Lab 4

Utilisation du modem WiFi sur Wemos D1

Dans ceTP nous allons explorer puis exploiter les fonctions du **modem WiFi** intégré dans la carte Wemos D1. Nous allons également utiliser un serveur **ThingSpeak** pour y envoyer , stocker (et recevoir) et les données captées par la carte.

4.1 Connexion WiFi et envoi des données sur le serveur ThingSpeak

Dans cet exemple les données du capteur DHT11 ou DHT22 sont envoyées sur le serveur **`ThingSpeak`**. Le serveur **`ThingSpeak`** peut être externe comme **`ThingSpeak.com`** ou interne installé sur une des cartes SBC disponibles dans notre laboratoire.

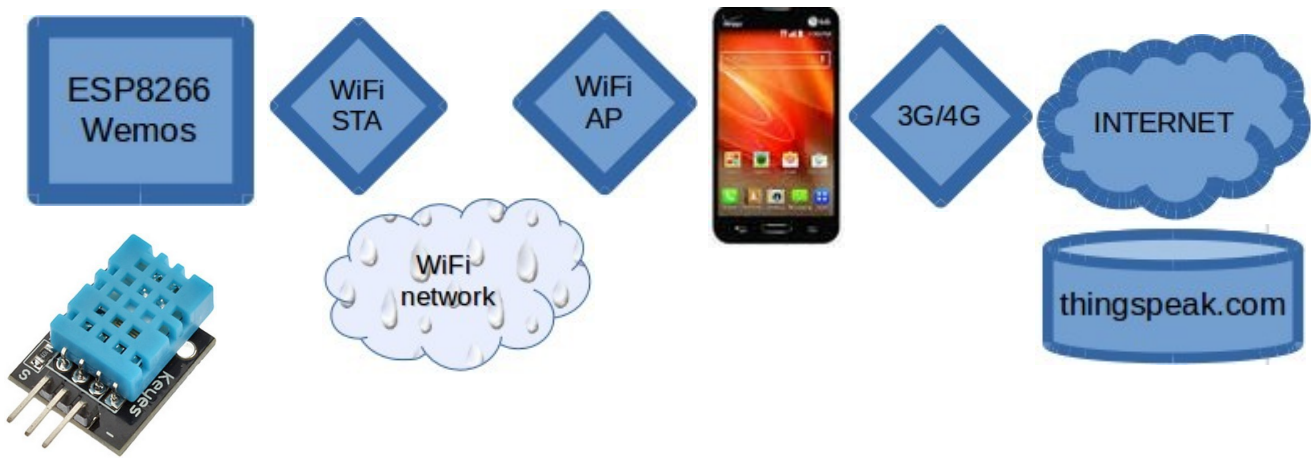

Figure 4.1. Une architecture IoT avec la carte Wemos D1, capteur DHT11/22 et une connexion WiFi/Internet avec le serveur **`ThingSpeak.com`**

Dans le cas illustré ci-dessus le serveur **`ThingSpeak.com`** est accessible via une connexion 3G/4G ouverte par le biais d'une application « softAP bridge » disponible sur les smartphones.
Une autre solution possible est de se connecter sur un serveur **ThingSpeak** installé sur une carte **SBC** type **RPI3** ou **Tinker Board** de ASUS avec un point d'accès type **softAP**..

```
#include <DHT.h>
#include <ESP8266WiFi.h>
String apiKey = "77QWEFDSMCU3TMGU";  // API write key
const char* ssid = "YotaPhoneAP";
const char* password = "admsmtr2017";
const char* server = "api.thingspeak.com";
// or IP address of local ThingSpeak server
#define DHTPIN D4 // wemos D1 shield with DHT11 sensor
#define DHTTYPE DHT11   // DHT 11 or  DHT 22
DHT dht(DHTPIN, DHTTYPE,11);
WiFiClient client;

void setup() {
 Serial.begin(9600);
 delay(100);
  dht.begin();
  WiFi.begin(ssid, password);
  Serial.println();
  Serial.print("Connecting to ");
  Serial.println(ssid);
```

```
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("");
  Serial.println("WiFi connected");
}

void loop() {
  float h = dht.readHumidity();
  float t = dht.readTemperature();
  if (isnan(h) || isnan(t)) {
    Serial.println("Failed to read from DHT sensor!");
    return;
  }
if (client.connect(server,80)) {
// "184.106.153.149" or api.thingspeak.com, 80 or private port number: 3000
    String postStr = apiKey;   // arguments to be sent to server
           postStr +="&field1=";
           postStr += String(t);
           postStr +="&field2=";
           postStr += String(h);
           postStr += "\r\n\r\n" ;
     client.print("POST /update HTTP/1.1\n");
     client.print("Host: api.thingspeak.com\n");  // modify if different server
     client.print("Connection: close\n");
     client.print("X-THINGSPEAKAPIKEY: "+apiKey+"\n");
     client.print("Content-Type: application/x-www-form-urlencoded\n");
     client.print("Content-Length: ");
     client.print(postStr.length());
     client.print("\n\n");
     client.print(postStr);
     Serial.print("Temperature: ");
     Serial.print(t);
     Serial.print(" degrees Celcius Humidity: ");
     Serial.print(h);
     Serial.println("% send to Thingspeak");
  }
  client.stop();
  Serial.println("Waiting...");
  // thingspeak needs minimum 15 sec delay between updates
  delay(30000);
}
```

