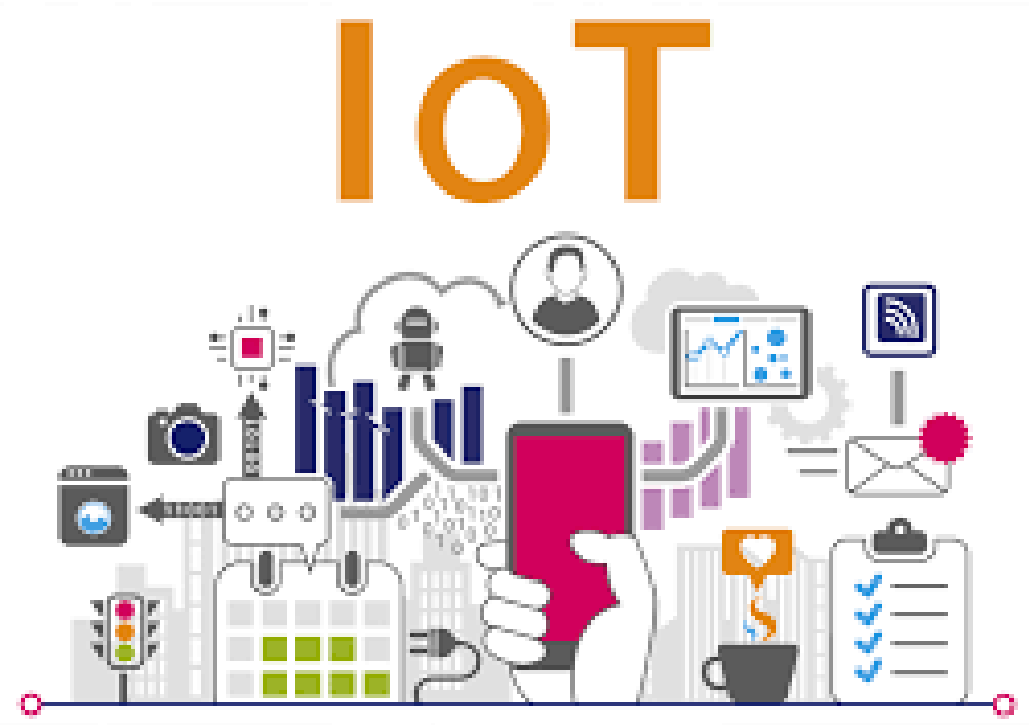
RAPPORT D'EXPÉRIENCE AU LABORATOIRE

TP1 : Serveur Web ESP8266 utilisant SPIFFS (Système de fichiers Flash avec SPI)



Guerziz Ines

12/11/2022

Spécialité :ISI G3

PARTIE -1-

Objectif:

Le sujet de ce TP est de créer un serveur Web contenant des fichiers HTML et CSS stockés sur le système de fichiers ESP8266 (SPIFFS) à l'aide d'IDE Arduino ou platformio avec Visual Studio Code.Au lieu d'avoir à écrire le texte HTML,javascript et CSS dans l'esquisse Arduino, en créant des fichiers HTML et CSS distincts.

Matériel:

