collected at the coordinator node connected to the Internet.

3. The coordinator node through the MQTT protocol publishes the sensor data in the IBM Cloud.

4. The sensor values are stored in the IBM Cloud platform in the json format as key-value pairs.

5. The sensor values are subscribed from the IBM Cloud platform through the MQTT broker.

6. The sensor values are converted from the json format and are passed to the Motor, Oxygenator and Feeder Functions.

7. The status of Motor, Oxygenator and Feeder are switched on and off depending on the threshold limit of sensor values which is required for producing healthier shrimps**.**

8. The status of the Motor, Oxygenator and Feeder can be made automatic based on threshold values and it can also be handled manually by the farmer if desired.

9. The status of the Motor, Oxygenator and Feeder are stored in the IBM cloud for further cost analysis.

10. The sensor values along with the status of Motor, Oxygenator and Feeder are visualised in a dashboard in Node Red using graphs and charts dynamically by the farmer.

11. The results stored in the IBM Cloud of the status of Motor, Feeder, Oxygenator are retrieved and the cost of turning on the motor, feeder and oxygenator on a daily basis and monthly basis are calculated by setting a unit price for the operations costs. The results of the costs incurred are displayed to the farmer for further optimization.

S. SURYA PRASAD 18BCE1240
RS HARISH 18BME1018

Source Code (Sample)

Motor Function for switching on/off motor

Edit function node

Delete | Cancel | Done

Properties

Name: motor function

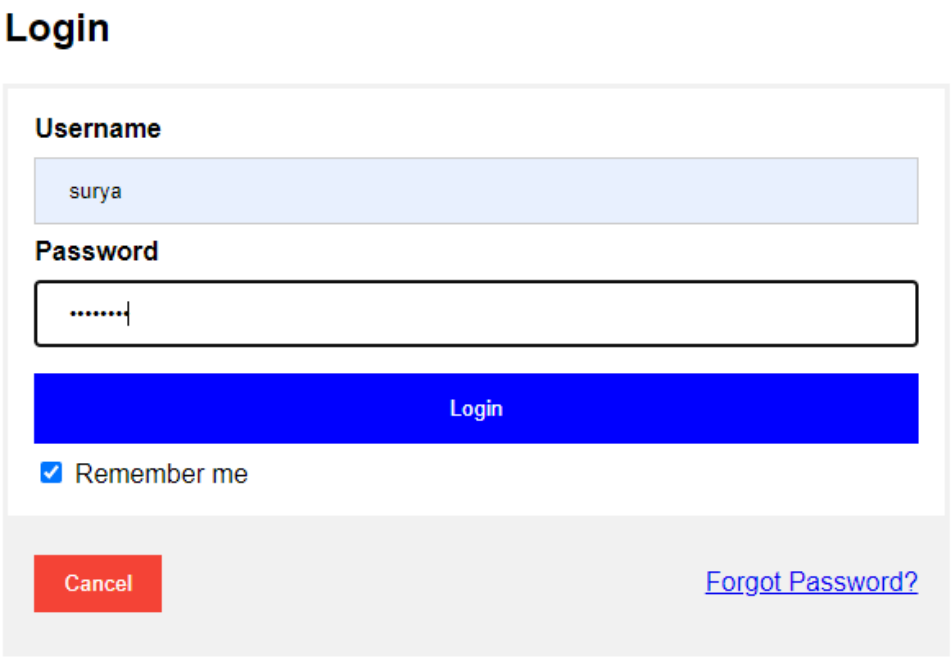
Setup | Function | Close

```
var payload=msg.payload;
var temp=payload.water_temperature;
var salt=payload.salinity;
var ammonia=payload.ammonia_level;
var ph=payload.ph;
var count1 = context.get('count1')||0;


if(temp>32||salt>25||ammonia>1||ph<6.5||ph>8){
  count1 += 1;
// store the value back
 context.set('count1',count1);
// make it part of the outgoing msg object

}


if(count1%5===0){ //trigger occured more than 5 times
    count1=0;
    context.set('count1',count1);
    msg.payload={"motor_status":"on"};
    //msg.payload={"motor_status":"on","count1":count1};

// make it part of the outgoing msg object;
}

else{
    msg.payload={"motor_status":"off"};
    //msg.payload={"motor_status":"off","count1":count1};

}




return msg;
```

