

|  |
| --- |
| **Systèmes Embarqués II** |
| Google Sheet |
| ISAT – EPHEC 2020-2021  Juan Alvarez et Olivier Grabenweger |

Table des matières

[1 Introduction 2](#_Toc60834207)

[2 Mindmapping - Logigramme 2](#_Toc60834208)

[a) Mindmap 2](#_Toc60834209)

[b) Logigramme 3](#_Toc60834210)

[3 Schéma de câblage 4](#_Toc60834211)

[a) Listes des composants : 4](#_Toc60834212)

[4 Code source 5](#_Toc60834213)

[a) Code ESP8266 : 5](#_Toc60834214)

[b) Script Google 8](#_Toc60834215)

[5 Conclusion 9](#_Toc60834216)

[6 Annexes et bibliographie 9](#_Toc60834217)

[a) Annexes : 9](#_Toc60834218)

[b) Bibliographies : 9](#_Toc60834219)

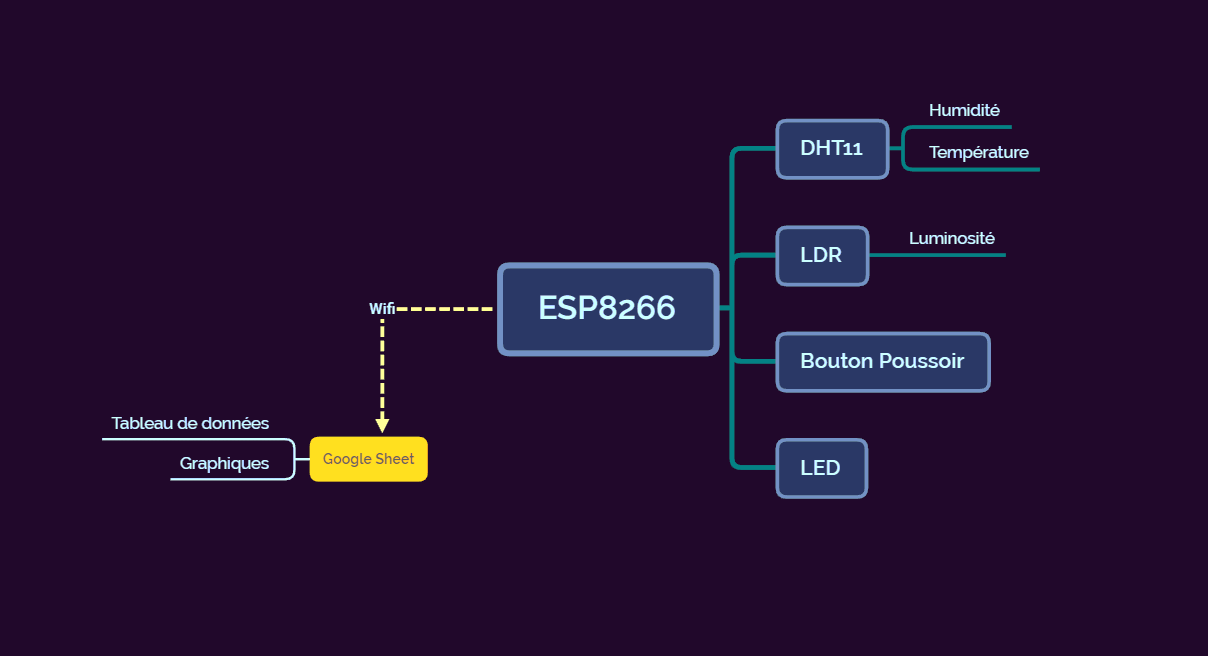
# Introduction

Le but de ce TP est de parvenir à envoyer les données de nos capteurs vers un google Sheet afin de faire un tableau de données et des graphiques. Les données envoyées via notre ESP8266 par WiFi sont la température et l’humidité grâce au DHT11 et la luminosité grâce au LDR.

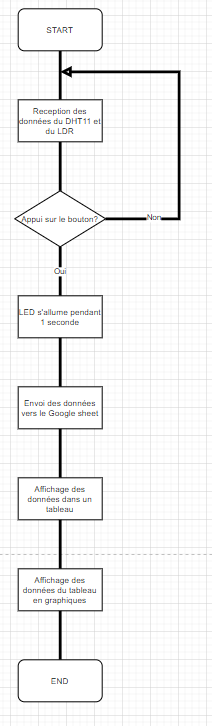
Pour cela, nous devons donc préparer et lier un google sheet et le script associé avec le code de l’ESP8266 qui transmettra les données de nos capteurs.

