Weather Adaptive Street Light Monitoring System

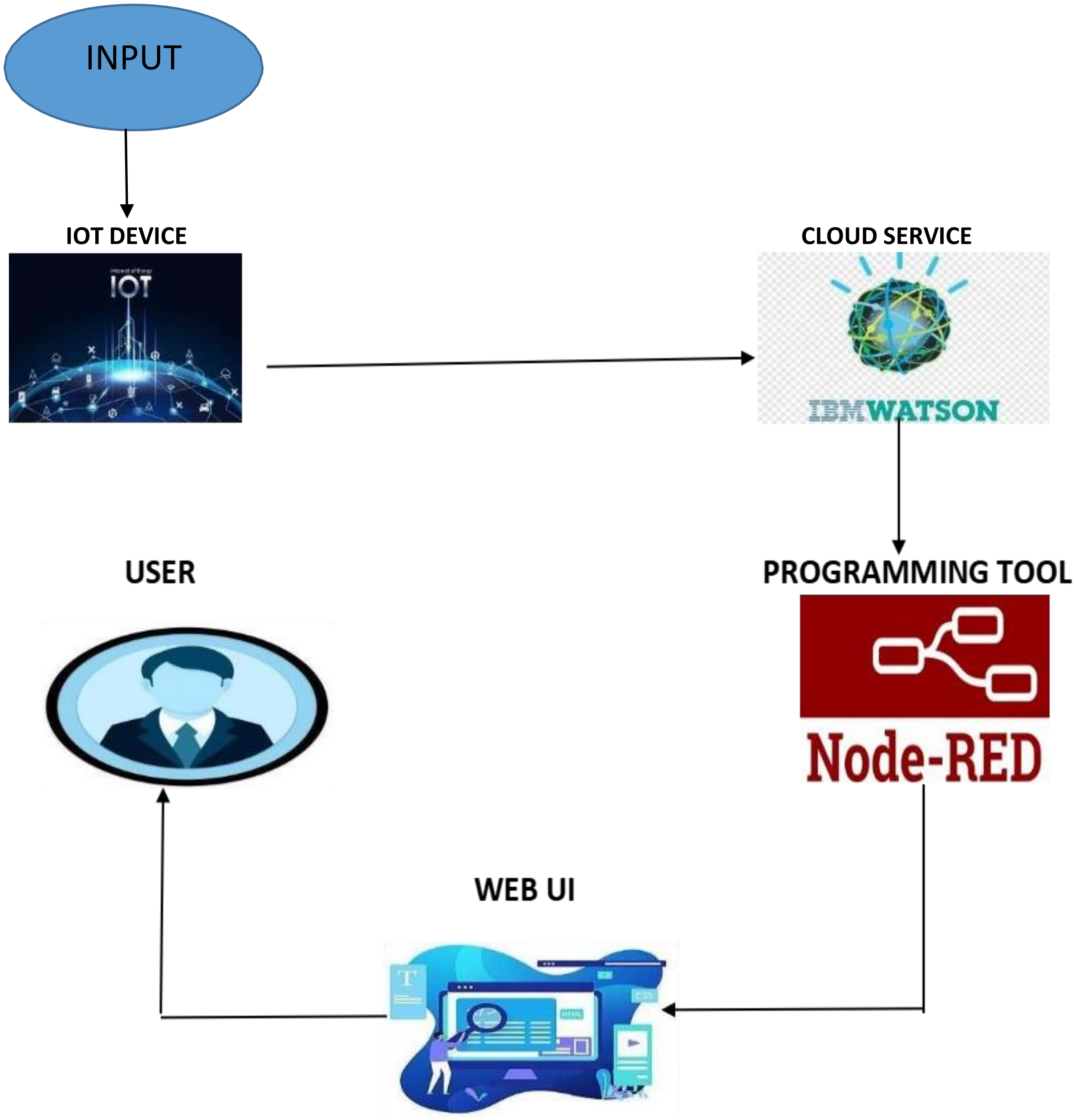
Description:

Lighting the streets account for a major part of the net electric power consumed by many countries. However, the electrical energy consumed by street lights is not efficiently used because the need of street lamps is not essential in every street and every time. This system switches off the light for the parts of the streets which are not in use and turns on the light for the parts of streets which are mostly used when it is dark. The system is connected to a mobile application over the cloud and the users are allowed to manually control the lighting of the street through it.

