20BCE10471 | VIT Bhopal Campus

# Suresh Mahalingam Konar

## Assignment : 3

In wokwi add LED and switch on and off from node-red

## Implementation

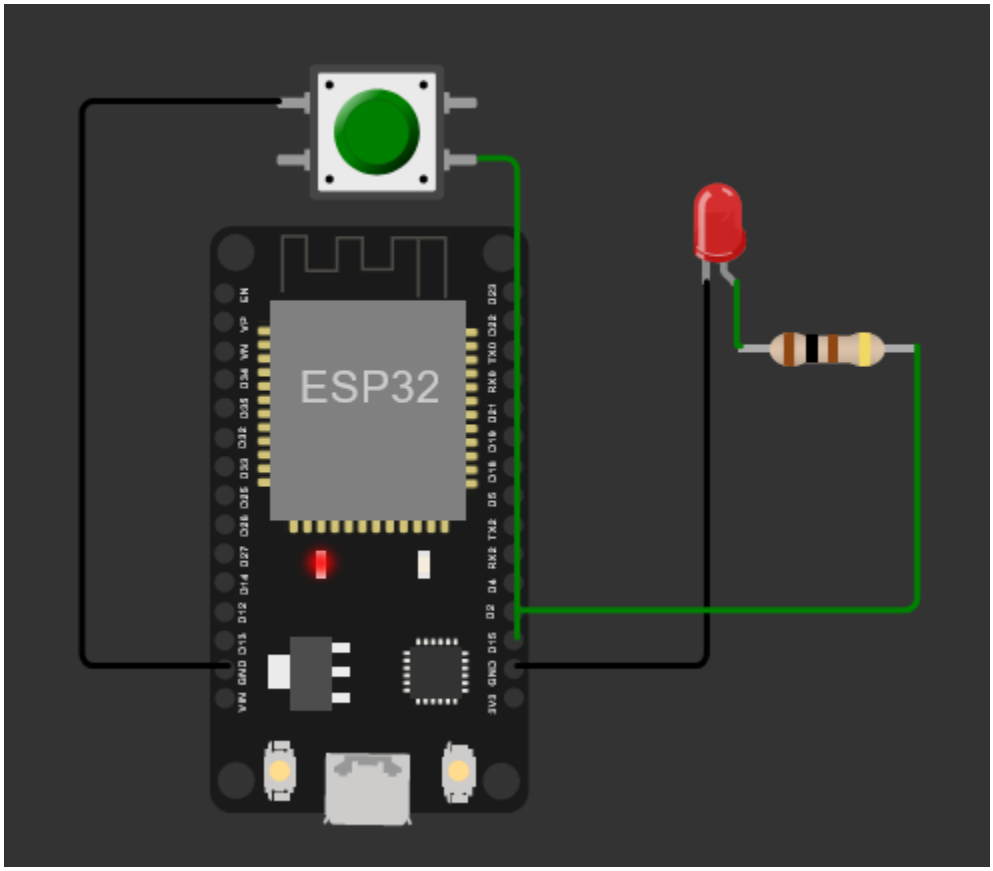
Code :