S. SURYA PRASAD 18BCE1240
RS HARISH 18BME1018

PROJECT SCREENSHOTS

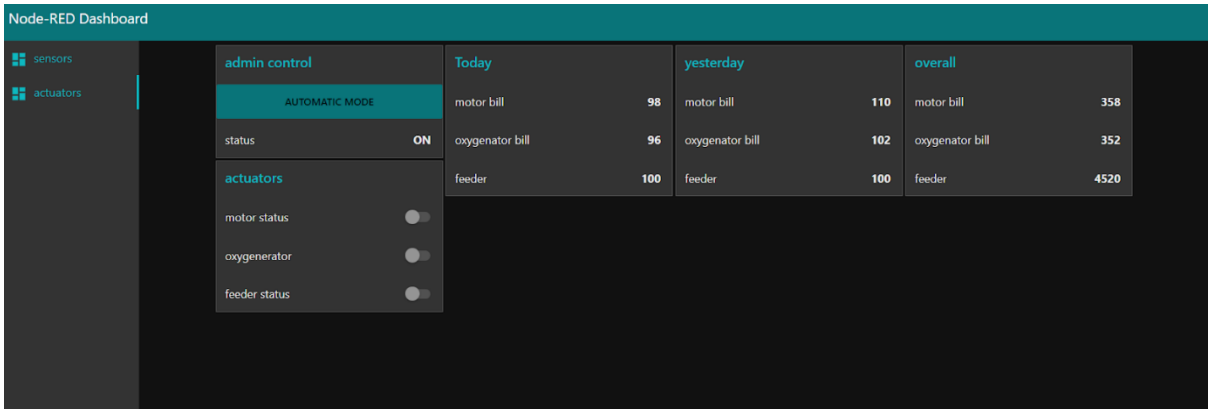
1. Login Page for Farmers



2. Dashboard

S. SURYA PRASAD 18BCE1240
RS HARISH 18BME1018

3. Admin Panel and Cost Analysis



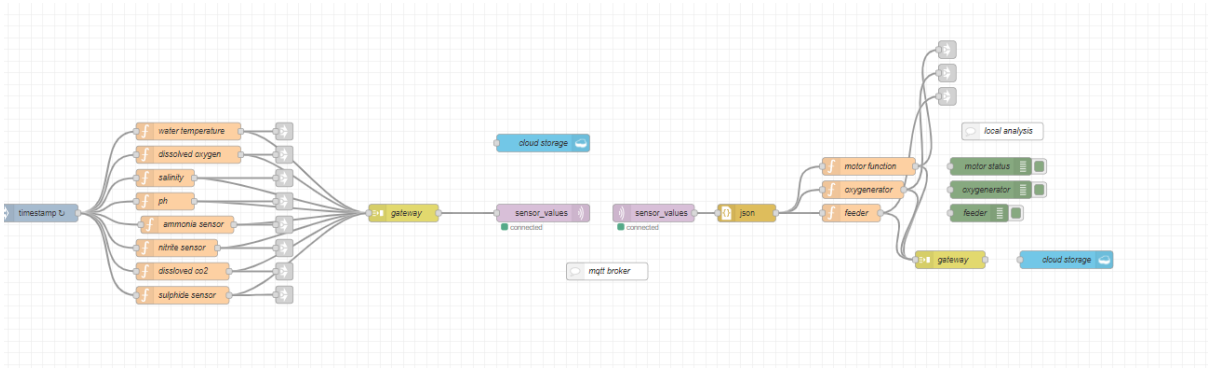
4. IBM Cloud Platform

sensor_values > 05dab40b1b0562ac69023b89e30e4d64

Save Changes Cancel


```
{
  "_id": "05dab40b1b0562ac69023b89e30e4d64",
  "_rev": "1-2a08304c6834fe8dc5e817c229b27844",
  "water_temperature": "30.3",
  "dissolved_oxygen": "4.4",
  "salinity": "27.4",
  "ph": "6.7",
  "ammonia_level": "0.902",
  "nitrite_level": "0.7",
  "dissolved_co2": "10.8",
  "sulphide_level": "0.00639"
}
```