Components & Technologies:

S.No	Component	Description	Technology
1.	ESP32	powerful dual-core processor, built-in Wi-Fi and Bluetooth connectivity, ample memory, and a rich set of peripheral interfaces	Secure boot and flash encryption, programming interface and seamless wi-fi and bluetooth
2.	Sensor	Collect data from the environments such as light intensity	Photo resistor
3.	LED	Glow led based on the light intensity	LED
4.	Cloud service	computing resources, including storage, processing power, and software applications, over the internet.	IBM Watson IOT service
5.	Node Red	Visual programming tool designed for wiring together hardware devices, APIs, and online services	Node.js ,Java Script ,JSON

PROJECT DESIGN:



Source code:

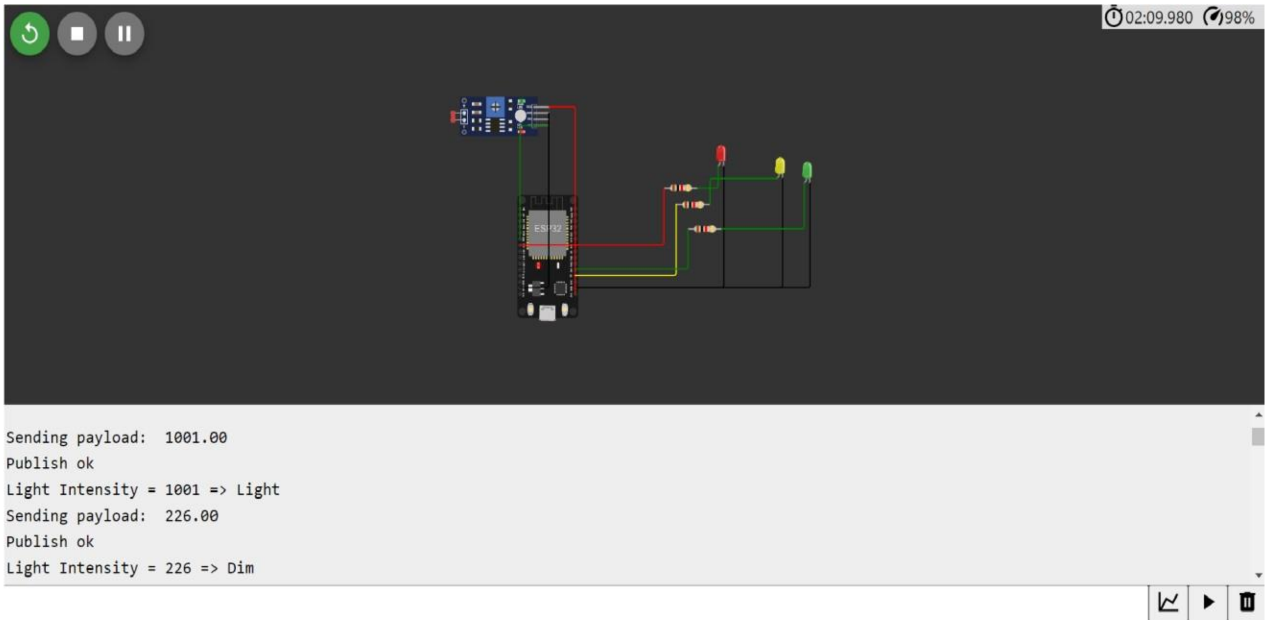
```
#include <WiFi.h> //library for wifi
#include <WiFiClient.h>
#include <PubSubClient.h> //library for MQtt
int LED1= 33;
int LED2 =2;
int LED3= 4;
int LDR = 32;
int LDRReading = 0;
int threshold_val = 800;
int flag=0;
void callback(char* subscribetopic, byte* payload, unsigned int payloadLength);

//-------credentials of IBM Accounts------
#define ORG "qeseew" //IBM ORGANITION ID
#define DEVICE_TYPE "Street-Light" //Device type mentioned in ibm watson
IOT Platform
#define DEVICE_ID "143143" //Device ID mentioned in ibm watson IOT
Platform
#define TOKEN "12345678" //Token
String data3;
float Light_Intensity ;

//-------- Customise the 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
char subscribetopic[] = "iot-2/cmd/test/fmt/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 wifi client
PubSubClient client(server, 1883, callback ,wifiClient);

void setup() // configuring the ESP32
```

```
{
Serial.begin(115200);
pinMode(LED1,OUTPUT);
pinMode(LED2,OUTPUT);
pinMode(LED3,OUTPUT);
delay(10);
Serial.println();
wificonnect();
mqttconnect();
}

void loop()
{
int Light_Intensity = analogRead(LDR);
PublishData(Light_Intensity);
delay(1000);
Serial.print("Light Intensity = ");
Serial.print(Light_Intensity);
if (Light_Intensity < 40) {
Serial.println(" => Dark");
} else if (Light_Intensity < 800) {
Serial.println(" => Dim");
} else if (Light_Intensity < 2000) {
Serial.println(" => Light");
} else if (Light_Intensity < 3200) {
Serial.println(" => Bright");
} else {
Serial.println(" => Very bright");
}
delay(500);
if (!client.loop()) {
mqttconnect();
}
}

/*....................................retrieving to Cloud..............................*/
void PublishData(float Light_Intensity ) {
mqttconnect();//function call for connecting to ibm*/
String payload = " ";
payload += Light_Intensity;
Serial.print("Sending payload: ");
Serial.println(payload);
if (client.publish(publishTopic, (char*) payload.c_str())) {
Serial.println("Publish ok"); // if it successfully 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 definition for wificonnect
{
Serial.println();
Serial.print("Connecting to ");
WiFi.begin("Wokwi-GUEST", "", 6);
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++) {
  data3 += (char)payload[i];
}
  Serial.println("data: "+ data3);
  If(data3=="lighton1")
 {
  Serial.println(data3);
  digitalWrite(LED1,HIGH);
}
else if(data3=="lightoff1")
{
Serial.println(data3);
digitalWrite(LED1,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="";
}
```

