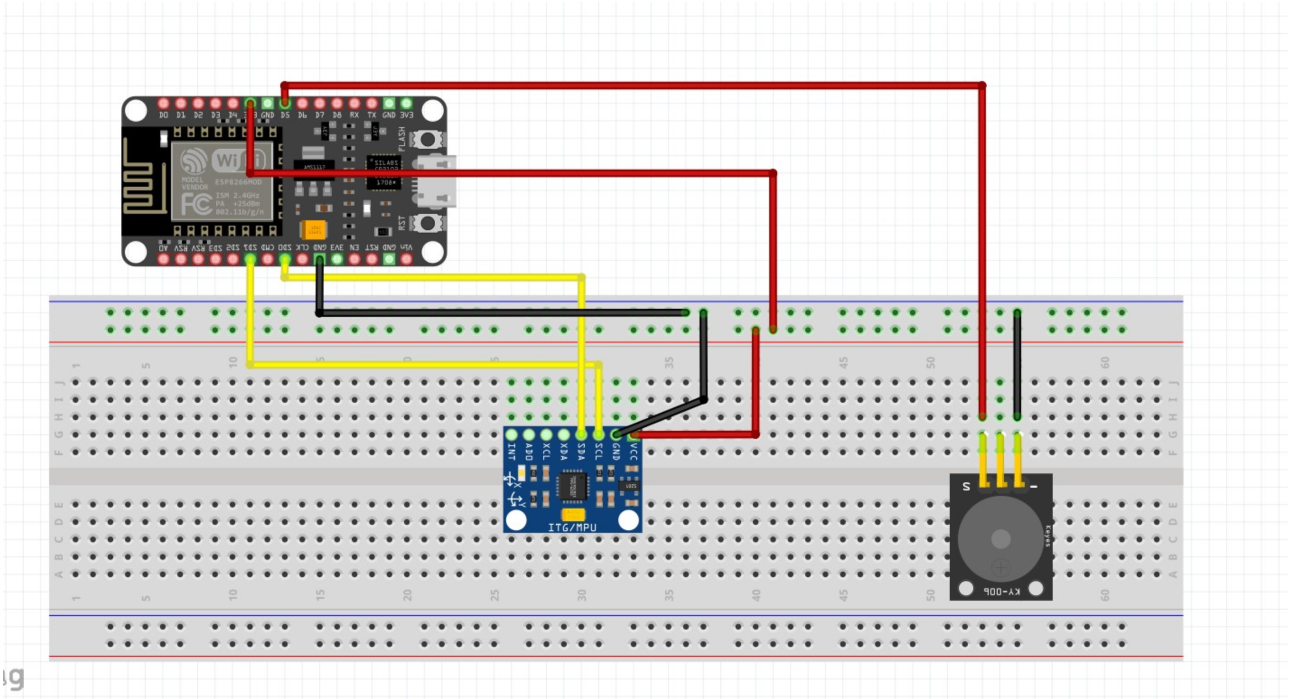
Title : Lab 5 Connecting to WIFI using Arduino based on ESP8266

The main objective of this lab is to design, build, and integrate the IoT nodes designed during the previous labs. To do this, we will learn how to use a simple circuit design software and see how to send and receive IoT commands to/from our Node-RED in the FlowFuse Cloud.