5. Node Red Implementation



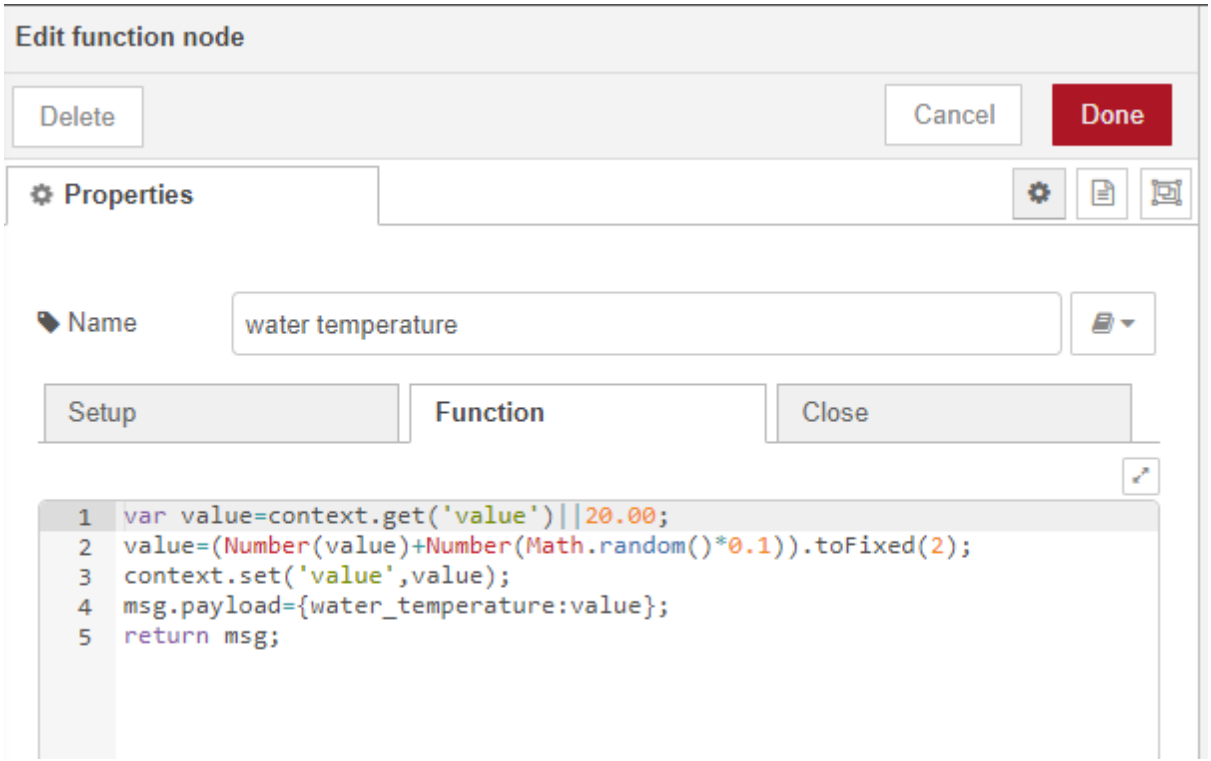
S. SURYA PRASAD 18BCE1240
RS HARISH 18BME1018

CONCLUSION

This project can help in improving the productivity and efficiency of the shrimps produced which increases the profit aimed by farmers in aqua-farming industries. Further analysis could be carried out on the sensor values data stored in Cloud to identify the different types of shrimps that can be grown in different conditions to improve productivity and cost earnings of people living in different climatic conditions. If possible, similar project can be carried out in farming at homes like terrace farming for improving the employability of women in India.

