

## SPRINT 4

DATE	9 NOVEMBER 2022
TEAM ID	PNT2022TMID25057
PROJECT NAME	IOT Based Real -Time River Water Quality Monitoring and Control System

PYTHON SCRIPT:

```
import random
import time
import sys
import ibmiotf.application
import ibmiotf.device

# Provide your IBM Watson Device Credentials
organization = "f5rl2v" # repalce it with organization ID
deviceType = "weather_device" # replace it with device type
deviceId = "weather_today" # repalce with device id
authMethod = "token"
authToken = "2VcVpo)hG4rnKKIG)x" # repalce with token
import os

from twilio.rest import Client
account_sid = 'ACb4d033465895822c34e656bf6be69384' auth_token =
'6916b3bf66a451937068378db5a9692a' client = Client(account_sid, auth_token)

def send_sms():
    message = client.messages.create(
        messaging_service_sid='MG3d02a8b50e684c345993182
610957703',
```

```

body='Alert the water is not in good quality
!',
from_='+16294006922', to='+9199992344234'
)
print(message.sid)
def myCommandCallback(cmd):
    print("Command received: %s" % cmd.data)
    if cmd.data['command'] == 'motoron':
        print("MOTOR ON")
    elif cmd.data['command'] == 'motoroff':
        print("MOTOR OFF")
try:
    deviceOptions = {"org": organization, "type": deviceType, "id": deviceId, "auth-method":
authMethod,
    "auth-token": authToken}
    deviceCli = ibmiotf.device.Client(deviceOptions)

# .....
except Exception as e:
    print("Caught exception connecting device: %s" % str(e))
    sys.exit()
deviceCli.connect()
while True:
    pH = random.randint(0,14)
    conductivity = random.randint(0,80)
    T = random.randint(0,100)
    oxygen = random.randint(0,80)
    turbidity = random.randint(0,100)
    # Send Temperature & Humidity to IBM Watson
    data = {"Ph":pH,'temperture': T,'turbidity':turbidity,'oxygen':oxygen}
    # print data

```

```

def myOnPublishCallback():
    print("Data publish ",data, "to IBM Watson")
    success = deviceCli.publishEvent("event", "json", data, 0, myOnPublishCallback)
    if not success:
        print("Not connected to IoT")
    time.sleep(5)
    deviceCli.commandCallback = myCommandCallback

```

ARDUINO SCRIPT:

```

#include "SPI.h"

Adafruit_ILI9341 tft = Adafruit_ILI9341(TFT_CS, TFT_DC);

#include libraries

#include <SoftwareSerial.h>
#include <LiquidCrystal.h>
#include "Adafruit_GFX.h"
#include "Adafruit_ILI9341.h"

//decraration of all our variables

float reads;

int pin = A0;

float vOut = 0 ;//voltage drop across 2 points

float vIn = 5;

float R1 = 1000;

float R2 = 0;

float buffer = 0;

float ph;

float R = 0;//resistance between the 2 wires

float r = 0;//resistivity

float L = 0.06;//distance between the wires in m

```

```

double A = 0.000154;//area of cross section of wire in m^2

float C = 0;//conductivity in S/m

float Cm = 0;//conductivity in mS/cm

int rPin = 9;

int bPin = 5;

int gPin = 6;

int rVal = 255;

int bVal = 255;

int gVal = 255;

//creating lcd object from Liquid Crystal library
LiquidCrystal lcd(7,8,10,11,12,13);

void setup() {

  //initialise IOT and serial monitor
  Serial.begin(9600);
  BTserial.begin(9600);

  //initialise lcd
  lcd.begin(16, 2);

  //set rgb led pins (all to be pwm pins on Arduino) as output
  pinMode(rPin,OUTPUT);
  pinMode(bPin,OUTPUT);
  pinMode(gPin,OUTPUT);
  pinMode(pin,INPUT);

  //Print stagnant message to LCD
  lcd.print("Conductivity: ");
}

void loop() {

  reads = analogRead(A0);

  //display corresponding colours on rgb led according to the analog read