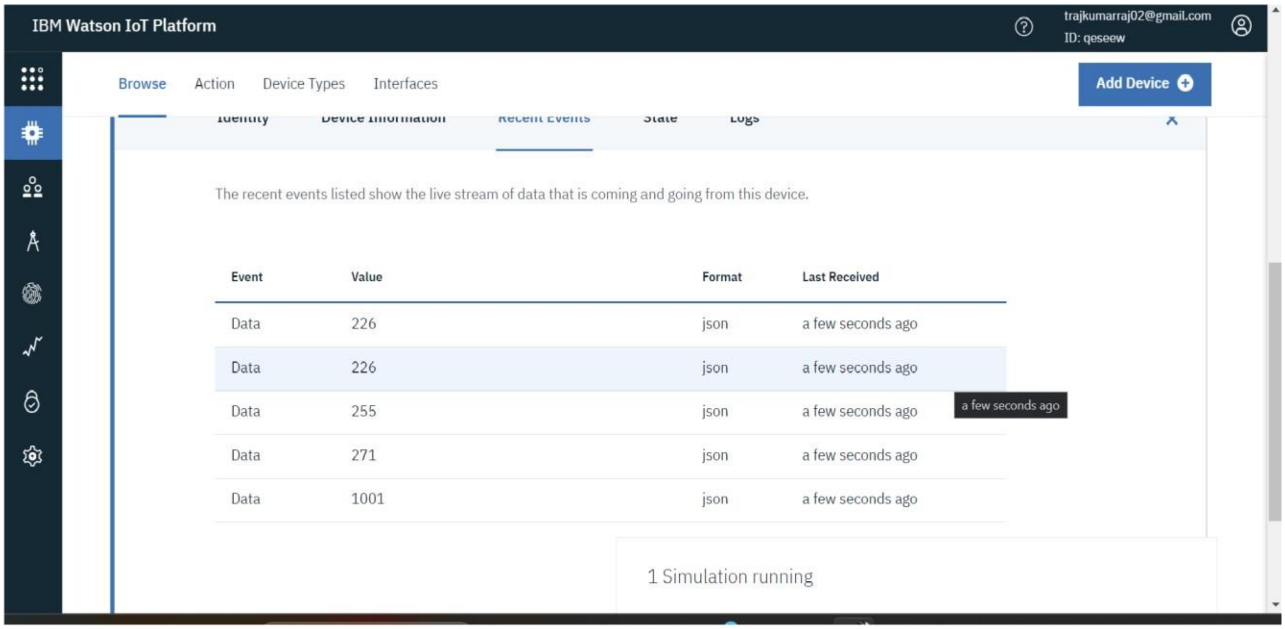
Output:

(I) WOKWI SIMULATION (WEATHER CONDITION:DIM):



```
Sending payload:  1001.00
Publish ok
Light Intensity = 1001 => Light
Sending payload:  226.00
Publish ok
Light Intensity = 226 => Dim
```