# Mindmapping - Logigramme

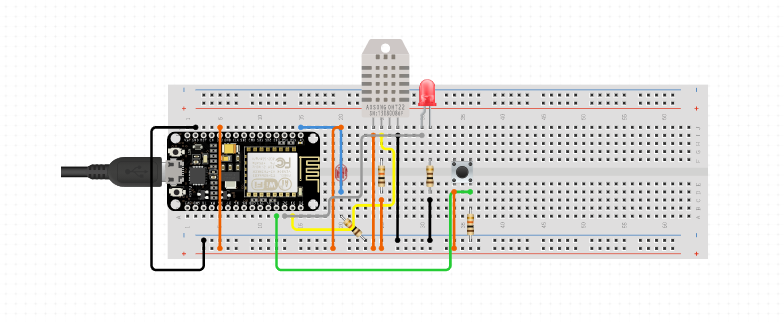
## Mindmap



## Logigramme



# Schéma de câblage



! Attention aux PINs qui ne reflètent pas celles utilisées dans le code !

## Listes des composants :

* 1 ESP8266
* 1 Bred Board
* 1 LED
* 1 LDR
* 1 DHT11
* 1 Bouton Poussoir
* 3 Résistances de 10k Ohm
* 1 Résistance de 330 Ohm

# Code source

## Code ESP8266 :

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90  91  92  93  94  95  96  97  98  99  100  101  102  103  104  105  106  107  108  109  110  111  112  113  114  115  116  117  118  119  120  121  122  123  124  125  126  127  128  129  130  131  132  133  134  135  136  137  138  139  140  141  142  143  144  145  146  147  148  149  150  151  152  153  154  155 | /\* Bibliothèques \*/  #include <Arduino.h>  #include <ESP8266WiFi.h>  #include <WiFiClientSecure.h>  #include "DHT.h"  /\* Variables \*/  WiFiClientSecure gsclient;  String GAS\_ID = "AKfycbybjsotK0WczFlCYcgxVW9FIZJPjjyWERmYpc7iVP3F1HJZBrA"; //getactivespreadsheetID  **const** **char**\* fingerprint = "46 B2 C3 44 9C 59 09 8B 01 B6 F8 BD 4C FB 00 74 91 2F EF F6"; //génération d'un fingerprint  **const** **char**\* host = "script.google.com"; // configuration de l'hôte  **const** **int** httpsPort = **443**; // configuration du port https  #ifndef STASSID //configuration nom du Wi-Fi et de son mot de passe  #define STASSID "bbox-Sophie1" // nom du Wi-Fi  #define STAPSK "20150509Sophi" // mot de passe du Wi-Fi  #endif  **const** **char**\* ssid = STASSID;  **const** **char**\* password = STAPSK;  **const** **int** BUTTON = D0; // Lier la Pin D0 à notre bouton  **const** **int** LEDred = D1; // Lier la Pin D1 à notre LED rouge  **const** **int** LDR = A0; // Lier la Pin A0 à notre capteur de Luminosité  **float** humy; // Variable pour l'humidité  **float** temp; // Variable pour la température  **int** lumi; // Variable pour la luminosité  **int** state\_button = **0**; // Variable pour l'état de notre bouton  **int** flag\_button = **0**; // flag du bouton  **int** count = **0**; // compteur du nombre d'appui sur le bouton  #define DHTPIN D6 // Lier la Pin D6 à notre capteur DHT  #define DHTTYPE DHT11 // Préciser le type de DHT, ici le DHT 11  DHT **dht**(DHTPIN, DHTTYPE); // configuration de notre DHT en précisant via nos variables: La Pin et le type de DHT  /\* Fonction du Bouton poussoir \*/  **void** **button\_Read**() //cette fonction va servir à ce que pour un appui il y ai un envoi de donnée  {  state\_button = digitalRead(BUTTON); // Lire l'état du bouton et l'attribuer à notre variable state\_button  **if** (state\_button == LOW) // comme le bouton est en pull-up on met le flag à 1 quand l'état du bouton est BAS  {  flag\_button = **1**;  }  **if** (flag\_button == **1** && state\_button == HIGH)  {  count += **1**; //incrémentation du compteur  flag\_button = **0**;  }  }  /\* Fonction d'envoi de données \*/  **void** **sendData**(**int** x, **int** y, **int** z) // prend comme paramètre les données que l'on souhaite envoyer, ici température, humidité et luminosité  {  Serial.print("connecting to ");  Serial.println(host);  gsclient.setInsecure();  **if** (!gsclient.connect("script.google.com", httpsPort)) // on vérifie le port et le fingerprint  {  Serial.println("connection failed");  **return**;  }  **if** (gsclient.verify(fingerprint, host)) // on vérifie le port et le fingerprint  {  Serial.println("certificate matches");  }  **else**  {  Serial.println("certificate doesn't match");  }  // x = température, y = humidité et z = luminosité  String string\_x = String(x, DEC); // conversion de x en string  String string\_y = String(y, DEC); // conversion de y en string  String string\_z = String(z, DEC); // conversion de z en string  Serial.println(string\_x);  Serial.println(string\_y);  Serial.println(string\_z);  //création de l'url qui enverra les données sur le google sheet  String url = "/macros/s/" + GAS\_ID + "/exec?temperature=" + string\_x + "&humidity="+ string\_y + "&luminosity="+ string\_z;  Serial.print("requesting URL: ");  Serial.println(url);  gsclient.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** (gsclient.connected()) {  String line = gsclient.readStringUntil('\n');  **if** (line == "**\r**")  {  Serial.println("headers received");  **break**;  }  }  String line = gsclient.readStringUntil('\n');  Serial.println(line);  **if** (line.startsWith("{**\"**state**\"**:**\"**success**\"**"))  {  Serial.println("esp8266/Arduino CI successfull!");  }  **else**  {  Serial.println("esp8266/Arduino CI has failed");  }  Serial.println("reply was:");  Serial.println("closing connection");  }  /\* fonction setup \*/  **void** **setup**()  {  Serial.begin (**9600**);  pinMode(LEDred, OUTPUT); // défini la LED en output  digitalWrite(LEDred, LOW); // Eteind la LED  pinMode(BUTTON, INPUT); // défini le bouton en input  dht.begin(); // Initialise le DHT  WiFi.mode(WIFI\_STA); // Précise le mode du WiFi  WiFi.begin(ssid, password); // Attribue le nom et le mot de passe du wifi  // Wait for connection  **while** (WiFi.status() != WL\_CONNECTED) { // Tant que la connection au wifi n'est pas établie, en imprime des points sur le terminal  delay(**1000**);  Serial.print(".");  }  }  /\* fonction loop \*/  **void** **loop**()  {  button\_Read();  **float** temp = dht.readTemperature(); //on lit la valeur de la température du DHT et on l'attibue à une variable  **float** humy = dht.readHumidity(); //on lit la valeur de l'humidité du DHT et on l'attibue à une variable  **int** lumi = analogRead(LDR); //on lit la valeur de la luminosité du capteur LDR et on l'attibue à une variable  **if** (flag\_button == **1**) // On envoie nos données quand on appuie sur le bouton et pour le vérifier une LED s'alume brièvement  {  sendData(temp,humy,lumi);  Serial.println(temp);  Serial.println(humy);  Serial.println(lumi);  digitalWrite(LEDred, HIGH);  delay(**100**);  digitalWrite(LEDred, LOW);  }  } |