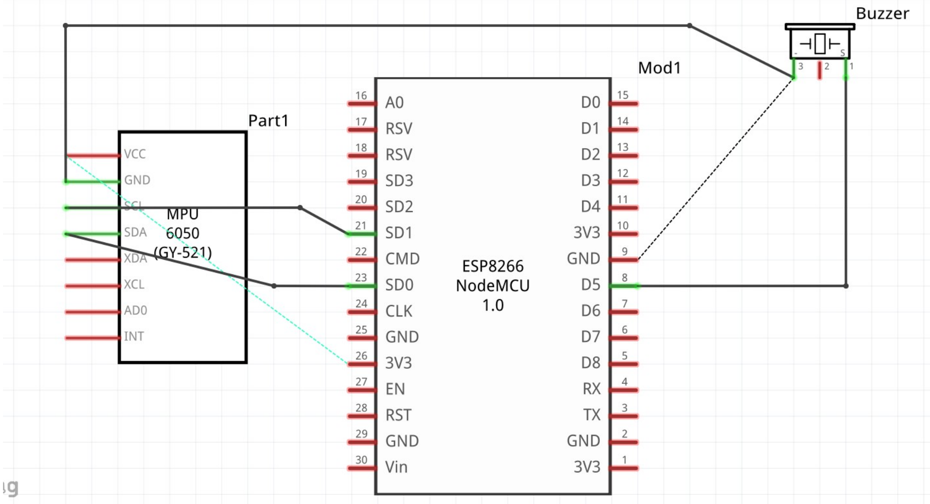
Tasks

1. Model the IoT nodes of your group project (ESP8266 and sensors/actuators from Lab 3) using Fritzing and upload the following to a new section of the GitHub Wiki:

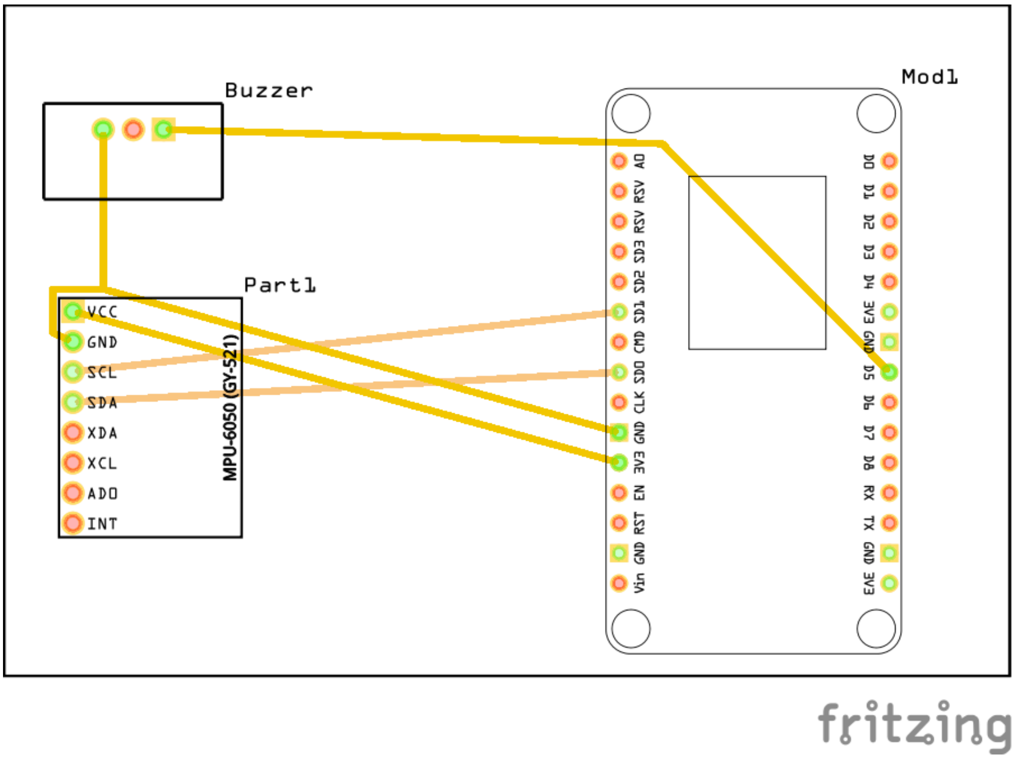
- **Screenshot of the Fritzing design for each IoT node.**


D0 D1 D2 D3 D4 3V3 GND D5 D6 D7 D8 RX TX GND 3V3
A0 RSV RSV SD3 SD2 SD1 CMD SD0 CLK GND 3V3 EN RST GND Vin
FLASH
RST
INT
AD0
XCL
XDA
SDA
SCL
GND
VCC
ITG/MPU
S
KY-006