1. ESP8266
2. Led
3. fils mâle-mâle
4. DHT11

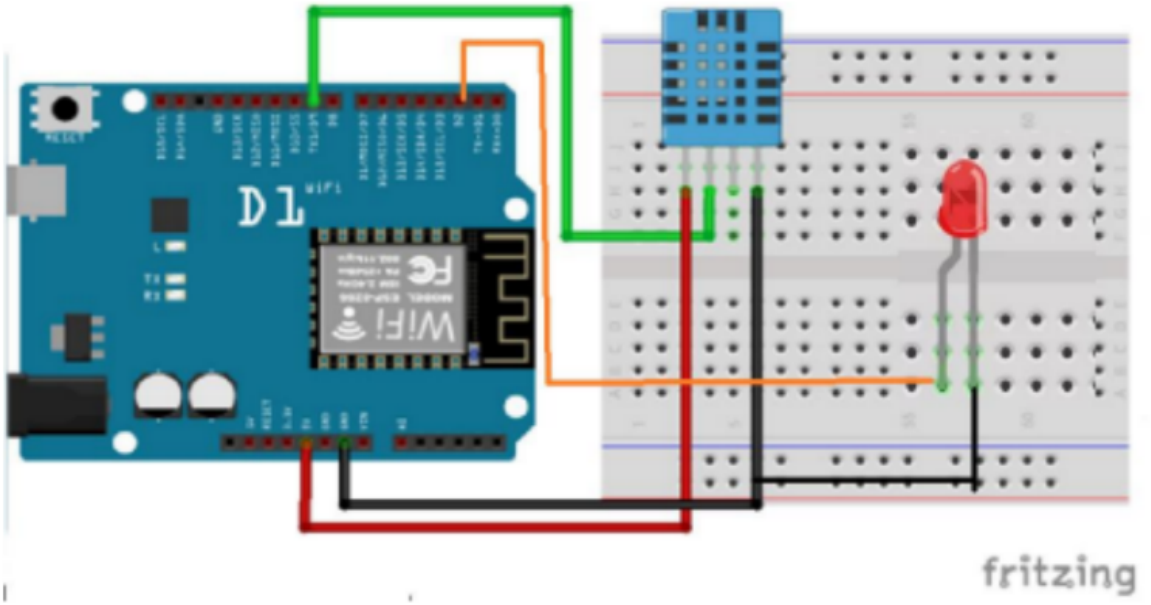
Procédure:

1. Le serveur Web contrôle une LED connecté à l'ESP8266 GPIO 2.
2. La page du serveur Web affiche deux boutons: ON et OFF - pour activer et désactiver GPIO 2;
3. La page du serveur Web affiche également l'état actuel du GPIO.
4. Utiliser également un capteur DHT11 pour afficher les lectures du capteur
5. (température, humidité).

Schéma explicatif:

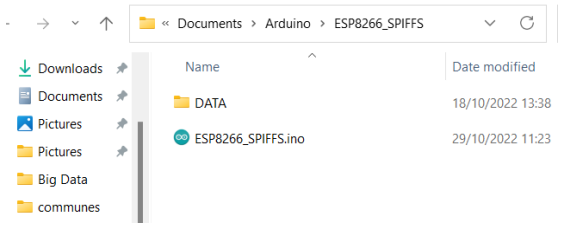
La figure suivante montre un schéma simplifié montrant comment tout fonctionne.

Diagramme schématique:



Organisation des fichiers:

Pour construire le serveur Web, nous avons besoin de trois fichiers différents. L'esquisse Arduino, le fichier HTML et deux dossiers js et CSS. Les fichiers HTML et les dossier CSS,js doivent être enregistrés dans un dossier appelé data dans le dossier Esquisse Arduino, comme indiqué ci-dessous:



Code:

File Edit Sketch Tools Help

ESP8266_SPIFFS

```
#include <ESP8266WiFi.h>
#include <WiFiClient.h>
#include <ESP8266WebServer.h>
#include <FS.h>

//#include "index.h" //Our HTML webpage contents with javascripts
//#include "DHTesp.h"  //DHT11 Library for ESP

#define LED 3        //On board LED

//SSID and Password of your WiFi router
const char* ssid = "INES";
const char* password = "ines12345678";

ESP8266WebServer server(80); //Server on port 80

//===============================================================
// This routine is executed when you open its IP in browser
//===============================================================

float humidity, temperature;

void handleTH() {

 String data = "{\"Temperature\":\""+ String(temperature) +"\", \"Humidity\":\""+ String(humidity) +"\"}";

  server.send(200, "text/plane", data); //Send ADC value, temperature and humidity JSON to client ajax request


  humidity +=1.;// dht.getHumidity();
  temperature +=2.;// dht.getTemperature();

  Serial.print(humidity, 1);
  Serial.println(temperature, 1);
  //Serial.print(dht.toFahrenheit(temperature), 1);
}
```