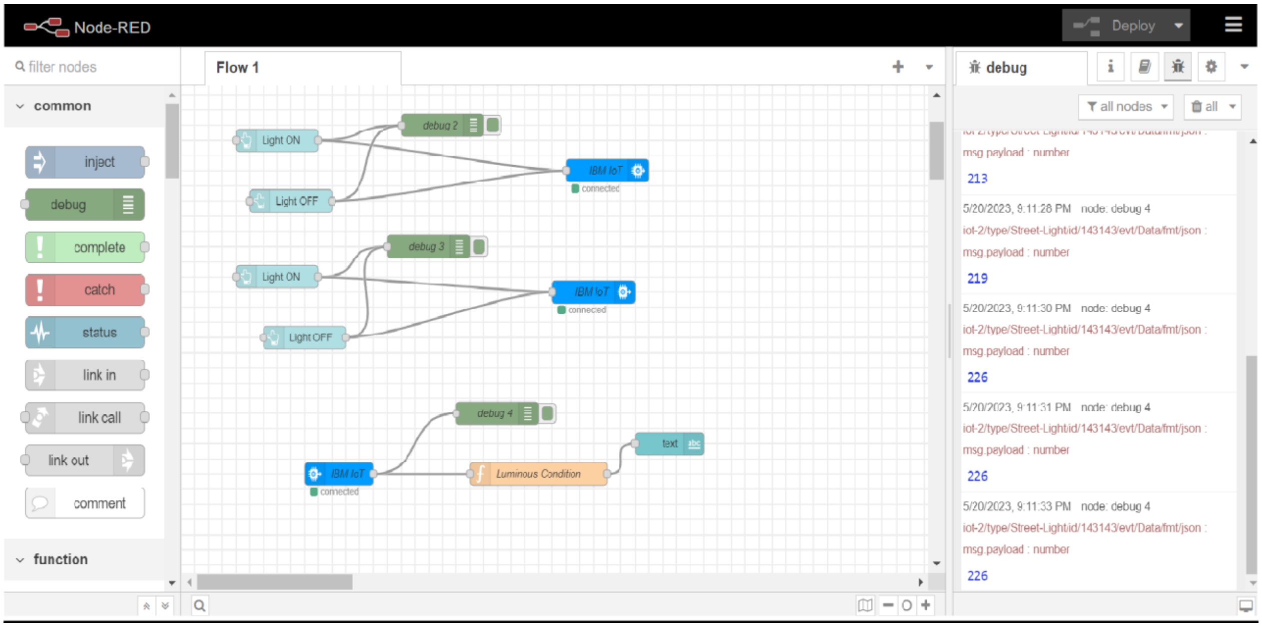
(II) IBM IOT WATSON PLATFORM EVENTS(VALUE FROM WOKWI):



The recent events listed show the live stream of data that is coming and going from this device.

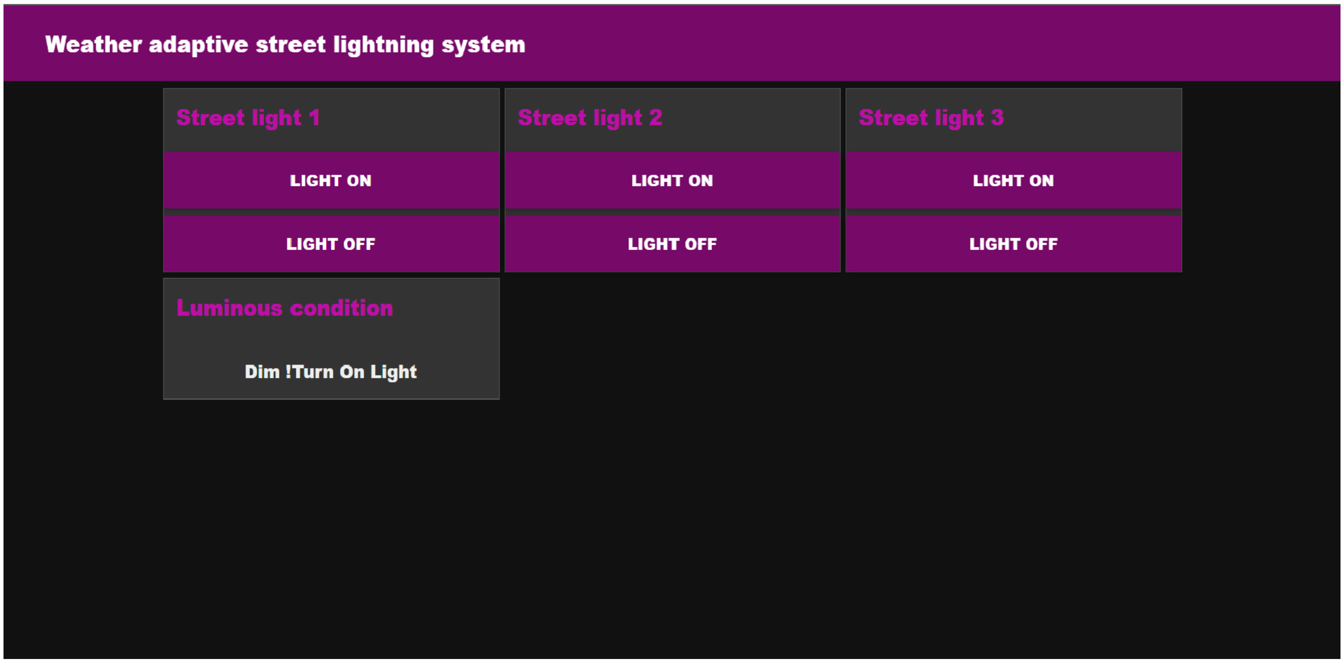
Event	Value	Format	Last Received
Data	226	json	a few seconds ago
Data	226	json	a few seconds ago
Data	255	json	a few seconds ago
Data	271	json	a few seconds ago
Data	1001	json	a few seconds ago