- **Schematic of each IoT node.**

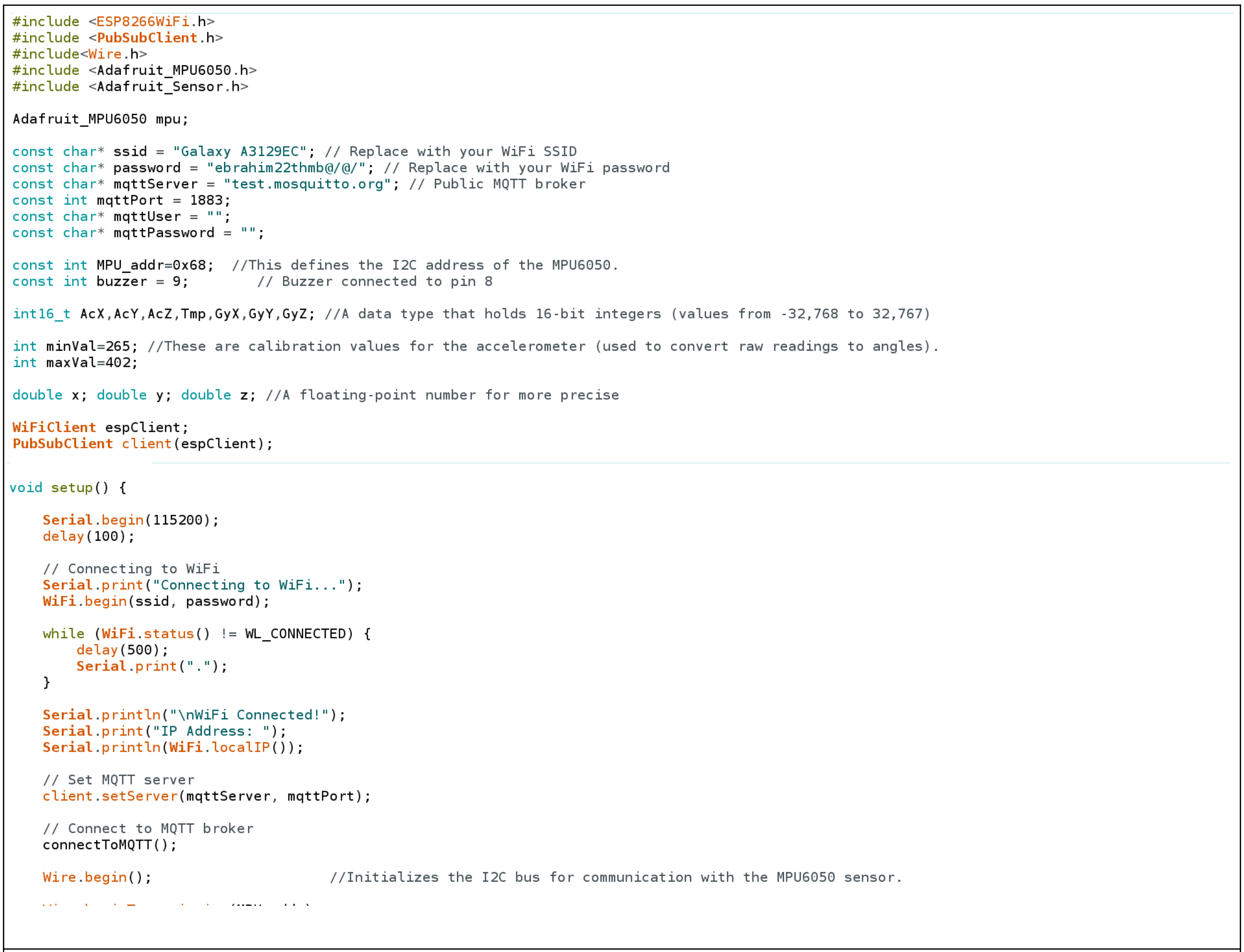


- **PCB of each IoT node.**

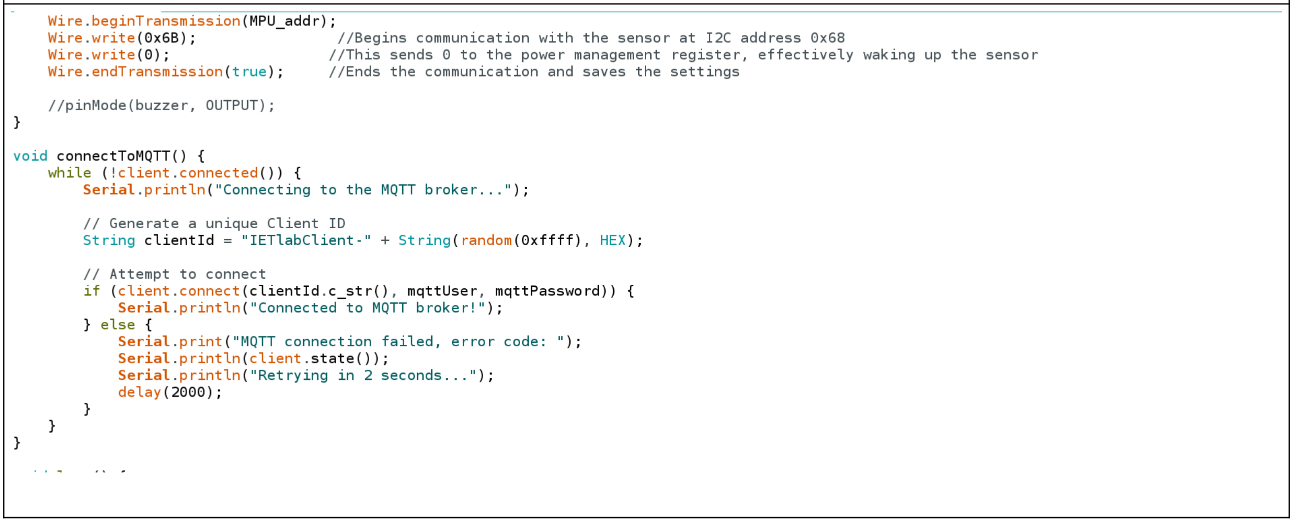


2. Connect each built IoT hardware node to Node-RED in FlowFuse via MQTT, displaying the data using a Debug component.