Dans l'exemple présenté ci-dessus l'envoi des données est effectué explicitement par le biais d'une **page WEB** avec la commande **POST**. Cet opération nécessite la connaissance des entêtes HTML nécessaires pour porter les données vers le serveur WEB.

4.2 L'envoi des données vers un serveur ThingSpeak moyennant la librairie `ThingSpeak.h`

Dans l'exemple suivant les données captées par le capteur SHT30 sont envoyé vers le serveur ThingSpeak moyennant les fonctionnalités de la librairie **`ThingSpeak.h`**.
Grâce à ces fonctions nous pouvons envoyer et recevoir les données enregistrées sur le serveur ThingSpeak sans la préparation des pages WEB ; elles sont crées automatiquement par les fonctions disponibles.

```
#include <WEMOS_SHT3X.h>
#include <ESP8266WiFi.h>
#include "ThingSpeak.h"
const char *apiKey = "77QWEFDSMCU3TMGU";  // API write key
unsigned long chNum = 178529;
const char* ssid = "YotaPhoneAP";
const char* password = "admsmtr2017";
const char* server = "api.thingspeak.com";  // or IP of  local server
SHT3X sht30(0x45);
WiFiClient client;

void setup() {
  Serial.begin(9600);
  WiFi.begin(ssid, password);
  Serial.println();
  Serial.println();
  Serial.print("Connecting to ");
  Serial.println(ssid);
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500); Serial.print(".");
  }
  Serial.println("");
  Serial.println("WiFi connected");
  ThingSpeak.begin(client);
}

void loop() {
  sht30.get();
    ThingSpeak.setField(1,sht30.cTemp);
    ThingSpeak.setField(2,sht30.humidity);
    ThingSpeak.writeFields(chNum, apiKey);
     Serial.print("Temperature: ");
     Serial.print(sht30.cTemp);
     Serial.print(" degrees Celcius Humidity: ");
     Serial.print(sht30.humidity);
     Serial.println("% send to Thingspeak");
  Serial.println("Waiting...");
  delay(30000);
}
```

4.3 Réception des données à partir d'un serveur `ThingSpeak` et leur affichage

Dans cet exercice nous recevons les données du serveur ThingSpeak et nous les affichons soit sur un écran OLED connecté sur le bus I2C soit sur un écran couleur TFT connecté sur le bus SPI.

4.3.1 L'affichage sur l'écran OLED (SSD1306)

```
#include <ESP8266WiFi.h>
#include <Wire.h>
#include "OLED.h"
#include "ThingSpeak.h"
const char *apiKey = "77QWEFDSMCU3TMGU";
unsigned long chNum = 178529;
const char* ssid = "YotaPhoneAP";
const char* password = "admsmtr2017";
  // for server different than thingspeak.com modify the content of ThingSpeak.h
  // #define THINGSPEAK_URL "api.thingspeak.com"
  // SMTR #define THINGSPEAK_IPADDRESS IPAddress(172,19,65,65)
  // #define THINGSPEAK_PORT_NUMBER 80
  // SMTR #define THINGSPEAK_PORT_NUMBER 3000

WiFiClient client;
OLED display(D2, D1);  // SDA - D2, CL - D1 : wemos d1

void setup() {
  Serial.begin(9600);
  delay(10);
  WiFi.begin(ssid, password);
  Serial.println();
  Serial.println();
  Serial.print("Connecting to ");
  Serial.println(ssid);
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("");
  Serial.println("WiFi connected");
  display.begin();
  ThingSpeak.begin(client);
  display.print("Waiting ..");
}

char buffer[32];

void loop() {
     Serial.print("Temperature: ");
     Serial.print(String(t));
     Serial.print(" degrees Celcius Humidity: ");
     Serial.println(String(h));
```

```
    Serial.println("from Thingspeak");
    display.clear();
    display.print("Received values:");
    sprintf(buffer,"Temper.: %2.2f",t);
    display.print(buffer,2);
    sprintf(buffer,"Humidity: %2.2f",h);
    display.print(buffer,4);
  Serial.println("Waiting...");
  // thingspeak needs minimum 15 sec delay between updates
  delay(20000);
}
```