(III) NODE-RED (VALUE FROM IBM IOT WATSON):



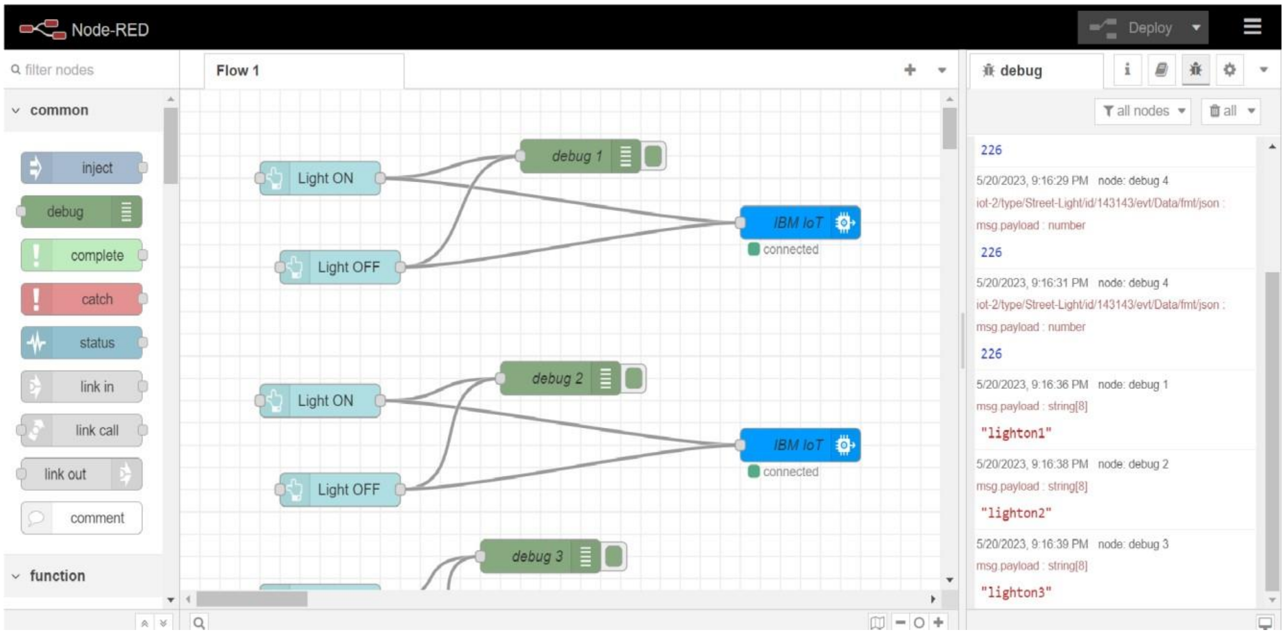
```
msg.payload : number
 213
5/20/2023, 9:11:28 PM   node: debug 4
iot-2/type/Street-Light/id/143143/evt/Data/fmt/json :
msg.payload : number
 219
5/20/2023, 9:11:30 PM   node: debug 4
iot-2/type/Street-Light/id/143143/evt/Data/fmt/json :
msg.payload : number
 226
5/20/2023, 9:11:31 PM   node: debug 4
iot-2/type/Street-Light/id/143143/evt/Data/fmt/json :
msg.payload : number
 226
5/20/2023, 9:11:33 PM   node: debug 4
iot-2/type/Street-Light/id/143143/evt/Data/fmt/json :
msg.payload : number
 226
```

(IV) WEB USER INTERFACE (TURN ON LIGHT):



