Coding and Solution

Team ID	PNT2022TMID11410
Project Name	Real-time river water quality monitoring and control system

Code Layout

```
#include
<WiFi.h>
#include
<PubSubCl
ient.h>
#include "DHT.h"// Library for dht11
#define DHTPIN 15     // what pin we're
connected to #define DHTTYPE DHT22
                      // define type of
sensor DHT 11DHT dht (DHTPIN, DHTTYPE);
void callback(char* subscribetopic, byte*
payload, unsigned intpayloadLength);

WiFiCl
ient
wifiCl
ient;
String
data3;
#define
ORG
"ks8pti"
#define
DEVICE_TYP
E "ESP32"
#define
DEVICE_ID
"143143"
#define TOKEN "123456789"
#define speed 0.034
#define led 14
```

```
char server[] = ORG
".messaging.internetofthings.ibmcloud.com";char
publishTopic[] = "iot-2/evt/Data/fmt/json";
char topic[] = "iot-
2/cmd/command/fmt/String";char
authMethod[] = "use-token-
auth";
char token[] = TOKEN;
char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;
PubSubClient client(server,
1883, wifiClient);void
publishData();


const
int
trigp
in=5;
const
int
echop
in=18
;
Strin
g
comma
nd;
Strin
g
data=
"";

l
o
n
g

d
u
r
a
t
i
o
n
```

```
;

f
l
o
a
t

d
i
s
t
;

f
l
o
a
t

T
e
m
p
;

i
n
t

p
H
;



void setup()

{
  Serial.be
  gin(11520
  0);
  dht.begin
  ();
  pinMode(l
```

```
  ed,
  OUTPUT);
  pinMode(t
  rigpin,
  OUTPUT);
  pinMode(e
  chopin,
  INPUT);
  wifiConne
  ct();
  mqttConne
  ct();
}

void loop() {
  bool isNearby
  = dist < 100;
  digitalWrite(
  led,
  isNearby);

  pH = dht.readHumidity();
  Temp =
  dht.readTempe
  rature();
  Serial.print(
  "Temperature:
  ");
  Serial.printl
  n(Temp);
  Serial.print(
  "Tubidity:");
  Serial.printl
  n(pH);

  p
  u
  b
  l
  i
  s
  h
  D
  a
  t
```

```
  a
  (
  )
  ;

  d
  e
  l
  a
  y
  (
  1
  0
  0
  0
  )
  ;
  if
    (!c
    lie
    nt.
    loo
    p()
    ) {
    mqt
    tCo
    nne
    ct(
    );
  }
}

void wifiConnect() {
  Serial.print("Connecting to ");
  Serial.print("Wifi");
  WiFi.begin("Wokwi-GUEST", "", 6);
  while (WiFi.status() !=
    WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.print("WiFi connected, IP address: ");
Serial.println(WiFi.localIP());
}
```

```
void mqttConnect() {
  if (!client.connected()) {
    Serial.print("Reconnecting MQTT client to ");
    Serial.println(server);while
    (!client.connect(clientId, authMethod, token)) {
      Se
      ri
      al
      .p
      ri
      nt
      ("
      ."
      );
      de
      la
      y(
      50
      0)
      ;
    }
    initManagedDevice();
    Serial.println();

  }
}

void initManagedDevice() {
  if (client.subscribe(topic)) {
    // Serial.println(client.subscribe(topic));
    Serial.println("IBM subscribe to cmd OK");
  } else {
    Serial.println("subscribe to cmd FAILED");
  }
}
void publishData()
{
  digitalWrite(tr
  igpin,LOW);
  digitalWrite(tr
  igpin,HIGH);
  delayMicrosecon
  ds(10);
  digitalWrite(tr
  igpin,LOW);
```

```
duration=pulseI
n(echopin,HIGH)
;
dist=duration*s
peed/2;
if(dist<100){
  String payload =
  "{\"Turbidity\":";
  payload += dist;
  payload +=
  ","\"Temperature
  \":";payload +=
  Temp;
  payload
  += ","
  "\"pH\":"
  ;payload
  += pH;
  payload += "}";

  Serial.print("\n")
  ;
  Serial.print("Send
  ing payload: ");
  Serial.println(pay
  load);
    if(client.publish(publishTopic, (char*)
    payload.c_str())) { Serial.println("Warning crosses
    110cm -- it automaticaly of the loop");
    digitalWrite(led,HIGH);
  }

}
  if(dist>101 && dist<111){
  String payload =
  "{\"Normal Distance\":";
  payload += dist;
  payload += "}";



  Serial.print("\n")
  ;
  Serial.print("Send
  ing payload: ");
```

```
    Serial.println(pay
    load);

    }

  }
```

```
  void callback(char* subscribeTopic, byte*
payload, unsigned intpayloadLength){
  Serial.print("callback invoked for topic:");
  Serial.println(subscr
  ibeTopic); for(int
  i=0; i<payloadLength;
  i++){
    dist += (char)payload[i];
  }
  Serial.println(
  "data:"+
  data3);
  if(data3=="ligh
  ton"){
    Serial.println(data3);
    digitalWrite(led,HIGH);
  }
  data3="";
}
```