4.3.2 L'affichage sur l'écran TFT (ILI9341)

Dans cet exemple nous utilisons l'écran TFT – ILI9341 qui permet d'afficher les données et les dessins en couleur avec la résolution de 320x240 pixels (QVGA).
L'afficheur est connecté à la carte Wemos D1 par un bus SPI.

```
#include "SPI.h"
#include "Adafruit_GFX.h"
#include "Adafruit_ILI9341.h"
#include <ESP8266WiFi.h>
#include <WiFiManager.h>
#include "ThingSpeak.h"
 WiFiManager wifiManager;
#define TFT_DC D4
#define TFT_CS D8  // Wemos
Adafruit_ILI9341 tft = Adafruit_ILI9341(TFT_CS, TFT_DC);
uint16_t channel_color[] = {
  ILI9341_RED,         /* 255,   0,   0 */
  ILI9341_ORANGE,      /* 255, 165,   0 */
  ILI9341_YELLOW,      /* 255, 255,   0 */
  ILI9341_GREEN,       /*   0, 255,   0 */
  ILI9341_CYAN,        /*   0, 255, 255 */
  ILI9341_MAGENTA,     /* 255,   0, 255 */
  ILI9341_RED,         /* 255,   0,   0 */
  ILI9341_ORANGE,      /* 255, 165,   0 */
  ILI9341_YELLOW,      /* 255, 255,   0 */
  ILI9341_GREEN,       /*   0, 255,   0 */
  ILI9341_CYAN,        /*   0, 255, 255 */
  ILI9341_MAGENTA,     /* 255,   0, 255 */
  ILI9341_RED,         /* 255,   0,   0 */
  ILI9341_ORANGE,      /* 255, 165,   0 */
  ILI9341_BLACK        /*   0,   0,   0 */
};
const char *apiKey = "77QWEFDSMCU3TMGU";  // put your API write key
unsigned long chNum = 178529;
const char* ssid = "YotaPhoneAP";
const char* password = "admsmtr2017";
const char* server = "api.thingspeak.com";

void initbanner()
{
tft.setRotation(3);
  tft.fillScreen(ILI9341_BLACK);
  tft.setTextSize(6);
  tft.setTextColor(ILI9341_WHITE, ILI9341_RED);
```

```
  tft.setCursor(0, 0);
  tft.print(" S ");
 delay(1000);
  tft.setTextColor(ILI9341_WHITE, ILI9341_YELLOW);
  tft.setCursor(70, 60);
  tft.print(" M ");
    delay(1000);
  tft.setTextColor(ILI9341_WHITE, ILI9341_BLUE);
  tft.setCursor(140, 120);
  tft.print(" T ");
    delay(1000);
  tft.setTextColor(ILI9341_WHITE, ILI9341_GREEN);
  tft.setCursor(210, 180);
  tft.print(" R ");
    delay(1000);
  tft.fillScreen(ILI9341_BLACK);
  tft.setTextSize(2);
  tft.setTextColor(ILI9341_WHITE, ILI9341_RED);
  tft.setCursor(0, 0);
  tft.print(" ETN-SMTR ");
  tft.setTextColor(ILI9341_WHITE, ILI9341_GREEN);
  tft.print(" from ThingSpeak");
}

WiFiClient client;

void setup() {
  Serial.begin(9600);
  delay(10);
  wifiManager.resetSettings();
  delay(1000);
  WiFi.begin(ssid, password);
  Serial.println();
  Serial.println();
  Serial.print("Connecting to ");
  Serial.println(ssid);
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("");
  Serial.println("WiFi connected");
  ThingSpeak.begin(client);
  // for server different than thingspeak.com you must modify the content of
ThingSpeak.h file
  // #define THINGSPEAK_URL "api.thingspeak.com"
  // #define THINGSPEAK_IPADDRESS IPAddress(184,106,153,149)
  // SMTR #define THINGSPEAK_IPADDRESS IPAddress(172,19,65,65)
  // #define THINGSPEAK_PORT_NUMBER 80
 tft.begin();
 delay(400);
 initbanner();
}
```

```
char buffer[32];
void loop() {
// uncomment if use real sensor

float t = ThingSpeak.readFloatField(chNum, 1);
delay(20000);
float h = ThingSpeak.readFloatField(chNum, 2);
     Serial.print("Temperature: ");
     Serial.print(String(t));
     Serial.print(" degrees Celcius Humidity: ");
     Serial.println(String(h));
     Serial.println("from Thingspeak");
     tft.setTextSize(2);  // default value
      tft.setCursor(12, 32);
      tft.setTextColor(ILI9341_BLACK, ILI9341_CYAN);
      tft.print("  Received values:");
      tft.setCursor(12, 64);
      tft.setTextColor(ILI9341_BLACK, ILI9341_YELLOW);
      sprintf(buffer,"  Temperature: %2.2f  ",t);
      tft.print(buffer);
      tft.setCursor(12, 96);
      tft.setTextColor(ILI9341_BLACK, ILI9341_BLUE);
      sprintf(buffer,"  Humidity: %2.2f  ",h);
      tft.print(buffer);
  Serial.println("Waiting...");
  // thingspeak needs minimum 15 sec delay between updates
  delay(20000);
}
```