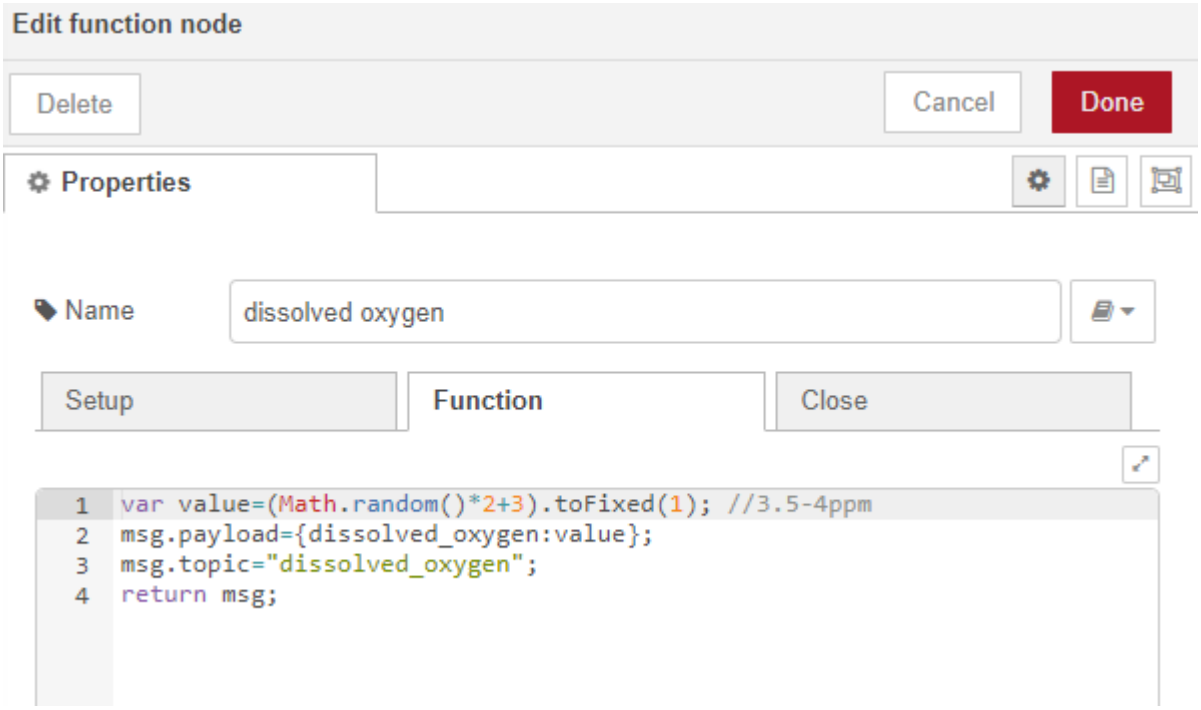
APPENDIX – Source Code

1. Water Temperature

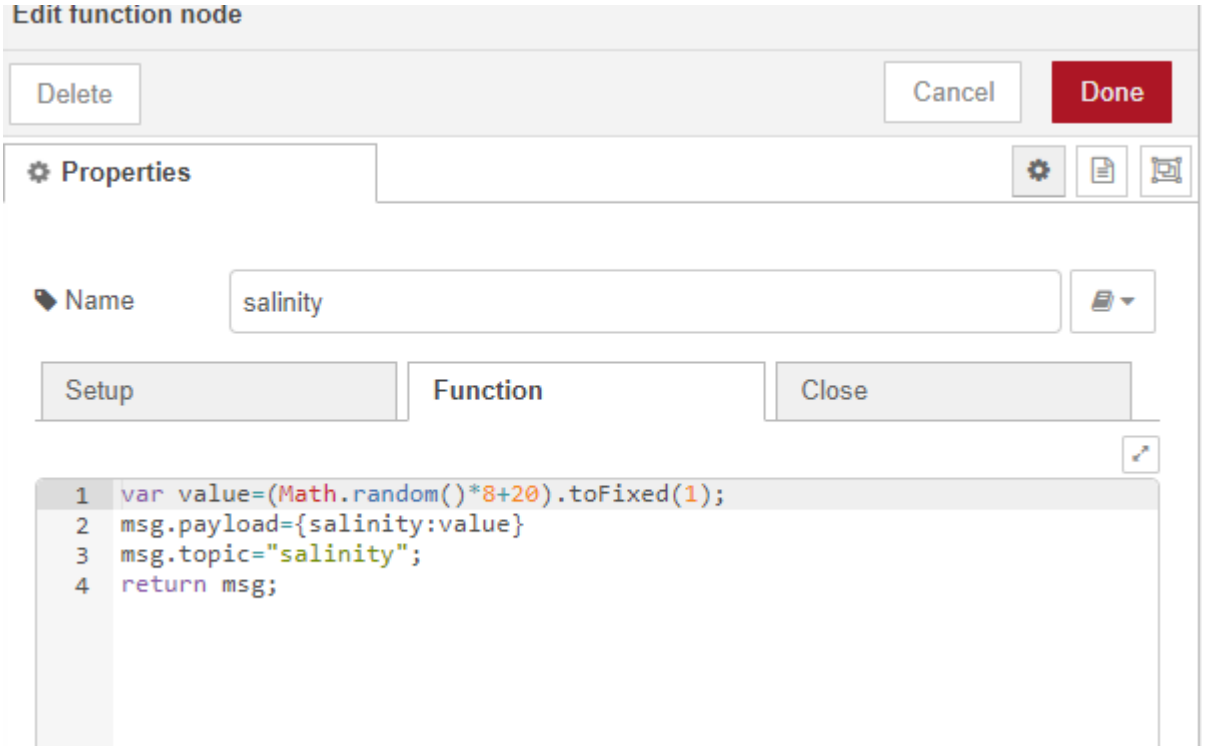


S. SURYA PRASAD 18BCE1240
RS HARISH 18BME1018

2. Dissolved Oxygen



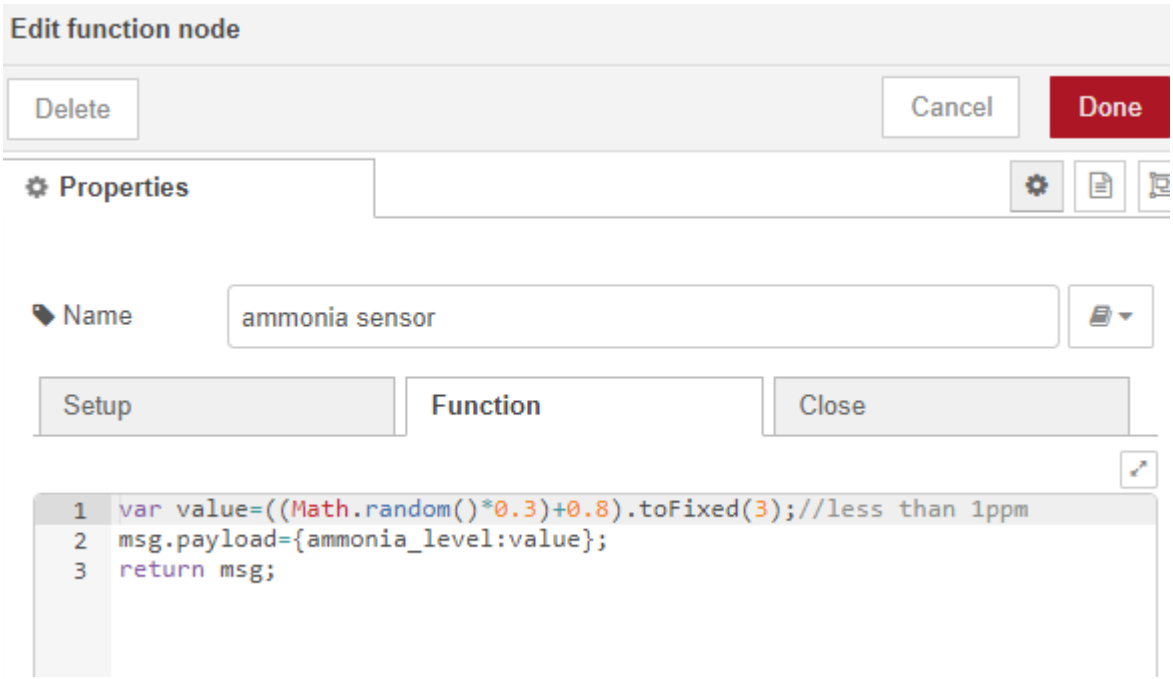
3. Salinity



S. SURYA PRASAD 18BCE1240
RS HARISH 18BME1018

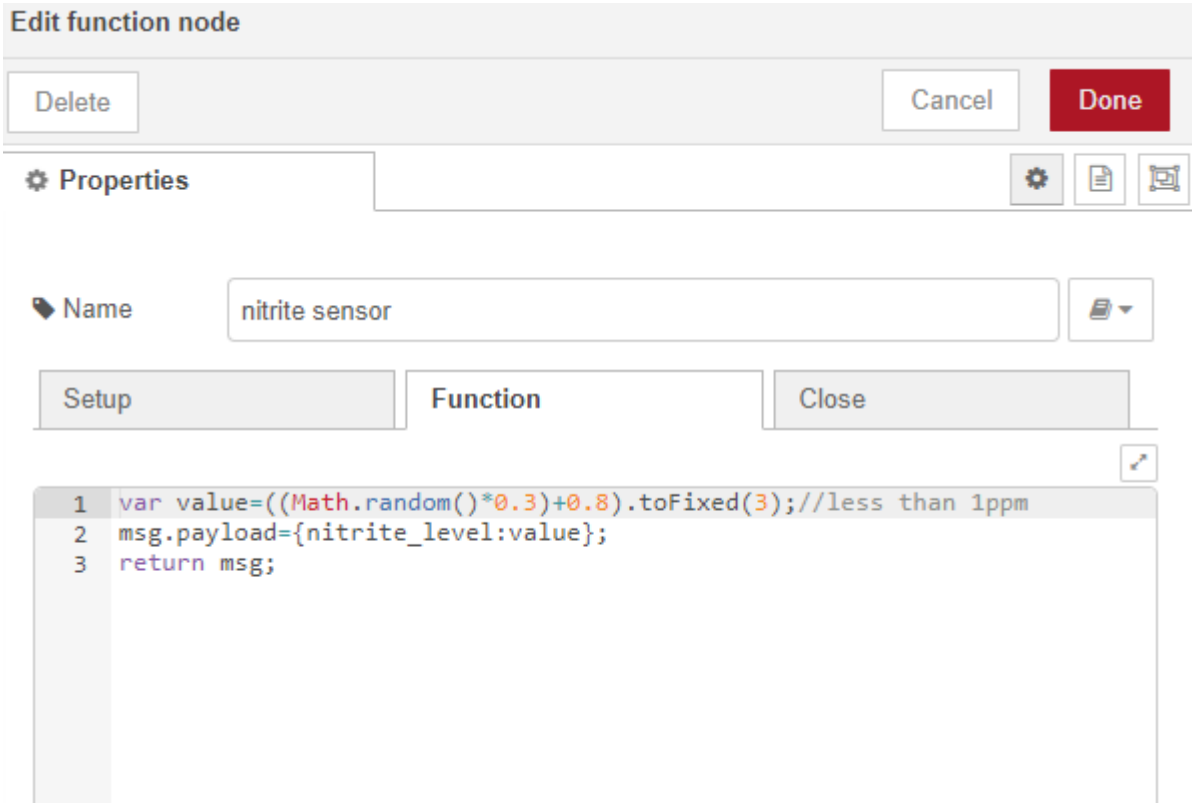
4. pH Value

5. Ammonia Sensor



S. SURYA PRASAD 18BCE1240
RS HARISH 18BME1018

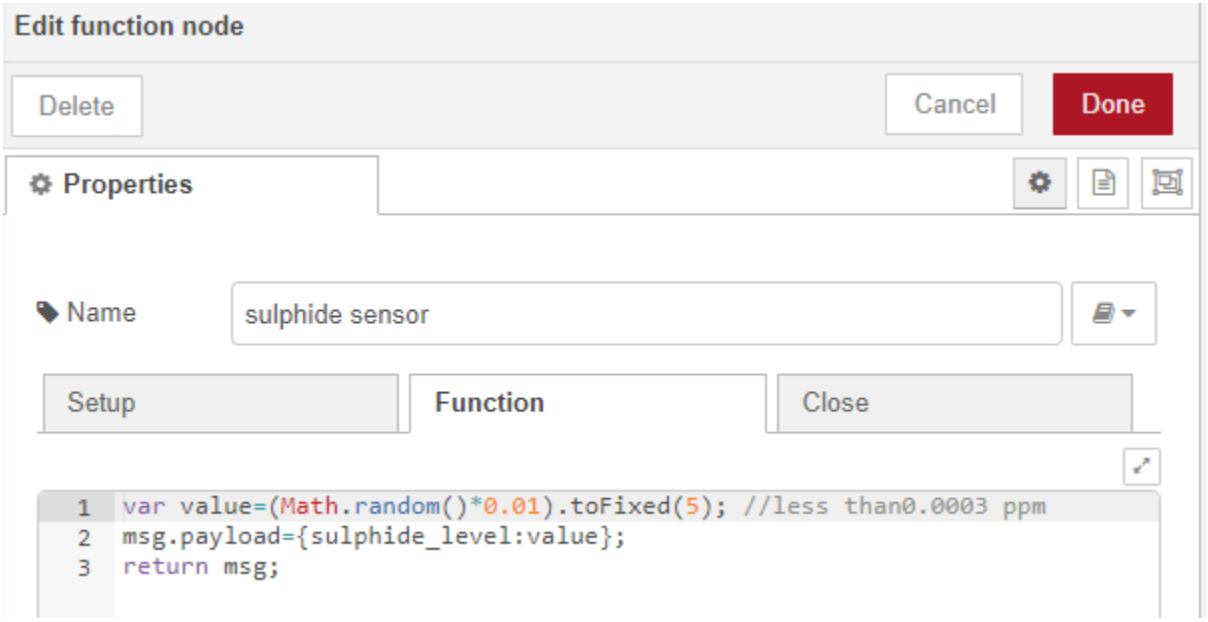
6. Nitrite Sensor



7. Dissolved CO2

S. SURYA PRASAD 18BCE1240
RS HARISH 18BME1018

8. Sulphide Sensor



S. SURYA PRASAD 18BCE1240
RS HARISH 18BME1018

9. Motor Function