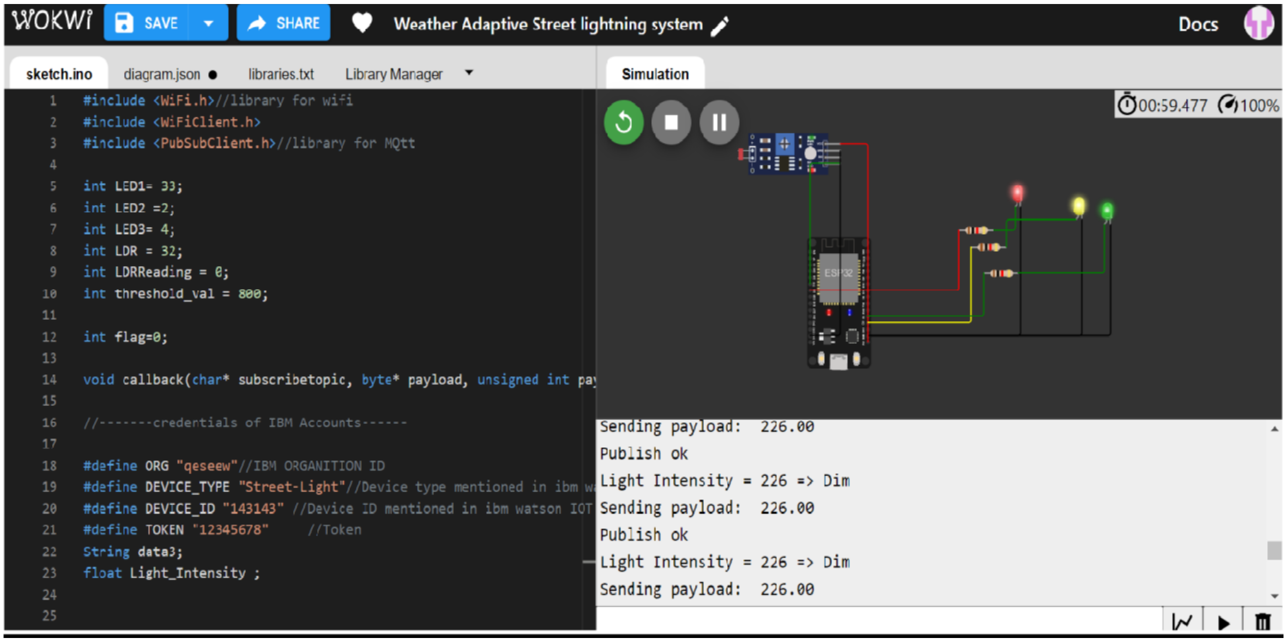
Dim !Turn On Light

(V) NODE-RED (MESSAGE FROM USER):



```
226
5/20/2023, 9:16:29 PM   node: debug 4
iot-2/type/Street-Light/id/143143/evt/Data/fmt/json :
msg.payload : number
226
5/20/2023, 9:16:31 PM   node: debug 4
iot-2/type/Street-Light/id/143143/evt/Data/fmt/json :
msg.payload : number
226
5/20/2023, 9:16:36 PM   node: debug 1
msg.payload : string[8]
"lighton1"
5/20/2023, 9:16:38 PM   node: debug 2
msg.payload : string[8]
"lighton2"
5/20/2023, 9:16:39 PM   node: debug 3
msg.payload : string[8]
"lighton3"
```