ESP8266_SPIFFS

```
void setup()
{


//make the LED pin output and initially turned off
 pinMode(LED, OUTPUT);
 digitalWrite(LED, LOW);
 Serial.begin(115200);
 Serial.println();


 if(!SPIFFS.begin()){
   Serial.println("An Error has occurred while mounting SPIFFS");
   return;
 }


 WiFi.begin(ssid, password);     //Connect to your WiFi router
 Serial.println("");

 //Onboard LED port Direction output


 // Wait for connection
 while (WiFi.status() != WL_CONNECTED) {
   delay(500);
   Serial.print(".");
 }


 //If connection successful show IP address in serial monitor
 Serial.println("");
 Serial.print("Connected to ");
 Serial.println(ssid);
 Serial.print("IP address: ");
 Serial.println(WiFi.localIP());  //IP address assigned to your ESP
 server.on("/LEDOn", [](){
   digitalWrite(LED, LOW);
   Serial.println("on");
 });
 server.on("/LEDOff", [](){
   digitalWrite(LED, HIGH);
   Serial.println("off");
 ..
```

```
  server.serveStatic("/css", SPIFFS, "/css");
  server.serveStatic("/js", SPIFFS, "/js");
  server.serveStatic("/", SPIFFS, "/index.html");
  server.on("/readTH", handleTH); //This page is called by java Script AJAX
  server.begin();                  //Start server
  Serial.println("HTTP server started");
}

//================================================================
//                    LOOP
//================================================================
void loop()
{
  server.handleClient();          //Handle client requests
}
```

- La fonction HandleTH() est responsable de la mise à jour de la variable de l'humidité et la température
- La fonction Setup:la mise de LED en OUTPUT
- Démarrer Serial avec débit de bauds de 115200 pour afficher l'adresse IP sur le moniteur sèrie
- avec SPIFFS.begin(),l'initialisation de notre système de fichiers en mémoire flash.
- Si une erreur se produit ,notre code s'arrête ici et le signale eu moniteur sèrie.Dans ce cas le téléchargement à nouveau de fichiers
- la structure "server.on" est un gestionnaire d'événement fourni par la librairie ESPAsyncWebServer.h
- Après la compilation et l'exécution de code,on vérifiant l'adresse IP du serveur web dans le moniteur série et accédez-y dans le navigateur

PARTIE 2 et 3

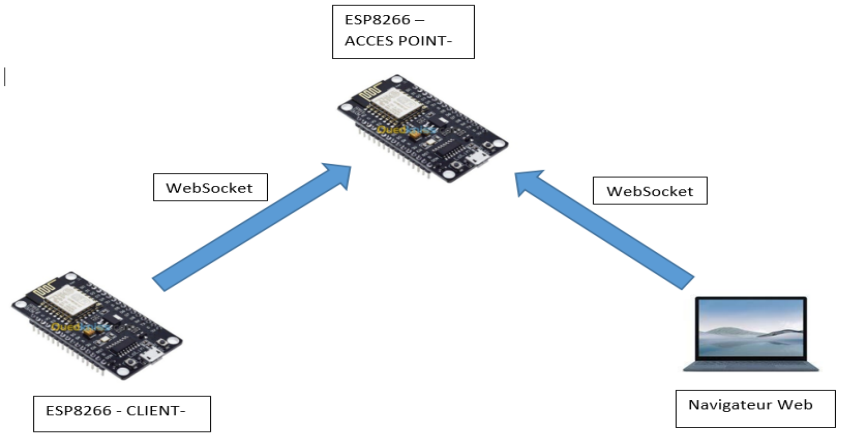
Objectif:

Établir une architecture de réseaux de capteurs avec un nœud principal comme point d'accès qui héberge le webserver pour l'affichage et des nœuds auxiliaires pour la collecte des données.en utilisant une communication avec WEBSOCKET

Matériel:

1. 2 ESP8266
2. 2 BreadBoard
3. DHT 11 sensor
4. Led green
5. Résistance
6. Ecran LCD
7. 2 Câbles USB
8. Cables male-male

Schéma explicatif:



- Nous avons 2 circuits ,l'un est pour le serveur WebSocket ESP8266 et un autre pour le client Websocket ESP8266
- lorsqu'il est allumé le client peut se connecter au serveur via une connexion websocket et obtenir des données en temps réel

Code:

Server

TP2-server

```
#include <WiFiClient.h> // Include WIFi Library for ESP32
#include <ESP8266WebServer.h> // Include WebSwever Library for ESP32
#include <ArduinoJson.h> // Include ArduinoJson Library
#include "DHT.h" // Include DHT Library
#include <WebSocketsServer.h>  // Include Websocket Library
#define DHTPIN 13 // DHT PIn
#define DHTTYPE DHT11 // DHT Type
const char* ssid = "ESP8266";  // Your SSID
const char* password = "ines12345678"; // Your Password
int interval = 1000; // virtual delay
unsigned long previousMillis = 0; // Tracks the time since last event fired
String web ="<!DOCTYPE html><html><head> <title>Websocket</title> <meta name='viewport' content='width=devic
String jsonString; // Temporary storage for the JSON String
String pin_status = ""; // Holds the status of the pin
float t; // holds the temperature value
float h;// holds the Humidity value
DHT dht(DHTPIN, DHTTYPE); // create instance for DHT sensor
ESP8266WebServer server(80);  // create instance for web server on port "80"
WebSocketsServer webSocket = WebSocketsServer(81);  //create instance for webSocket server on port"81"
void webSocketEvent(byte num, WStype_t type, uint8_t * payload, size_t length) {
  switch (type) {
    case WStype_DISCONNECTED: // enum that read status this is used for debugging.
      Serial.print("WS Type ");
      Serial.print(type);
      Serial.println(": DISCONNECTED");
      break;
    case WStype_CONNECTED:  // Check if a WebSocket client is connected or not
      Serial.print("WS Type ");
      Serial.print(type);
      Serial.println(": CONNECTED");
      if (digitalRead(5) == HIGH) {  //check if pin 22 is high or low
        pin_status = "ON";
        update_webpage(); // update the webpage accordingly
      }
      else {
        pin_status = "OFF"; //check if pin 22 is high or low
        update_webpage();// update the webpage accordingly
```

TP2-server

```
  object["Hum"] = h ;
  serializeJson(doc, jsonString); // serialize the object and save teh result to teh string variable.
  Serial.println( jsonString ); // print the string for debugging.
  webSocket.broadcastTXT(jsonString); // send the JSON object through the websocket
  jsonString = ""; // clear the String.
}
void setup() {
  // put your setup code here, to run once:
  pinMode(5, OUTPUT); // Set PIN5 As output(LED Pin)
  Serial.begin(115200);
  Serial.print("Configuring access point...");// Init Serial for Debugging.
  WiFi.softAP(ssid); // Connect to Wifi
  IPAddress myIP =WiFi.softAPIP();

  Serial.println();
  // Print the IP address in the serial monitor windows.
  Serial.print("IP address: ");
  Serial.println(myIP);
  // Initialize a web server on the default IP address. and send the webpage as a response.
  server.on("/", []() {
    server.send(200, "text\html", web);
  });
  server.begin(); // init the server
  webSocket.begin();  // init the Websocketserver
  webSocket.onEvent(webSocketEvent);  // init the webSocketEvent function when a websocket event occurs
  dht.begin(); // Init DHT sensor
}
void loop() {
  server.handleClient();  // webserver methode that handles all Client
  webSocket.loop(); // websocket server methode that handles all Client
  unsigned long currentMillis = millis(); // call millis  and Get snapshot of time
  if ((unsigned long)(currentMillis - previousMillis) >= interval) { // How much time has passed, account
    update_temp_hum(); // update temperature data.
    update_webpage(); // Update Humidity Data
    previousMillis = currentMillis;   // Use the snapshot to set track time until next event

  // This function gets a call when a WebSocket event occurs
```