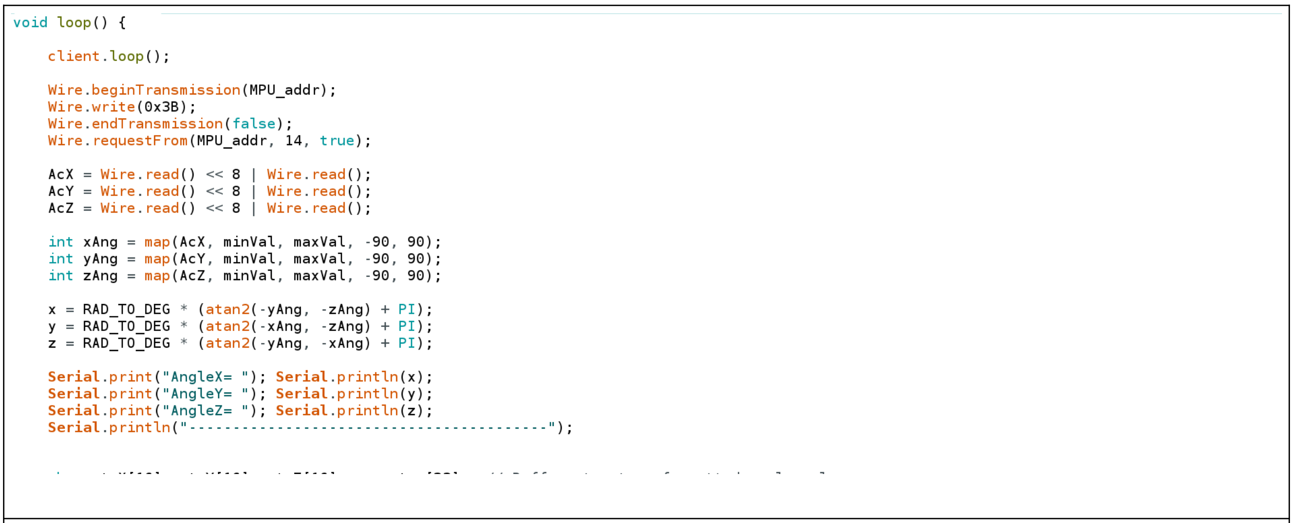
The sensor that is used in this section is Gy-521 which is for recording acceleration, angle, and temperature. In this part, it is tried to send the angle of gyroscope to Node-RED in FlowFuse by using following code in Arduino (esp8266).