## Script Google

|  |  |
| --- | --- |
|  |  |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48 | function doGet(e) {  Logger.log( JSON.stringify(e) ); // view parameters  var result = 'Ok'; // assume success  if (e.parameter == 'undefined') {  result = 'No Parameters';  }  else {  var sheet\_id = '1NzX6m5DuVqKsICEiNpMvH5jggmSMWx3CYS1PRBigUyc'; // Sheet ID  var sheet = SpreadsheetApp.openById(sheet\_id).getActiveSheet(); // get Active sheet //getSheetByName('nameOfSheet') for another sheet  var newRow = sheet.getLastRow() + 1;  var rowData = [];  rowData[0] = new Date(); // Timestamp in column A  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': //Parameter  rowData[1] = value; //Value in column B  result = 'Write on Column B';  break;  case 'humidity': //Parameter  rowData[2] = value; //Value in column C  result += ', Write on column 3';  break;  case 'luminosity': //Parameter  rowData[3] = value; //Value in column D  result += ', Write on column 4';  break;  default:  result = "unsupported parameter";  }  }  Logger.log(JSON.stringify(rowData));  // Write new row below  var newRange = sheet.getRange(newRow, 1, 1, rowData.length);  newRange.setValues([rowData]);  }  // Return result of operation  return ContentService.createTextOutput(result);  }  /\*\*  \* Remove leading and trailing single or double quotes  \*/  function stripQuotes( value ) {  return value.replace(/^["']|['"]$/g, "");  } |