Client

TP2-client

```
#include <WiFiServer.h>
#include <ESP8266WebServer.h>
#include <WebSocketsClient.h>
#include <ArduinoJson.h>
#include "rgb_lcd.h"
rgb_lcd lcd;
const int colorR=255;
const int colorG=0;
const int colorB=0;
// Wifi Credentials
const char* ssid = "ESP8266"; // Wifi SSID
const char* password = "ines12345678"; //Wi-FI Password
WebSocketsClient webSocket; // websocket client class instance
StaticJsonDocument<100> doc; // Allocate a static JSON document
void webSocketEvent(WStype_t type, uint8_t * payload, size_t length) {
  // Make sure the screen is clear
  //  u8g2.clearBuffer();
  if (type == WStype_TEXT)
  {
    DeserializationError error = deserializeJson(doc, payload); // deserialize incoming Jsor
    if (error) { // Print erro msg if incomig String is not JSON formated
      Serial.print(F("deserializeJson() failed: "));
      Serial.println(error.c_str());
      return;
    }
    const String pin_stat = doc["PIN_Status"]; // String variable tha holds LED status
    const float t = doc["Temp"]; // Float variable that holds temperature
    const float h = doc["Hum"]; // Float variable that holds Humidity
    // Print the received data for debugging
    Serial.print(String(pin_stat));
    Serial.print(String(t));
    Serial.println(String(h));
    // Send acknowledgement
    // webSocket.sendTXT("OK");
    // LED: OFF
      // TMP: Temperature
     //  Hum: Humidity

    lcd.setCursor(0, 1); // Set the cursor position to (0,0)
    lcd.print("LED:"); // Print LED: on the display
    lcd.print(pin_stat); // print LED Status to the display
```

TP2-client

```
    lcd.print(pin_stat); // print LED Status to the display
     lcd.print("    "); // Set the cursor
     lcd.print("TMP:"); // Print TMP: on the display
      lcd.print(t); // Print temperature value
      lcd.print("    ");

     lcd.print("HUM:"); // Print TMP: on the display
      lcd.print(h); // Print temperature value
      lcd.print("    ");
      lcd.display();

  }
}

void setup() {
  // Connect to local WiFi
  lcd.begin(16,2);
  lcd.setRGB(colorR,colorG,colorB);
  WiFi.begin(ssid);
  Serial.begin(115200);
  while (WiFi.status() != WL_CONNECTED) {
    Serial.print(".");
    delay(500);
  }
  Serial.println();
  Serial.print("IP Address: ");
  Serial.println(WiFi.localIP()); // Print local IP address

  delay(2000); // wait for 2s


  //address, port, and URL path
  webSocket.begin("192.168.4.1", 81, "/");
  // webSocket event handler
  webSocket.onEvent(webSocketEvent);
  // if connection failed retry every 5s
  webSocket.setReconnectInterval(5000);
}
void loop() {
  webSocket.loop(); // Keep the socket alive
```

```
}
void setup() {
  // put your setup code here, to run once:
  pinMode(5, OUTPUT); // Set PIN5 As output(LED Pin)
  Serial.begin(115200);
  Serial.print("Configuring access point...");// Init Serial for Debugging.
  WiFi.softAP(ssid); // Connect to Wifi
  IPAddress myIP =WiFi.softAPIP();
```

On utilise la fonction Wifi.softAP() pour configurer le point d'accès

```
  Serial.println(myIP);
  // Initialize a web server on the default IP address. and send the webpage as a re
  server.on("/", []() {
    server.send(200, "text\html", web);
  });
  server.begin(); // init the server
  webSocket.begin();  // init the Websocketserver
```

La connexion Websocket sera commencé et on initialize également un gestionnaire d'événement pour le Websocket,le gestionnaire appelle la fonction WebSocket Event lorsqu'un événement WebSocket se produit

```
void loop() {
  server.handleClient();  // webserver methode that handles all Cli
  webSocket.loop(); // websocket server methode that handles all Cl
  unsigned long currentMillis = millis(); // call millis  and Get s
  if ((unsigned long)(currentMillis - previousMillis) >= interval)
    update_temp_hum(); // update temperature data.
    update_webpage(); // Update Humidity Data
```