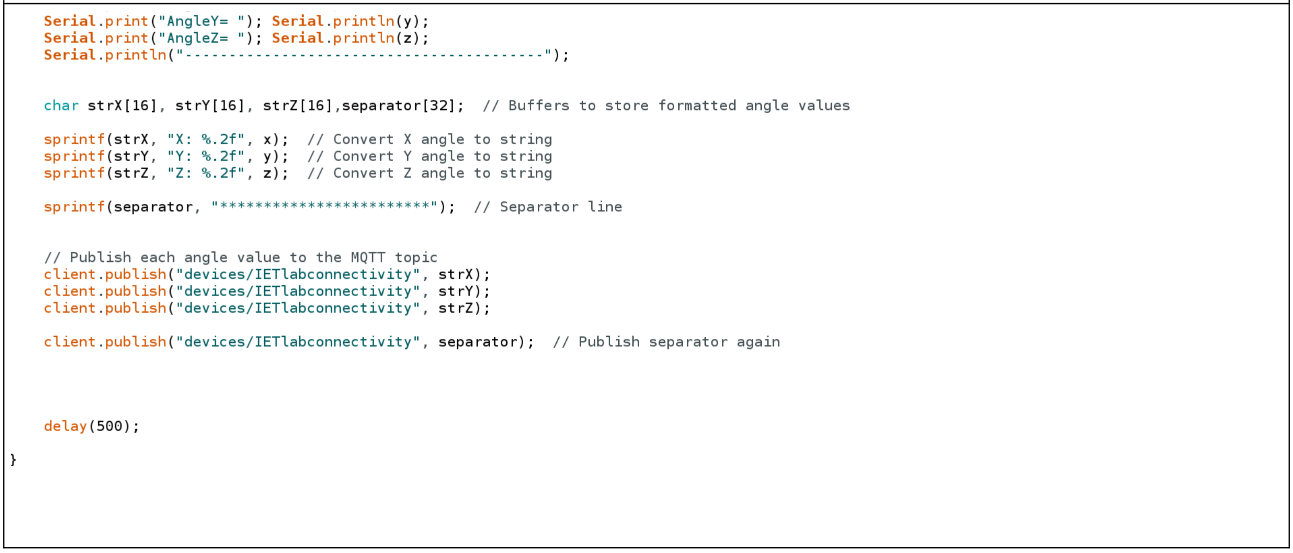
```
#include <ESP8266WiFi.h>
#include <PubSubClient.h>
#include<Wire.h>
#include <Adafruit_MPU6050.h>
#include <Adafruit_Sensor.h>

Adafruit_MPU6050 mpu;

const char* ssid = "Galaxy A3129EC"; // Replace with your WiFi SSID
const char* password = "ebrahim22thmb@/@/"; // Replace with your WiFi password
const char* mqttServer = "test.mosquitto.org"; // Public MQTT broker
const int mqttPort = 1883;
const char* mqttUser = "";
const char* mqttPassword = "";

const int MPU_addr=0x68;  //This defines the I2C address of the MPU6050.
const int buzzer = 9;        // Buzzer connected to pin 8

int16_t AcX,AcY,AcZ,Tmp,GyX,GyY,GyZ; //A data type that holds 16-bit integers (values from -32,768 to 32,767)

int minVal=265; //These are calibration values for the accelerometer (used to convert raw readings to angles).
int maxVal=402;

double x; double y; double z; //A floating-point number for more precise

WiFiClient espClient;
PubSubClient client(espClient);


void setup() {

    Serial.begin(115200);
    delay(100);

    // Connecting to WiFi
    Serial.print("Connecting to WiFi...");
    WiFi.begin(ssid, password);

    while (WiFi.status() != WL_CONNECTED) {
        delay(500);
        Serial.print(".");
    }

    Serial.println("\nWiFi Connected!");
    Serial.print("IP Address: ");
    Serial.println(WiFi.localIP());

    // Set MQTT server
    client.setServer(mqttServer, mqttPort);

    // Connect to MQTT broker
    connectToMQTT();

    Wire.begin();                     //Initializes the I2C bus for communication with the MPU6050 sensor.
```



```
    Wire.beginTransmission(MPU_addr);
    Wire.write(0x6B);                //Begins communication with the sensor at I2C address 0x68
    Wire.write(0);                  //This sends 0 to the power management register, effectively waking up the sensor
    Wire.endTransmission(true);     //Ends the communication and saves the settings

    //pinMode(buzzer, OUTPUT);
}

void connectToMQTT() {
    while (!client.connected()) {
        Serial.println("Connecting to the MQTT broker...");

        // Generate a unique Client ID
        String clientId = "IETlabClient-" + String(random(0xffff), HEX);

        // Attempt to connect
        if (client.connect(clientId.c_str(), mqttUser, mqttPassword)) {
            Serial.println("Connected to MQTT broker!");
        } else {
            Serial.print("MQTT connection failed, error code: ");
            Serial.println(client.state());
            Serial.println("Retrying in 2 seconds...");
            delay(2000);
        }
    }
}
```

