```
/*
  Your client is smart things solution provider. He wants to have LED
light in the room to be of same intensity in day and night
automatically. In case of problem, there should be manual control as
well. If there is big change in luminosity, it should be logged to
remote server. Days with big luminosity changes should be studied to
find out pattern.
 */
#include <SPI.h>
#include <Ethernet.h>
#include <Dhcp.h>
#include <Dns.h>
#include <EthernetClient.h>
#include <Temboo.h>

// These constants won't change:
const int sensorPin = 0;    // pin that the sensor is attached to
const int ledPin = 9;        // pin that the LED is attached to

// variables:
int sensorValue = 0;            // the sensor value
int sensorConstrainValue = 0;  // the sensor value with Constrain
int sensorMin = 1023;        // minimum sensor value
int sensorMax = 0;            // maximum sensor value

String HTTP_req;              // stores the HTTP request
int LedStatus = 0;            // state of LED, 0->off    1->on     2->manual
int ManualValue = 0;            // manual  value

// MAC address from Ethernet shield sticker under board
byte mac[] = { 0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED };
IPAddress ip(10, 0, 0, 20); // IP address, may need to change depending on network
EthernetServer server(80);  // create a server at port 80

/*************temboo ********************/
#define TEMBOO_ACCOUNT "jayjayswal"  // Your Temboo account name
#define TEMBOO_APP_KEY_NAME "myFirstApp"  // Your Temboo app key name
#define TEMBOO_APP_KEY "7f299e882e9942e38b2a645b29ff034e"  // Your Temboo app key

#define ETHERNET_SHIELD_MAC { 0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED };
byte ethernetMACAddress[] = ETHERNET_SHIELD_MAC;
EthernetClient tclient;
/*********************************************/


void setup() {
  // turn on LED to signal the start of the calibration period:
  pinMode(ledPin, OUTPUT);
  digitalWrite(ledPin, HIGH);
  Serial.begin(9600);

  Ethernet.begin(mac, ip);  // initialize Ethernet device

  // calibrate during the first five seconds
  while (millis() < 5000) {
    sensorValue = analogRead(sensorPin);
    // record the maximum sensor value
    if (sensorValue > sensorMax) {
      sensorMax = sensorValue;
    }
    // record the minimum sensor value
    if (sensorValue < sensorMin) {
      sensorMin = sensorValue;
    }
  }
  Serial.println(sensorMin);  //print minumum value selected
  Serial.println(sensorMax);  //print maximum value selected

   /***************** Temboo ******************************/
  Serial.print("DHCP:");
  if (Ethernet.begin(ethernetMACAddress) == 0) {
    Serial.println("FAIL");
    while(true);
  }
  Serial.println("OK");
   /*****************************************************/

  // signal the end of the calibration period
  digitalWrite(ledPin, LOW);
}

void loop() {
  EthernetClient client = server.available();

 /*-------------------------check for user server request---------------------------------------*/
  if (client) {  // got client?
    boolean currentLineIsBlank = true;
    while (client.connected()) {
      if (client.available()) {   // client data available to read
        char c = client.read(); // read 1 byte (character) from client
        HTTP_req += c;  // save the HTTP request 1 char at a time
        // last line of client request is blank and ends with \n
        // respond to client only after last line received
        if (c == '\n' && currentLineIsBlank) {
          // send a standard http response header
          client.println("HTTP/1.1 200 OK");
          client.println("Content-Type: text/html");
          client.println("Connection: close");
          client.println();
          // send web page
          client.println("<!DOCTYPE html>");
          client.println("<html>");
          client.println("<head>");
          client.println("<title>Arduino LED Control</title>");
          client.println("</head>");
          client.println("<body>");
          client.println("<h1>Arduino LED Control</h1>");
          client.println("<form method='get' >");
          client.println("<input type='radio' name='status' value='0' />Off &nbsp;&nbsp;&nbsp;");
          client.println("<input type='radio' name='status' value='1' checked />On  &nbsp;&nbsp;&nbsp;");
          client.println("<input type='radio' name='status' value='2' />Manual  &nbsp;&nbsp;&nbsp;<br/>");
          client.println("<select name='val'>");
          client.println("<option value='0'>10%</option><option value='1'>20%</option>");