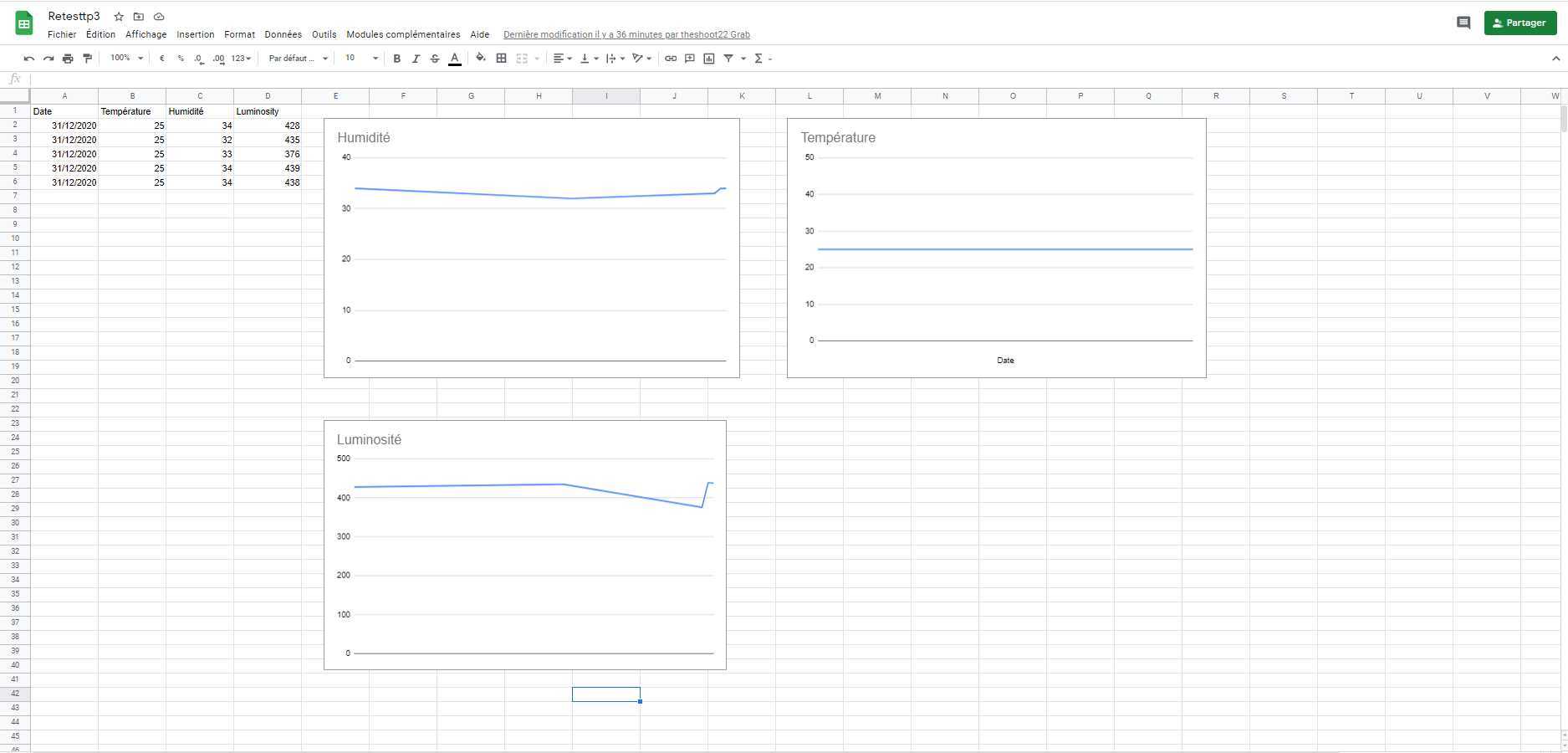
# Conclusion

Finalement, nous découvrons grâce à ce TP combien il est simple et pratique d’utiliser un outil tel que google Sheet afin de répertorier et traiter de simples données. En faisant des recherches sur le sujet, on a remarqué qu’il y a beaucoup plus de gens qui utilisent cet outil que nous le pensions.

Ça nous a aussi donné envie de réutiliser nos connaissances acquises ici dans le TP sur nodered.

# Annexes et bibliographie

## Annexes :



## Bibliographies :

* https://script.google.com/
* <https://docs.google.com/spreadsheets/>
* <https://www.circuito.io/> : schéma de câblage
* <http://draw.io/> : organigramme
* Xmind : Mindmap