S. SURYA PRASAD 18BCE1240
RS HARISH 18BME1018

10. Oxygenator

Edit function node

Delete | Cancel | Done

Properties

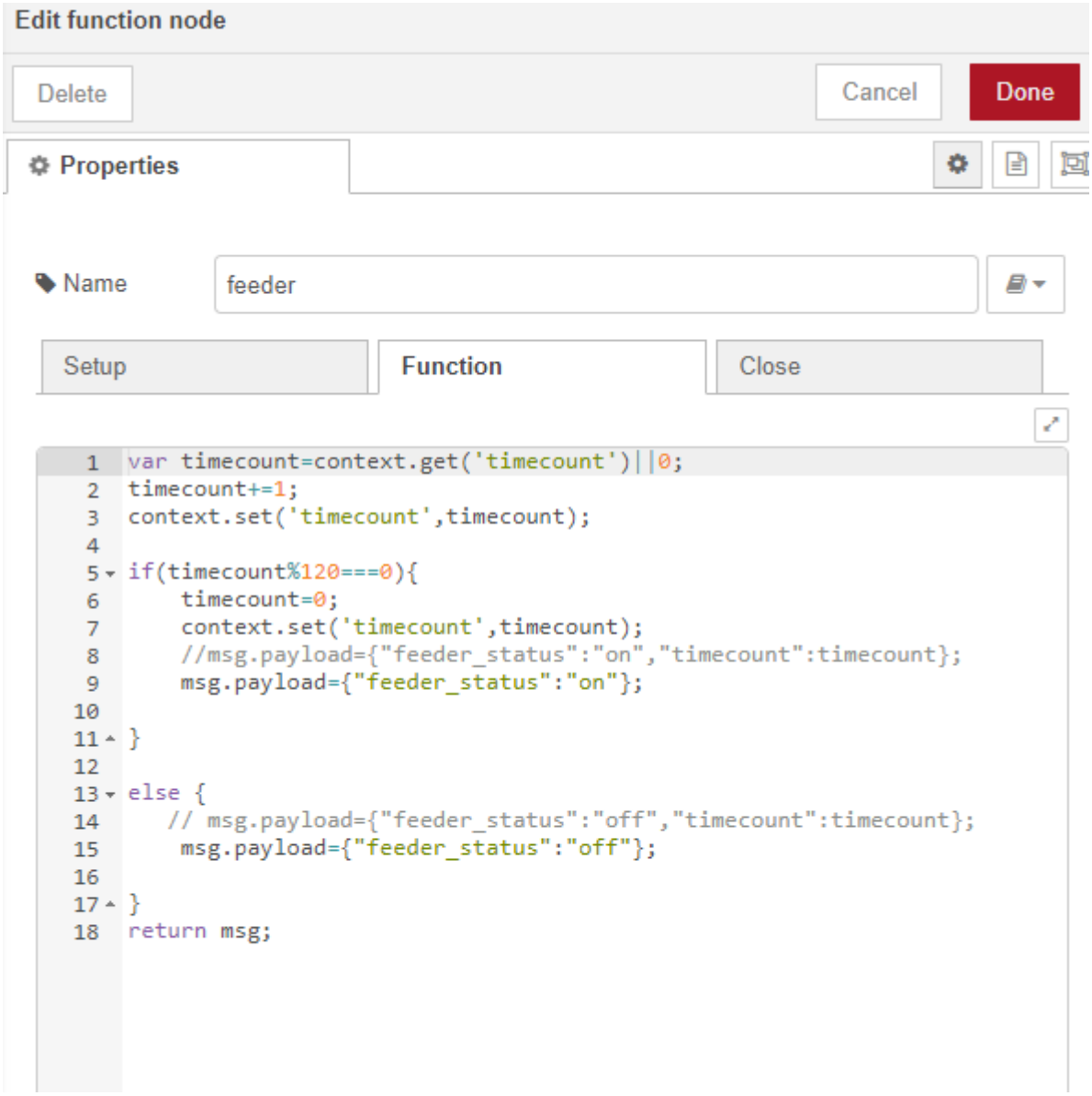
Name: oxygenerator

Setup | Function | Close

```
var payload=msg.payload;
var sulphide=payload.sulphide_level;
var nitrite=payload.nitrite_level;
var oxygen=payload.dissolved_oxygen;
var co2=payload.dissolved_co2;

var count2=context.get("count2")||0;

if(oxygen<3.5 ||oxygen>4||nitrite>1||sulphide>1){
   count2+=1;

    context.set('count2',count2);

}


if(count2%5===0){ //trigger occured more than 5 times
    count2=0;

    context.set('count2',count2);