          client.println("<option value='2'>30%</option><option value='3'>40%</option>");
          client.println("<option value='4'>50%</option><option value='5'>60%</option>");
          client.println("<option value='6'>70%</option><option value='7'>80%</option>");
          client.println("<option value='8'>90%</option><option value='9'>100%</option>");
          client.println("</select>");
          client.println("<input type='submit' value='update' />");
          client.println("</form>");
          client.println("</body>");
          client.println("</html>");
          ProcessForm();
          Serial.print(HTTP_req);
          HTTP_req = "";    // finished with request, empty string
          break;
        }//if c == '\n'
        // every line of text received from the client ends with \r\n
        if (c == '\n') {
          // last character on line of received text
          // starting new line with next character read
          currentLineIsBlank = true;
        }
        else if (c != '\r') {
          // a text character was received from client
          currentLineIsBlank = false;
        }
      }//end  if client.available()
    }//end  while (client.connected())
    delay(10);      // give the web browser time to receive the data
    client.stop(); // close the connection
  }// end if  (client)

 /*-----------------------------------------------------------------------------------------------------------*/
/*------------------------------- Update Led brightness ---------------------------------------*/
  // read the sensor:
  sensorValue = analogRead(sensorPin);
  if(LedStatus==1){ // see if  On selescted
    // in case the sensor value is outside the range seen during calibration
    sensorConstrainValue = constrain(sensorValue, sensorMin, sensorMax);
    // apply the calibration to the sensor reading
    sensorConstrainValue = map(sensorConstrainValue, sensorMin, sensorMax, 0, 255);
  }
  else if(LedStatus==2){  // see if  Manual selescted
    sensorConstrainValue = map(sensorConstrainValue, 10, 100, 0, 255);
  }
  else{// see if  Off selescted
    sensorConstrainValue=0;
  }
  // fade the LED using the calibrated value:
  analogWrite(ledPin, sensorConstrainValue);
/*-----------------------------------------------------------------------------------------------------------*/

/*----------------------------------  Temboo log Google spredsheet----------------------------------------*/
  if(sensorValue<sensorMin || sensorValue>sensorMax){
    TembooChoreo AppendRowChoreo(tclient);
    // Invoke the Temboo client
    AppendRowChoreo.begin();

    // Set Temboo account credentials
    AppendRowChoreo.setAccountName(TEMBOO_ACCOUNT);
    AppendRowChoreo.setAppKeyName(TEMBOO_APP_KEY_NAME);
    AppendRowChoreo.setAppKey(TEMBOO_APP_KEY);

    // Set Choreo inputs
    String ClientSecretValue = "YFfokVLQqCYcIzKD74Ub_Zab";
    AppendRowChoreo.addInput("ClientSecret", ClientSecretValue);
    String RefreshTokenValue = "1/jRSVdtwz3ITcKeXJrLlKydCUMCR0MDLAfc5c10GhVsI";
    AppendRowChoreo.addInput("RefreshToken", RefreshTokenValue);
    String RowDataValue = "=TODAY(),=myFunction(),";
    RowDataValue.concat(sensorMin);RowDataValue.concat(",");
    RowDataValue.concat(sensorMax);RowDataValue.concat(",");
    RowDataValue.concat(sensorValue);
    AppendRowChoreo.addInput("RowData", RowDataValue);
    String SpreadsheetTitleValue = "IOTTest";
    AppendRowChoreo.addInput("SpreadsheetTitle", SpreadsheetTitleValue);
    String ClientIDValue = "32386991167-5h0oro6lia6t7v69e31083r6m2b5h3ms.apps.googleusercontent.com";
    AppendRowChoreo.addInput("ClientID", ClientIDValue);

    // Identify the Choreo to run
    AppendRowChoreo.setChoreo("/Library/Google/Spreadsheets/AppendRow");

    // Run the Choreo; when results are available, print them to serial
    AppendRowChoreo.run();

    while(AppendRowChoreo.available()) {
      char c = AppendRowChoreo.read();
      Serial.print(c);
    }
    AppendRowChoreo.close();
  }
  /*-----------------------------------------------------------------------------------------------------------*/
}


// Change parameter according to status
void ProcessForm()
{
  if (HTTP_req.indexOf("status=0") > -1) {  // see if  Off selescted
    LedStatus=0;
  }
  else if (HTTP_req.indexOf("status=1") > -1) {  // see if  Onselescted
    LedStatus=1;
  }
  else if (HTTP_req.indexOf("status=2") > -1) {  // see if  Manual selescted
    LedStatus=2;
    int index=HTTP_req.indexOf("val=");
    ManualValue=int(HTTP_req.charAt(index+4));
    ManualValue=(ManualValue*10)+10;
  }
}
```