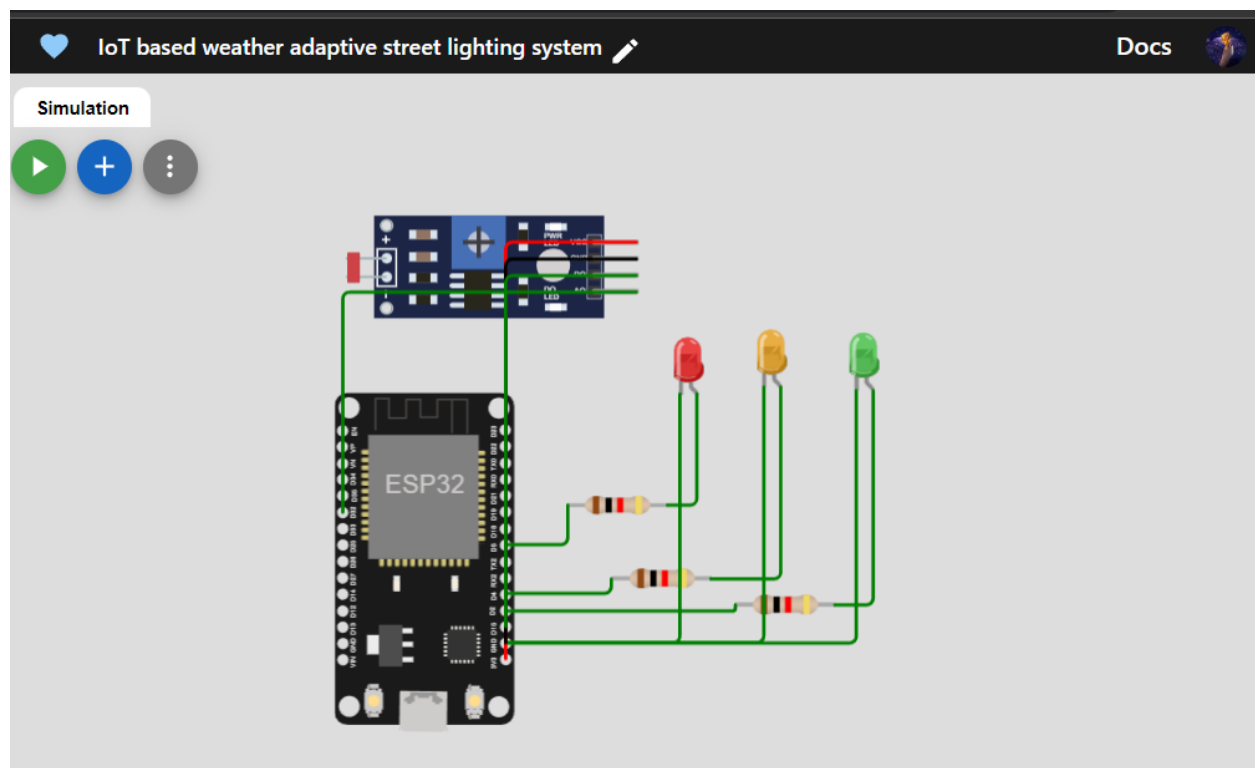


<b>TEAM ID</b>	NM2023TMID15296
<b>PROJECT NAME</b>	IoT BASED WEATHER ADAPTIVE STREET LIGHTING SYSTEM

**DEVELOP A DEVICE CODE:**

**IoT BASED WEATHER ADAPTIVE STREET LIGHTING SYSTEM**

**WOKWI CONNECTIONS:**



**WOKWI CODE:**

```
#include <WiFi.h> //library for wifi
#include <PubSubClient.h> //library for MQTT
#define LED 5
#define LED2 4
#define LED3 2
int LDR = 32;
int LDRReading = 0;
int threshold_val = 800;
```

```

int IEDBrightness = 0;
int flag=0;
void callback(char* subscribtopic, byte* payload, unsigned int payloadLength);
//-----credentials of IBM Accounts-----
#define ORG "mjktqb"//IBM ORGANITION ID
#define DEVICE_TYPE "streetlight"//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 subscribtopic[] = "iot-2/cmd/test/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()// configuring the ESP32
{
  Serial.begin(115200);
  pinMode(LED,OUTPUT);
  pinMode(LED2,OUTPUT);
  pinMode(LED3,OUTPUT);
  delay(10);
  Serial.println();
  wificonnect();
  mqttconnect();
}
void loop()// Recursive Function
{
  //PublishData(t, h);
  //delay(1000);
  /* LDRReading = analogRead(LDR);
  Serial.print("LDR READING:");

```

```

Serial.println(LDRReading);
if (LDRReading > threshold_val){
  IEDBrightness = map(LDRReading, 0, 1023, 0, 255);
  Serial.print("LED BRIGHTNESS:");
  Serial.println(IEDBrightness);
  analogWrite(LED, IEDBrightness);
  analogWrite(LED2, IEDBrightness);
  analogWrite(LED3, IEDBrightness);
}
else{
  analogWrite(LED, 0);
  analogWrite(LED2, 0);
  analogWrite(LED3, 0);
}
delay(300);*/
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 = "{\"temperature\":";
payload += temp;
payload += ", \"humidity\":";
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=="lighton1")  
{  
Serial.println(data3);  
digitalWrite(LED,HIGH);  
}  
else if(data3=="lightoff1")  
{  
Serial.println(data3);  
digitalWrite(LED,LOW);  
}  
else if(data3=="lighton2")  
{  
Serial.println(data3);  
digitalWrite(LED2,HIGH);  
}  
else if(data3=="lightoff2")  
{  