```

```
if( reads < 600 )
{
  if (reads <= 300){
    setColor( 255, 0, 255 ) ;
  }
  if (reads > 200){
    setColor( 200, 0, 255 ) ;
  }
}
else{
  if( reads <= 900 )
  {
    setColor( 0, 0, 255 ) ;
  }
  if( reads > 700 )
  {
    setColor( 0, 255, 255 ) ;
  }
}
```

```
void setColor(int red, int green, int blue)
{
  analogWrite( rPin, 255 - red ) ;
  analogWrite( gPin, 255 - green ) ;
  analogWrite( bPin, 255 - blue ) ;
}
```

HTML SCRIPT:

```
<!DOCTYPE html>
```

```
<html>
```

```
<head>
```

```
<h1> Real time water quality monitoring system</h1>

<metaname="viewport" content="width=device-width, initial-scale=1">

<style>

body {font-family: Arial,Impact, 'Arial Narrow Bold', sans-serif, sans-serif;}

/* Full-width input fields */

input[type=text], input[type=password] {

width: 150;

padding: 23px 24px;

margin: 8px 0;

display: inline-block;

border: 1px solid #ccc;

box-sizing: border-box;

}

/* Set a style for all buttons */

button {

background-color: #04AA6D;

color:blue;

padding: 15px 21px;

margin: 8px 0;

border: none;

cursor: pointer;

width: 102;

}

button:hover {

opacity: 0.7;

}

/* Extra styles for the cancel button */

.cancelbtn {

width: min-content

padding: 10px 18px;

background-color: #f4455f
```

```

}

/* Center the image and position the close button */
{.imgcontainer { }
text-align: right;
margin: 24px 0 12px 0;
position: relative

}

img {Real time water quality monitoring and control system}: {
width: 56;
border-radius:50%;
}

.container {
padding: 16px;
}

span.psw {
float: right;
padding-top: 16px;
}

/* The Modal (background) */
.modal {
display: none; /* Hidden by default */
position: fixed; /* Stay in place */
z-index: 1; /* Sit on bottom*/
left: 0;
top: 0;
width: 100%; /* full width */
height: 100%; /* medium height */
overflow: auto; /* Enable scroll if needed */
background-color: ybg(0,0,0); /* Fallback color */
background-color: rgba(0,0,0,0.4); /* Black w/ transparent */

```

```
padding-top: 60px;
}

/* Modal Content/Box */
.modal-content {
background-color: #fefefe;

margin: 5% auto 15% auto; /* 5% from the top, 15% from the bottom and centered */

border: 1px solid #888;

width: 65%; /* Could be more or less, depending on screen size */
}

/* The Close Button (x) */
.close {
position: absolute;

right: 25px;

top: 0;

color: #888;

font-size: 35px;

font-weight: initial;


}

.close:hover,
.close:focus {
color: red;

cursor: pointer;
}

/* Add Zoom Animation */
.animate {
-webkit-animation: animatezoom 0.6s;

animation: animatezoom 0.6s
}

@-webkit-keyframes animatezoom {
from {-webkit-transform: scale(0)}
```



```
to {-webkit-transform: scale(1)}  
}
```

```
@keyframes animatezoom {  
  from {transform: scale(2)}  
  to {transform: scale(1)}  
}
```

```
/* Change styles for span and cancel button on extra small screens */
```

```
@media screen and (max-width: 300px) {  
  span.psw {  
    display: block;  
    float: none;  
  }  
  .cancelbtn {  
    width: 100%;  
  }  
}
```

```
</style>
```

```
</head>
```

```
<body>
```

```
<h2>Modal Login Form</h2>
```

```
<button onclick="document.getElementById('id01').style.display='block'"  
style="width:auto;">Login</button>
```

```
<div id="id01" class="modal">
```

```
<form class="modal-content animate" action="/action_page.php" method="post">
```

```
<div class="imgcontainer">
```

```
60
```

```
<span onclick="document.getElementById('id01').style.display='none'" class="close"  
title="Close Modal">&times;</span>
```

```
</div>
```

```
<div class="container">

<label for="uname"><b>Username</b></label>

<input type="text" placeholder="Enter Username" name="uname" required>


<label for="psw"><b>Password</b></label>

<input type="password" placeholder="Enter Password" name="psw" required>

<label for="captch"></label><123gh@></label>

<input type="captcha" 123@g="Enter captcha" name="captcha" required>

<button type="submit">Login</button>

</label>


<input type="checkbox" checked="checked" name="remember"> Remember me

</label>

</div>
```

```
<div class="container" style="background-color:#f1f1f1">

<button type="button" onclick="document.getElementById('id01').style.display='none'"
class="cancelbtn">Cancel</button>

<span class="psw">Forgot <a href="#">password?</a></span>

</div>

</form>

</div>

<script>

// Get the modal
var modal = document.getElementById('id03');

// When the user clicks anywhere outside of the modal, close it
window.onclick = function(event) {
  if (event.target == modal) {
    modal.style.display = "none";
  }
}
```