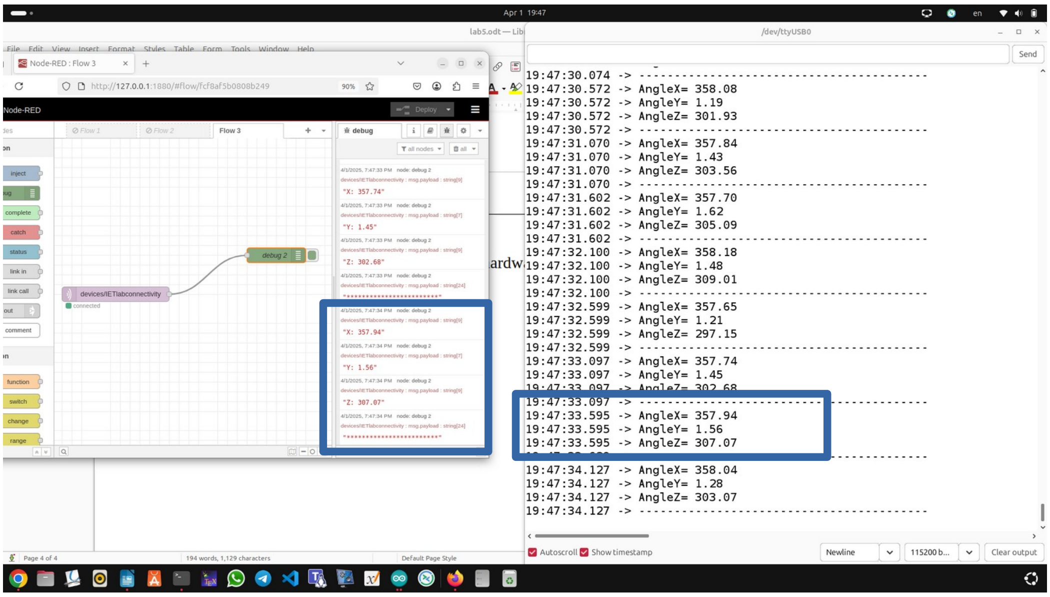


```
void loop() {

    client.loop();

    Wire.beginTransmission(MPU_addr);
    Wire.write(0x3B);
    Wire.endTransmission(false);
    Wire.requestFrom(MPU_addr, 14, true);

    AcX = Wire.read() << 8 | Wire.read();
    AcY = Wire.read() << 8 | Wire.read();
    AcZ = Wire.read() << 8 | Wire.read();

    int xAng = map(AcX, minVal, maxVal, -90, 90);
    int yAng = map(AcY, minVal, maxVal, -90, 90);
    int zAng = map(AcZ, minVal, maxVal, -90, 90);

    x = RAD_TO_DEG * (atan2(-yAng, -zAng) + PI);
    y = RAD_TO_DEG * (atan2(-xAng, -zAng) + PI);
    z = RAD_TO_DEG * (atan2(-yAng, -xAng) + PI);

    Serial.print("AngleX= "); Serial.println(x);
    Serial.print("AngleY= "); Serial.println(y);
    Serial.print("AngleZ= "); Serial.println(z);
    Serial.println("-----------------------------------------");
```



```
    Serial.print("AngleY= "); Serial.println(y);
    Serial.print("AngleZ= "); Serial.println(z);
    Serial.println("-----------------------------------------");


    char strX[16], strY[16], strZ[16],separator[32];  // Buffers to store formatted angle values

    sprintf(strX, "X: %.2f", x);  // Convert X angle to string
    sprintf(strY, "Y: %.2f", y);  // Convert Y angle to string
    sprintf(strZ, "Z: %.2f", z);  // Convert Z angle to string

    sprintf(separator, "************************");  // Separator line


    // Publish each angle value to the MQTT topic
    client.publish("devices/IETlabconnectivity", strX);
    client.publish("devices/IETlabconnectivity", strY);
    client.publish("devices/IETlabconnectivity", strZ);

    client.publish("devices/IETlabconnectivity", separator);  // Publish separator again



    delay(500);

}
```