01. DESIGN

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Integer nec odio. Praesent libero. Sed cursus ante dapibus diam. Sed nisi. Nulla quis sem at nibh elementum imperdiet.

02. DEVELOP

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Integer nec odio. Praesent libero. Sed cursus ante dapibus diam. Sed nisi. Nulla quis sem at nibh elementum imperdiet.

03. TEST

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Integer nec odio. Praesent libero. Sed cursus ante dapibus diam. Sed nisi. Nulla quis sem at nibh elementum imperdiet.

04. DELIVER

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Integer nec odio. Praesent libero. Sed cursus ante dapibus diam. Sed nisi. Nulla quis sem at nibh elementum imperdiet.

05. RINSE

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Integer nec odio. Praesent libero. Sed cursus ante dapibus diam. Sed nisi. Nulla quis sem at nibh elementum imperdiet.

06. REPEAT

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Integer nec odio. Praesent libero. Sed cursus ante dapibus diam. Sed nisi. Nulla quis sem at nibh elementum imperdiet.

Code Readability and Reusability

- This code can easy to read and understand everythingfaster.
- In this code we can reuse every part code

Python Random Value Generation Code

```
i
m
p
o
r
t

t
i
m
e
```

```
i
m
p
o
r
t

s
y
s
import
ibmiotf.applic
ationimport
ibmiotf.devic
e import
random
```

```
#Provide your IBM Watson
Device Credentialsorganization =
"ks8pti"
deviceType =
"ESP32"
deviceId =
"143143"
authMethod
= "token"
authToken =
"123456789"

# Initialize GPIO


def myCommandCallback(cmd):
    print("Command received: %s" %
    cmd.data['command'])
    status=cmd.data['command']
```

```
if status=="START":
    print
("Motor is
Started")elif
status=="STO
P":
    print ("Motor
is oFF state")elif
status=="LEFT":
    print ("Left
Side is Closed")
elif
status=="RIGHT
":
    print ("Right
Side is Closed")elif
status=="FORWA
RD":
```

```
        print ("Message is Forward to
    the chief")else :
        print ("Send a proper

    command")#print(cmd)




try:
                                                    devic
                                                    eOpti
ons = {"org": organization, "type": deviceType, "id":

deviceId, "auth-method": authMethod,
"auth-token":authToken}


= ibmiotf.device.Client(deviceOptions)

.....
```

```
        except Exception as e:

        ght exception connecting device: %s" % str(e))

deviceCli #.................



print("Causys.exit()



        # Connect and send a datapoint "hello" with
        value "world"into the cloud as an event of
        type "greeting" 10 times
        deviceCli.connect()

        while True:
                #Get Sensor Data from DHT11
```

```
        Temperature=random.randint(
        0,100)
        Turbidity=random.randint(0,10
        0) pH=random.randint(0,14)


        data = { 'Temperature' : Temperature,
'Turbidity':Turbidity, 'pH' : pH }
        #print data
        def myOnPublishCallback():
            print ("Published Temperature = %s C" %
Temperature, "Turbidity = %s %%" % Turbidity,
"pH = %s L"
% pH, "to IBM Watson")


        success =
deviceCli.publishEvent("IoTSensor", "json",data,
qos=0, on_publish=myOnPublishCallback)
        if not success:
```

```
        print("Not
    connected to IoTF")
    time.sleep(20)

    deviceCli.commandCallback =
    myCommandCallback

# Disconnect the device and application
from the clouddeviceCli.disconnect()
```