</script>

</body>

</html>

## APPENDIX 2

### ASSIGNMENT 1 TITLE:

Build a smart home in tinker cad Use at least Solution: 2 sensors, led, buzzer in a circuit.

Simulate in asingle code.

CODE:

```
// C++ code
```

```
//
```

```
#include<Servo.h>
```

```
#define LED 13
```

```
#define FAN 10
```

```
#define TEMP A0
```

```
#define BUZZER 11
```

```
#define PIR 12
```

```
#define DOOR 5
```

```
#define TRIGGER 6
```

```
#define ECHO 7
```

```
#define TRIGGER1 9
```

```
#define ECHO1 8
```

```
Servo S;
```

```
void setup()
```

```
{
```

```
Serial.begin(9600);
```

```
pinMode(LED,OUTPUT);
```

```
pinMode(FAN,OUTPUT);
```

```
pinMode(BUZZER,OUTPUT);
```

```

pinMode(PIR,INPUT);
pinMode(DOOR,OUTPUT);
pinMode(TRIGGER,OUTPUT);
pinMode(ECHO,INPUT);
pinMode(TRIGGER1,OUTPUT);
pinMode(ECHO1,INPUT);
S.attach(DOOR);
S.write(90);
}
void loop()
{
//Car Garage
digitalWrite(TRIGGER,0);
digitalWrite(TRIGGER,1);
delayMicroseconds(10);
digitalWrite(TRIGGER,0); float
d = pulseIn(ECHO,1); float l =
(d*0.0343)/2;
int m = map(l,0,330,0,255);
if(m<=50)
{ tone(BUZZER,294,700);
delay(1000);
noTone(BUZZER);
Serial.println("Buzzer horn when Car parked");
}
else
analogWrite(BUZZER,0);
//Door Open
int z = digitalRead(PIR);
delay(1000);
if(z==1)

```

```

{
S.write(0);
Serial.println("Door Opened");
delay(3000);
S.write(90);
delay(1000);
}
else
{
S.write(90);
delay(1000);
}
digitalWrite(TRIGGER1,0);
digitalWrite(TRIGGER1,1);
delayMicroseconds(10);
digitalWrite(TRIGGER1,0);
float d1 = pulseIn(ECHO1,1);
float l1 = (d1*0.0343)/2;
if(l1<330)
{
//IN ROOM
Serial.println("Person in Room");
digitalWrite(LED,1);
double a = analogRead(TEMP);
double t = (((a/1024)*5)-0.5)*100;
int s = map(t,-40,120,0,255);
if(s>100)
analogWrite(FAN,s);
delay(2000);
}
else

```

```
{  
  digitalWrite(LED,0);  
  analogWrite(FAN,0);  
}  
}
```