Develop the necessary code to control the IoT hardware actuators of the group project via the remote Node-RED dashboard.

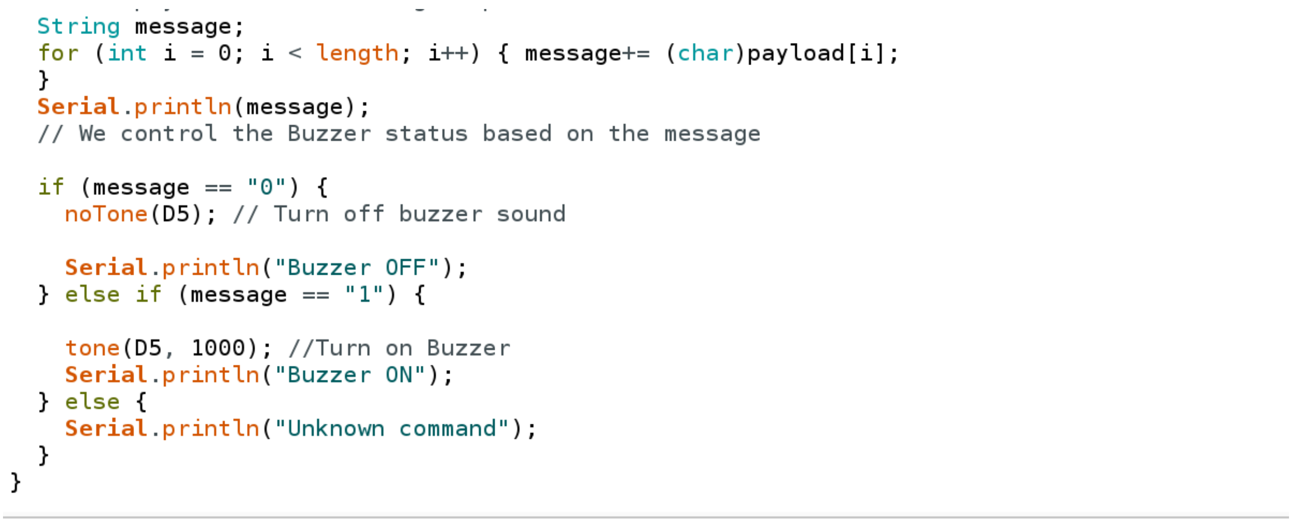


```
4/1/2025, 7:47:33 PM   node: debug 2
devices/IETlabconnectivity : msg.payload : string[9]
"X: 357.74"
4/1/2025, 7:47:33 PM   node: debug 2
devices/IETlabconnectivity : msg.payload : string[7]
"Y: 1.45"
4/1/2025, 7:47:33 PM   node: debug 2
devices/IETlabconnectivity : msg.payload : string[9]
"Z: 302.68"
4/1/2025, 7:47:33 PM   node: debug 2
devices/IETlabconnectivity : msg.payload : string[24]
"************************"
4/1/2025, 7:47:34 PM   node: debug 2
devices/IETlabconnectivity : msg.payload : string[9]
"X: 357.94"
4/1/2025, 7:47:34 PM   node: debug 2
devices/IETlabconnectivity : msg.payload : string[7]
"Y: 1.56"
4/1/2025, 7:47:34 PM   node: debug 2
devices/IETlabconnectivity : msg.payload : string[9]
"Z: 307.07"
4/1/2025, 7:47:34 PM   node: debug 2
devices/IETlabconnectivity : msg.payload : string[24]
"************************"
```

```
19:47:30.074 -> -----------------------------------------
19:47:30.572 -> AngleX= 358.08
19:47:30.572 -> AngleY= 1.19
19:47:30.572 -> AngleZ= 301.93
19:47:30.572 -> -----------------------------------------
19:47:31.070 -> AngleX= 357.84
19:47:31.070 -> AngleY= 1.43
19:47:31.070 -> AngleZ= 303.56
19:47:31.070 -> -----------------------------------------
19:47:31.602 -> AngleX= 357.70
19:47:31.602 -> AngleY= 1.62
19:47:31.602 -> AngleZ= 305.09
