Sprint – 4
Hazardous Area Monitoring for Industrial Plant powered by IoT

Program:

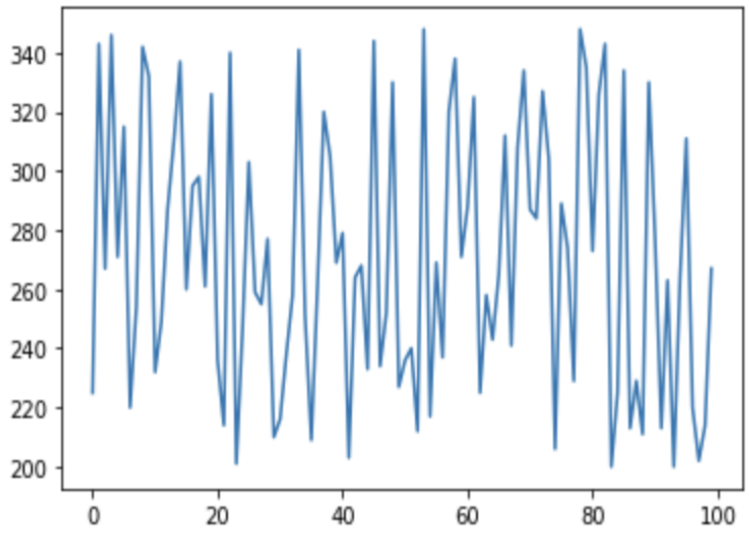
```
#include <WiFi.h>
#<PubSubClient.h>
#define DHTPIN 15
DHTTYPE DHT22
#define LED 2

DHT dht (DHTPIN, DHTTYPE);// creating the instance by passing
pin and typr of dht connected void callback(char*
subscribetopic, byte* payload, unsigned int payloadLength);
//-------credentials of IBM Accounts------

#define ORG "$$$$$$"//IBM ORGANITION ID
#define DEVICE_TYPE "nodeMCU"//Device type mentioned in ibm
watson IOT Platform #define DEVICE_ID "12345"//Device ID
mentioned in ibm watson IOT Platform #define TOKEN "12345678"
//Token
String data3; float h, t;

//-------- Customise the above values -------- char server[] =
ORG ".messaging.internetofthings.ibmcloud.com";// Server Name
char publishTopic[]
= "iot-2/evt/Data/fmt/json";// topic name and type of event
perform and format in which data to be send
char subscribetopic[] = "iot-2/cmd/command/fmt/String";// cmd
REPRESENT command type AND COMMAND IS TEST OF FORMAT STRING
char authMethod[] = "use-token-auth";// authentication method
char token[] = TOKEN; char clientId[] = "d:" ORG ":" DEVICE_TYPE
":" DEVICE_ID;//client id

 //  -
WiFiClient wifiClient; // creating the instance for wificlient
```

```
PubSubClient client(server, 1883, callback ,wifiClient);
//calling the predefined client id by passing parameter like
server id,portand wificredential


void setup()// configureing the ESP32
{
 Serial.begin(115200); dht.begin();
 pinMode(LED,OUTPUT); delay(10);   Serial.println();
wificonnect(); mqttconnect();
} void loop()// Recursive Function
{    h = dht.readHumidity(); t = dht.readTemperature();
 Serial.print("temp:"); Serial.println(t);
 Serial.print("Humid:"); Serial.println(h); PublishData(t, h);
delay(1000);   if (!client.loop()) { mqttconnect();
 }
}




 void PublishData(float temp, float humid)
{
 mqttconnect();//function call for connecting to ibm
 /*       creating the String in in form JSon to update the data
to ibm cloud   */
 String payload = "{\"temp\":"; payload += temp; payload += ","
"\"Humid\":";       payload += humid;   payload += "}";


 Serial.print("Sending payload: ");
 Serial.println(payload);

 if (client.publish(publishTopic, (char*) payload.c_str())) {
  Serial.println("Publish ok");// if it sucessfully upload data
on the cloud then it will print publish ok in Serial monitor or
else it will print publish failed } else
{
  Serial.println("Publish failed");
 }
```

```
}

void mqttconnect() {     if (!client.connected()) {
   Serial.print("Reconnecting client to ");
Serial.println(server);
   while (!!!client.connect(clientId, authMethod, token)) {
Serial.print("."); delay(500);
   }
    initManagedDevice();
    Serial.println();
}
}
 void wificonnect() //function defination for wificonnect
{
  Serial.println(); Serial.print("Connecting to ");

  WiFi.begin("Wokwi-GUEST", "", 6);//passing the wifi credentials
to establish the connection   while (WiFi.status() !=
WL_CONNECTED) {     delay(500);
   Serial.print(".");
  }
  Serial.println(""); Serial.println("WiFi connected");
  Serial.println("IP address: "); Serial.println(WiFi.localIP());
}
void initManagedDevice() {
  if (client.subscribe(subscribetopic)) {
   Serial.println((subscribetopic)); Serial.println("subscribe to
     cmd OK");
  } else {
   Serial.println("subscribe to cmd FAILED");
}
}
void callback(char* subscribetopic, byte* payload, unsigned int
payloadLength)
{

  Serial.print("callback invoked for topic: ");
Serial.println(subscribetopic);
```

```
  for (int i = 0; i < payloadLength; i++) {
//Serial.print((char)payload[i]);
data3+= (char)payload[i];
  }
  Serial.println("data: "+ data3); if(data3=="lighton")
  {
  Serial.println(data3); digitalWrite(LED,HIGH);
  }
  else
  {
Serial.println(data3); digitalWrite(LED,LOW);
} data3="";
}
```

Output Obtained From one of the Environmental Sensors:


340
320
300
280
260
240
220
200
0
20
40
60
80
100