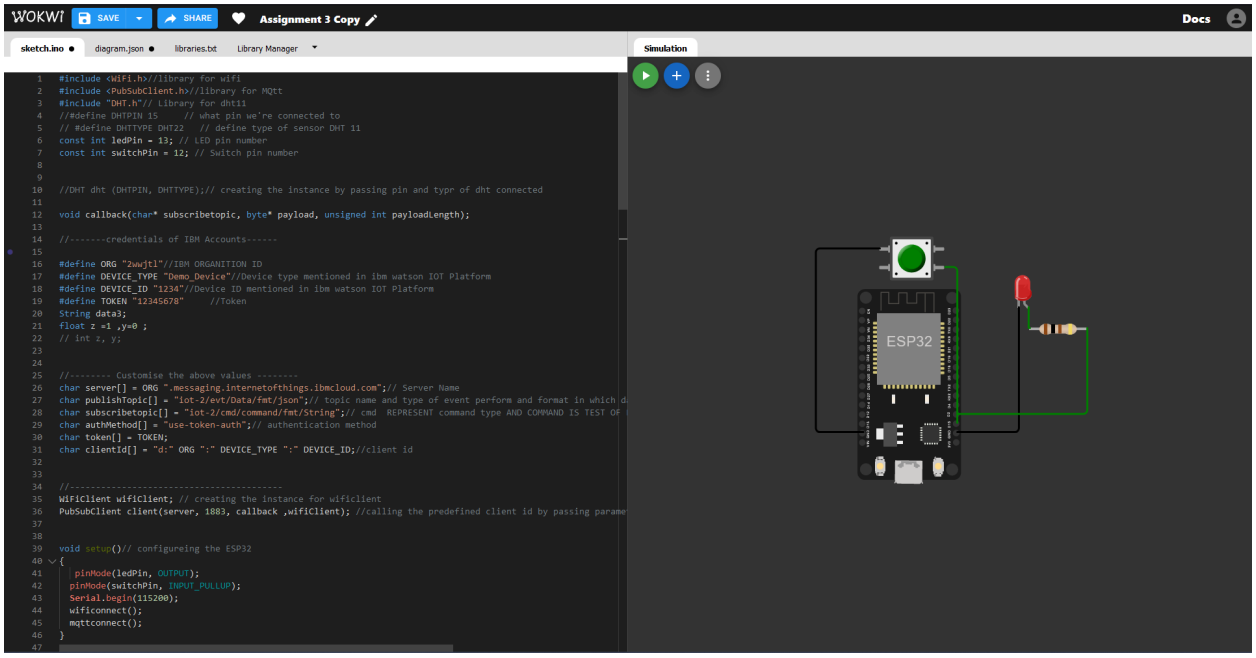
```
#include <WiFi.h>//library for wifi
#include <PubSubClient.h>//library for MQtt
#include "DHT.h"// Library for dht11
//#define DHTPIN 15     // what pin we're connected to
// #define DHTTYPE DHT22   // define type of sensor DHT 11
const int ledPin = 13; // LED pin number
const int switchPin = 12; // Switch pin number


//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 "2wwjtl"//IBM ORGANITION ID
#define DEVICE_TYPE "Demo_Device"//Device type mentioned in ibm watson IOT Platform
#define DEVICE_ID "1234"//Device ID mentioned in ibm watson IOT Platform
#define TOKEN "12345678"     //Token
String data3;
float z =1 ,y=0 ;
// int z, y;


//-------- 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,por


void setup()// configureing the ESP32
{
  pinMode(ledPin, OUTPUT);
  pinMode(switchPin, INPUT_PULLUP);
  Serial.begin(115200);
  wificonnect();
  mqttconnect();
}
```

```
void loop()// Recursive Function
{
if (Serial.available() > 0) {
    char command = Serial.read();
    if (command == '1') {
      digitalWrite(ledPin, HIGH); // Turn the LED on
    } else if (command == '0') {
      digitalWrite(ledPin, LOW); // Turn the LED off
    }
  }

  if (!client.loop()) {
    mqttconnect();
  }
}



/*.....................................retrieving to Cloud.............................*/

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 = "{\"ON\":";
  payload += z;
  payload += "," "\"OFF\":";
  payload += y;
  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
  } else {
    Serial.println("Publish failed");
  }

}