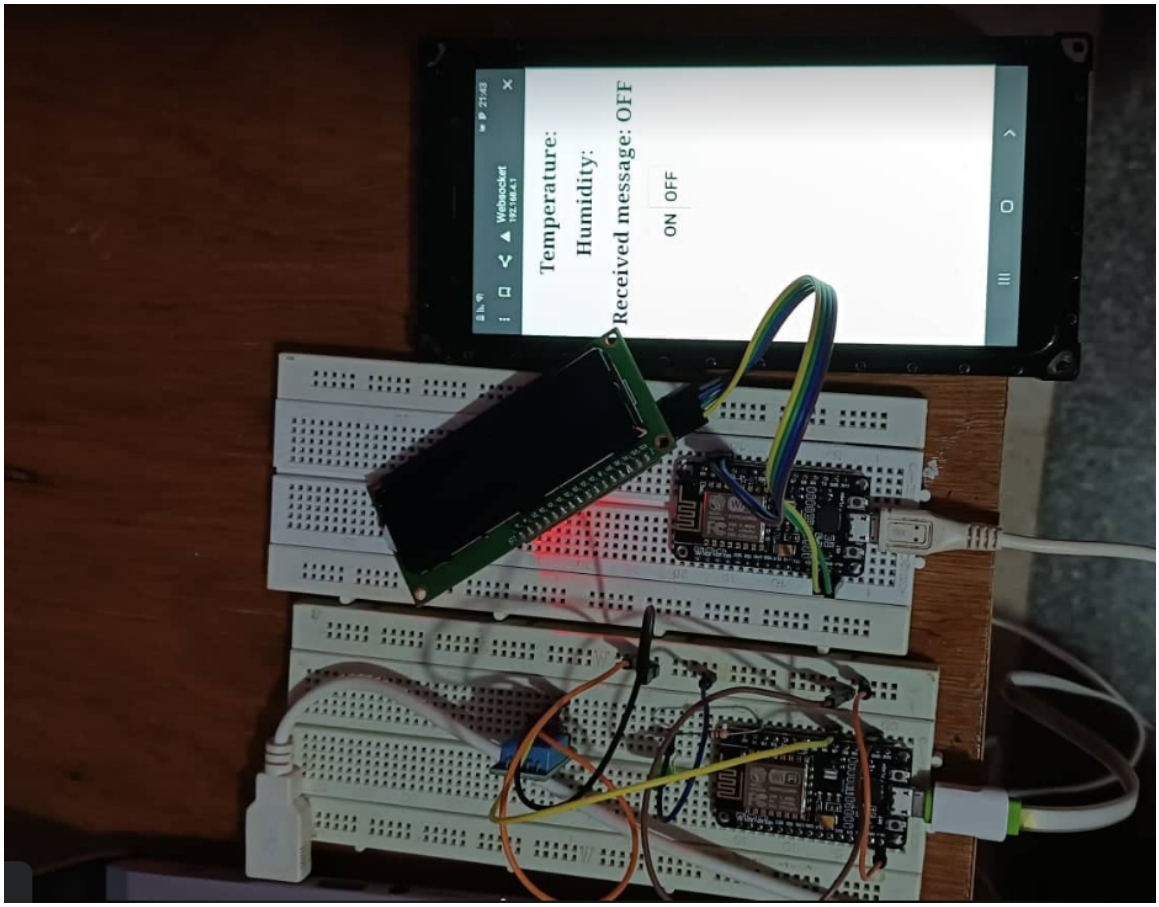
Ensuite,dans la fonction loop(), on garde le websocket actif en appelant la fonction WebSocket.loop()

```
void webSocketEvent(byte num, WStype_t type, uint8_t * payload, size_t length) {
  switch (type) {
    case WStype_DISCONNECTED: // enum that read status this is used for debugging.
      Serial.print("WS Type ");
      Serial.print(type);
      Serial.println(": DISCONNECTED");
      break;
    case WStype_CONNECTED:  // Check if a WebSocket client is connected or not
      Serial.print("WS Type ");
      Serial.print(type);
      Serial.println(": CONNECTED");
      if (digitalRead(5) == HIGH) {  //check if pin 22 is high or low
        pin_status = "ON";
        update_webpage(); // update the webpage accordingly
      }
      else {
        pin_status = "OFF"; //check if pin 22 is high or low
        update_webpage();// update the webpage accordingly
      }
      break;
    case WStype_TEXT: // check responce from client
      Serial.println(); // the payload variable stores teh status internally
      Serial.println(payload[0]);
      if (payload[0] == '1') {
        pin_status = "ON";
```

Une fois la JSON string désérialiser,on a déclaré trois variables pour contenir 3 paramètres envoyés par le serveur,LED STATUS,les données de température et humidité du capteur DHT11

Une fois qu' on a fait cela,on configure l’affichage sur le LCD display.

Résultat:



Maintenant ,Comme on a connectè via une connexion websocket.Si les données du capteur DHT ou l'état de LED changeant,les changements seront reflétés sur le téléphone,On peut également observer les changements sur LCD