    msg.payload={"oxygenerator":"on"};

   // msg.payload={"oxygenerator":"on","count2":count2};

// make it part of the outgoing msg object;
}

else{
    msg.payload={"oxygenerator":"off"};
    //msg.payload={"oxygenerator":"off","count2":count2};

}




return msg;
```

S. SURYA PRASAD 18BCE1240
RS HARISH 18BME1018

11. Feeder



S. SURYA PRASAD 18BCE1240
RS HARISH 18BME1018

12. Cost Analysis

```
var length = msg.payload.length;
var i;
var count_motor0 = 0;
var count_motor1 = 0;
var count_motor2 = 0;
var count_oxygenator0 = 0;
var count_oxygenator1 = 0;
var count_oxygenator2 = 0;

//today
for (i = 0; i < 240; i++) {
   var input = msg.payload[i];
   if (input.motor_status == "on")
      count_motor1++;
   if (input.oxygenerator == "on")
      count_oxygenator1++;
}
var cost_motor1 = count_motor1 * 2;
var cost_oxygenator1 = count_oxygenator1 * 2;
var cost_feeder1 = 2 * 50;

//yesterday
for (i = 241; i < 480; i++) {
   var input = msg.payload[i];
   if (input.motor_status == "on")
      count_motor2++;
   if (input.oxygenerator == "on")
      count_oxygenator2++;
}
var cost_motor2 = count_motor2 * 2;
var cost_oxygenator2 = count_oxygenator2 * 2;
var cost_feeder2 = 2 * 50;

//overall
for (i = 0; i < length; i++) {
   var input = msg.payload[i];
   if (input.motor_status == "on")
      count_motor0++;
   if (input.oxygenerator == "on")
      count_oxygenator0++;
```

```
}
var cost_motor0 = count_motor0 * 2;
var cost_oxygenator0 = count_oxygenator0 * 2;
var cost_feeder0 = length * 5;


msg.payload = {motor0: cost_motor0, oxygenator0: cost_oxygenator0, feeder0:
cost_feeder0, motor1: cost_motor1, oxygenator1: cost_oxygenator1, feeder1:
cost_feeder1, motor2: cost_motor2, oxygenator2: cost_oxygenator2, feeder2:
cost_feeder2};

return msg;
```

REFERENCES:

1. **https://www.zdnet.com/article/how-the-iot-is-helping-shrimp-farmers-get-healthy-bumper-crops/**

2. **https://ieeexplore.ieee.org/document/8365307**