```

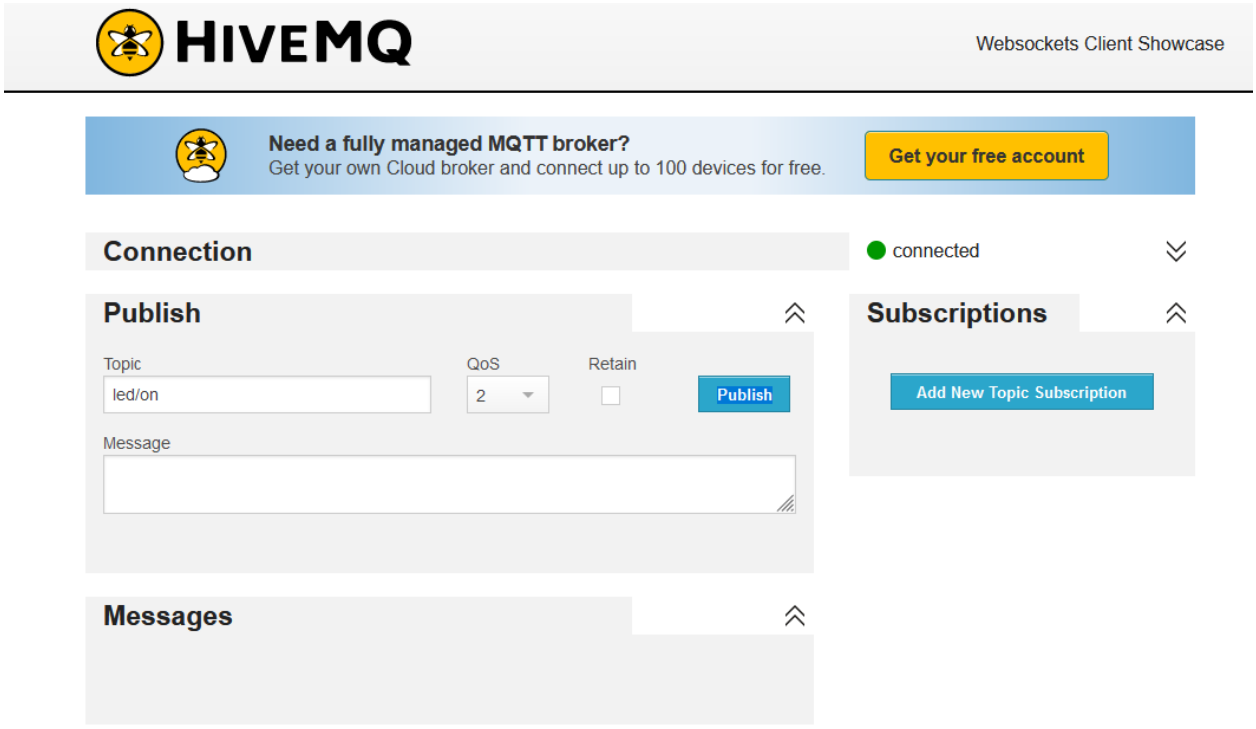
```
  Serial.println();
  Serial.print("Connecting to ");

  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="";
}
```

Simulation:

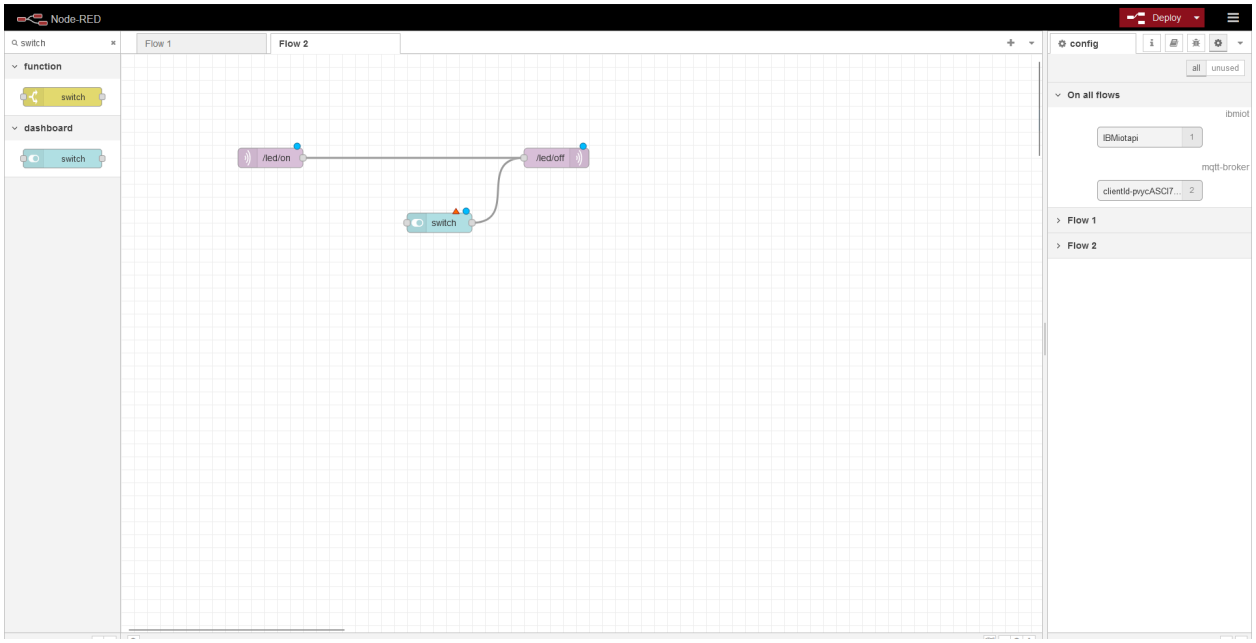





HiveMQ MQTT Server :




Node-red :




# Wokwi Link

Assignemnt Open link : [Link]