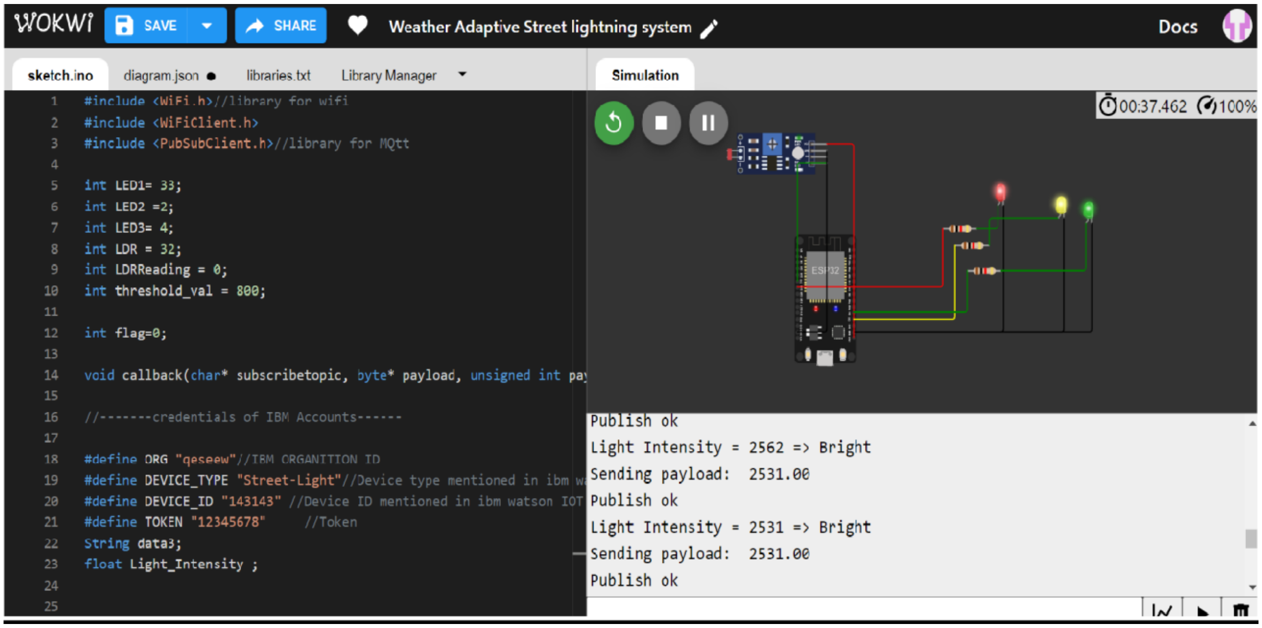
(VI) WOKWI SIMULATION (LIGHT ON):



```
#include <WiFi.h>//library for wifi
#include <WiFiClient.h>
#include <PubSubClient.h>//library for MQtt

int LED1= 33;
int LED2 =2;
int LED3= 4;
int LDR = 32;
int LDRReading = 0;
int threshold_val = 800;

int flag=0;

void callback(char* subscribetopic, byte* payload, unsigned int pay

//-------credentials of IBM Accounts------

#define ORG "qeseew"//IBM ORGANITION ID
#define DEVICE_TYPE "Street-Light"//Device type mentioned in ibm wa
#define DEVICE_ID "143143" //Device ID mentioned in ibm watson IOT
#define TOKEN "12345678"     //Token
String data3;
float Light_Intensity ;
```

```
Sending payload:  226.00
Publish ok
Light Intensity = 226 => Dim
Sending payload:  226.00
Publish ok
Light Intensity = 226 => Dim
Sending payload:  226.00
```

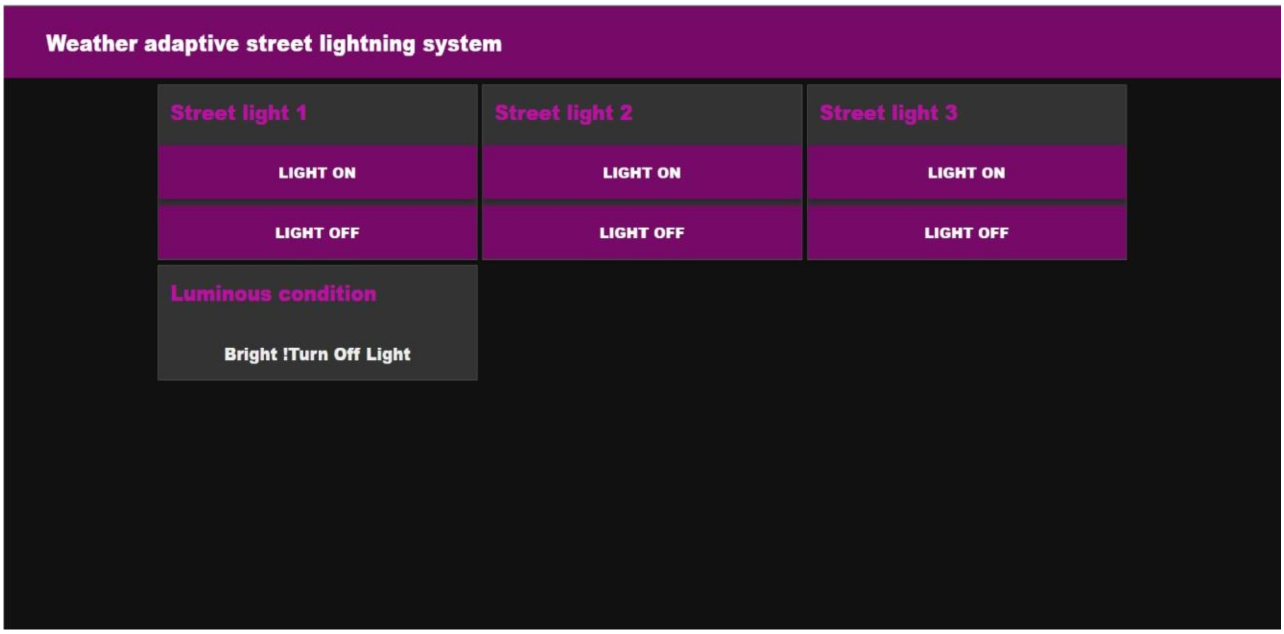
(VII) WOKWI SIMULATION (WEATHER CONDITION :BRIGHT):



```
#include <WiFi.h>//library for wifi
#include <WiFiClient.h>
#include <PubSubClient.h>//library for MQtt

int LED1= 33;
int LED2 =2;
int LED3= 4;
int LDR = 32;
int LDRReading = 0;
int threshold_val = 800;

int flag=0;

void callback(char* subscribetopic, byte* payload, unsigned int pay

//-------credentials of IBM Accounts------

#define ORG "qeseew"//IBM ORGANITION ID
#define DEVICE_TYPE "Street-Light"//Device type mentioned in ibm wa
#define DEVICE_ID "143143" //Device ID mentioned in ibm watson IOT
#define TOKEN "12345678"     //Token
String data3;
float Light_Intensity ;
```

```
Publish ok
Light Intensity = 2562 => Bright
Sending payload:  2531.00
Publish ok
Light Intensity = 2531 => Bright
Sending payload:  2531.00
Publish ok
```