19:47:31.602 -> -----------------------------------------
19:47:32.100 -> AngleX= 358.18
19:47:32.100 -> AngleY= 1.48
19:47:32.100 -> AngleZ= 309.01
19:47:32.100 -> -----------------------------------------
19:47:32.599 -> AngleX= 357.65
19:47:32.599 -> AngleY= 1.21
19:47:32.599 -> AngleZ= 297.15
19:47:32.599 -> -----------------------------------------
19:47:33.097 -> AngleX= 357.74
19:47:33.097 -> AngleY= 1.45
19:47:33.097 -> AngleZ= 302.68
19:47:33.097 -> -----------------------------------------
19:47:33.595 -> AngleX= 357.94
19:47:33.595 -> AngleY= 1.56
19:47:33.595 -> AngleZ= 307.07
19:47:34.127 -> AngleX= 358.04
19:47:34.127 -> AngleY= 1.28
19:47:34.127 -> AngleZ= 303.07
19:47:34.127 -> -----------------------------------------
```

4. Develop the necessary code to control the IoT hardware actuators of the group project via the remote Node-RED dashboard.

For this part, it is tried to just change the parameter of the example case of tutorial file (turn on/off by using switch in Node-Red in Flow Fuse. It is shown in the following picture.



```
  String message;
  for (int i = 0; i < length; i++) { message+= (char)payload[i];
  }
  Serial.println(message);
  // We control the Buzzer status based on the message

  if (message == "0") {
    noTone(D5); // Turn off buzzer sound

    Serial.println("Buzzer OFF");
  } else if (message == "1") {

    tone(D5, 1000); //Turn on Buzzer
    Serial.println("Buzzer ON");
  } else {
    Serial.println("Unknown command");
  }
}
```