Serial.println(data3);  
digitalWrite(LED2,LOW);  
}  
else if(data3=="lighton3")  
{  
Serial.println(data3);  
digitalWrite(LED3,HIGH);  
}  
else if(data3=="lightoff3")  
{  
Serial.println(data3);  
digitalWrite(LED3,LOW);  
}  
data3="";  
}
```

## OUTPUT:

The image displays the Wokwi IoT simulator interface. The top bar shows the project name "IoT based weather adaptive street lighting system" and a "Docs" link. The left sidebar contains tabs for "sketch.ino", "diagram.json", "libraries.txt", and "Library Manager". The "sketch.ino" tab is active, showing the following code:

```
1 #include <WiFi.h> //library for wifi
2 #include <PubSubClient.h> //library for MQTT
3 #define LED 5
4 #define LED2 4
5 #define LED3 2
6 int LDR = 32;
7 int LDRReading = 0;
8 int threshold_val = 800;
9 int LEDBrightness = 0;
10 int flag=0;
11
12 void callback(char* subscribetopic, byte* payload, unsigned int length) {
13
14 //-----credentials of IBM Accounts-----
15
16 #define ORG "mjktqb" //IBM ORGANITION ID
17 #define DEVICE_TYPE "streetlight" //Device type mentioned in IBM Cloud
18 #define DEVICE_ID "12345" //Device ID mentioned in IBM Cloud
19 #define TOKEN "12345678" //Token
20 String data3;
21 float h, t;
22
23 //----- Customise the above values -----
24
25 char server[] = ORG ".messaging.internetofthings.ibmcloud.com";
26 char publishTopic[] = "iot-2/evt/Data/fmt/json"; // to publish data
27 char subscribetopic[] = "iot-2/cmd/test/fmt/String"; // to receive data
28
```

The right sidebar shows the "Simulation" tab. It features a visual representation of the hardware: an ESP32 microcontroller, an LDR sensor, and three LEDs (red, yellow, and green). The simulation status bar at the top right indicates a runtime of 00:04.628 and 7% battery. The bottom status bar shows the following output:

```
WiFi connected
IP address:
10.10.0.2
Reconnecting client to mjktqb.messaging.internetofthings.ibmcloud.com
iot-2/cmd/test/fmt/String
subscribe to cmd OK
```

**WOKWI LINK:** <https://wokwi.com/projects/364901036326401025>