(VIII) USER INTERFACE (TURN OFF LIGHT ALERT):



Weather adaptive street lightning system

Street light 1

LIGHT ON

LIGHT OFF

Street light 2

LIGHT ON

LIGHT OFF

Street light 3

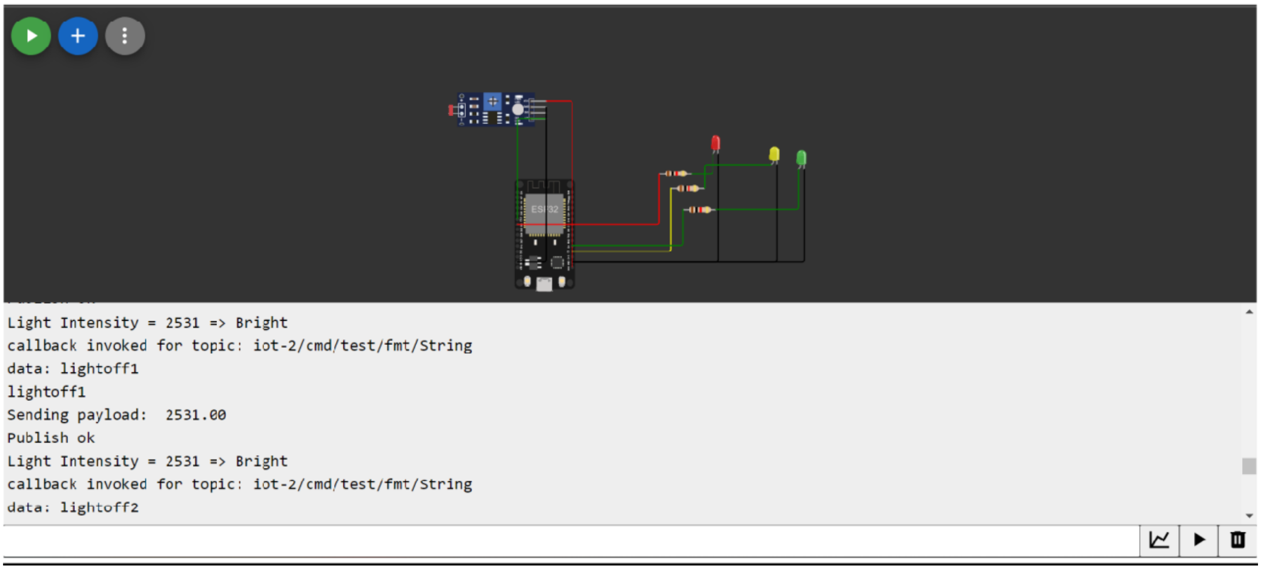
LIGHT ON

LIGHT OFF

Luminous condition

Bright !Turn Off Light

(IX) WOKWI SIMULATION (LIGHT OFF):



```
Light Intensity = 2531 => Bright
callback invoked for topic: iot-2/cmd/test/fmt/String
data: lightoff1
lightoff1
Sending payload:  2531.00
Publish ok
Light Intensity = 2531 => Bright
callback invoked for topic: iot-2/cmd/test/fmt/String
data: lightoff2
```

THANK YOU