```
#include <ESP8266WiFi.h>
#include <WiFiClientSecure.h>
#include "DHT.h"
#define DHTTYPE DHT11
#define ON_Board_LED 2

const int DHTPin = 5;
DHT dht(DHTPin, DHTTYPE);
const char* ssid = "ssid";
const char* password = "pass";
const char* host = "script.google.com";
WiFiClientSecure client;
String ID = "AKfycbyCivA_50thygQkMSnWI1o2CcY3Z7bCkdzaqByRHy2s0OnrSri2tKFo-
YbRFK536Bkr5g";

void setup() {
  Serial.begin(9600);
  delay(500);
  dht.begin();
  delay(500);
  WiFi.begin(ssid, password);
  Serial.println("");
  pinMode(ON_Board_LED,OUTPUT);
  digitalWrite(ON_Board_LED, HIGH);
  Serial.print("Connecting");
  while (WiFi.status() != WL_CONNECTED) {
    Serial.print(".");
    digitalWrite(ON_Board_LED, LOW);
    delay(250);
    digitalWrite(ON_Board_LED, HIGH);
    delay(250);
```

```
   }
  digitalWrite(ON_Board_LED, HIGH);
  Serial.println("");
  Serial.print("Successfully connected to : ");
  Serial.println(ssid);
  Serial.print("IP address: ");
  Serial.println(WiFi.localIP());
  Serial.println();
  client.setInsecure();
}

void loop() {
  int h = dht.readHumidity();
  float t = dht.readTemperature();
  if (isnan(h) || isnan(t)) {
    Serial.println("Failed to read from DHT sensor !");
    delay(500);
    return;
  }
  String Temp = "Temperature : " + String(t) + " °C";
  String Humi = "Humidity : " + String(h) + " %";
  Serial.println(Temp);
  Serial.println(Humi);

  sendData(t, h); //--> Calls the sendData Subroutine
  delay(30000);
}

void sendData(float tem, int hum) {
  Serial.println("==========");
  Serial.print("connecting to ");
  Serial.println(host);
  if (!client.connect(host, httpsPort)) {
    Serial.println("connection failed");
    return;
  }
  String string_temperature =  String(tem);
  String string_temperature =  String(tem, DEC);
  String string_humidity =  String(hum, DEC);
  String url = "/macros/s/" + _ID + "temperature=" + string_temperature + "&humidity=" +
string_humidity;
  Serial.print("requesting URL: ");
  Serial.println(url);

  client.print(String("GET ") + url + " HTTP/1.1\r\n" +
         "Host: " + host + "\r\n" +
         "User-Agent: BuildFailureDetectorESP8266\r\n" +
         "Connection: close\r\n\r\n");

  Serial.println("request sent");
  while (client.connected()) {
    String line = client.readStringUntil('\n');
    if (line == "\r") {
      Serial.println("headers received");
      break;
    }
  }
  String line = client.readStringUntil('\n');
  if (line.startsWith("{\"state\":\"success\"")) {
    Serial.println("esp8266/Arduino CI successfull!");
  } else {
    Serial.println("esp8266/Arduino CI has failed");
  }
  Serial.println("closing connection");
}
```

```
function doGet(e) {
  Logger.log( JSON.stringify(e) );
  var result = 'Ok';
  if (e.parameter == 'undefined') {
    result = 'No Parameters';
  }
  else {
    var sheet_id = '';         // Spreadsheet ID
    var sheet = SpreadsheetApp.openById(sheet_id).getActiveSheet();
    var newRow = sheet.getLastRow() + 1;
    var rowData = [];
    var Curr_Date = new Date();
    rowData[0] = Curr_Date; // Date in column A
    var Curr_Time = Utilities.formatDate(Curr_Date, "Asia/Jakarta", 'HH:mm:ss');
    rowData[1] = Curr_Time; // Time in column B
    for (var param in e.parameter) {
      Logger.log('In for loop, param=' + param);
      var value = stripQuotes(e.parameter[param]);
      Logger.log(param + ':' + e.parameter[param]);
      switch (param) {
        case 'temperature':
          rowData[2] = value; // Temperature in column C
          result = 'Temperature Written on column C';
          break;
        case 'humidity':
          rowData[3] = value; // Humidity in column D
          result += ' ,Humidity Written on column D';
          break;
        default:
          result = "unsupported parameter";
      }
    }
    Logger.log(JSON.stringify(rowData));
    var newRange = sheet.getRange(newRow, 1, 1, rowData.length);
    newRange.setValues([rowData]);
  }
  return ContentService.createTextOutput(result);
}
function stripQuotes( value ) {
  return value.replace(/^["']|['"]$/g, "");
}
```