4.4 Point d'accès et serveur WEB pour l'affichage des messages

Dans l'exemple suivant l'écran la carte Wemos D1 et l'écran TFT servent à recevoir et à afficher les messages envoyés à partir d'un smartphone. Le smartphone doit se connecter sur le point d'accès créé par l'application et utiliser le navigateur WEB sur l'adresse **`http://192.168.4.1/affichage`** pour demander la page d'envoi du message.

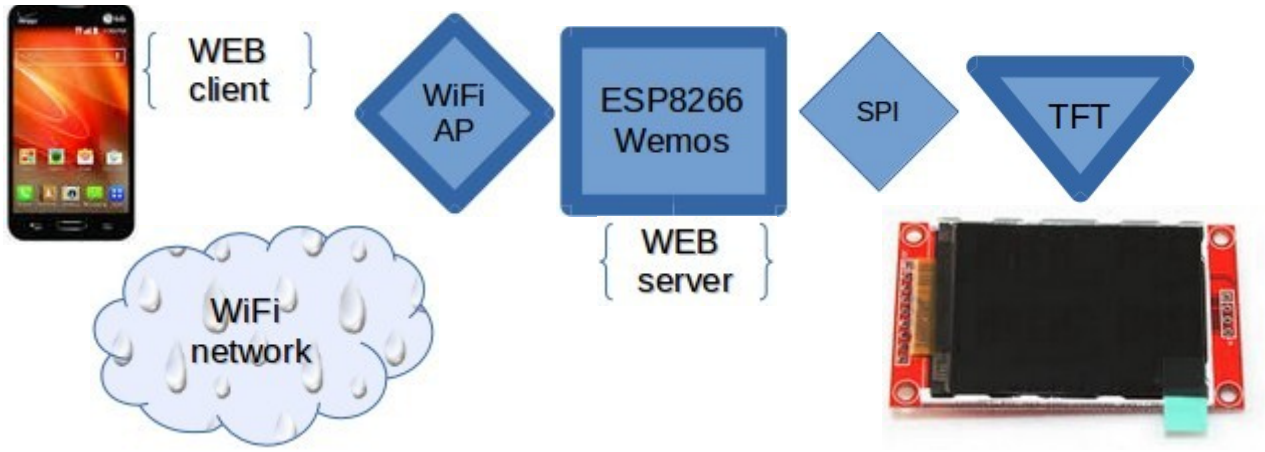

Figure 4.2. Une architecture IoT avec la carte Wemos D1 en mode AP - serveur WEB et un écran TFT pour afficher les messages envoyés par un client WEB.

```
#include <ESP8266WiFi.h>
#include <WiFiClient.h>
#include <ESP8266WebServer.h>
#include <WiFiManager.h>
#include "SPI.h"
#include "Adafruit_GFX.h"
#include "Adafruit_ILI9341.h"
#define TFT_DC D4  // Wemos D1
#define TFT_CS D8  // Wemos D1
WiFiManager wifiManager;
Adafruit_ILI9341 tft = Adafruit_ILI9341(TFT_CS, TFT_DC);
uint16_t channel_color[] = {
  ILI9341_RED,         /* 255,   0,   0 */
  ILI9341_ORANGE,      /* 255, 165,   0 */
  ILI9341_YELLOW,      /* 255, 255,   0 */
  ILI9341_GREEN,       /*   0, 255,   0 */
  ILI9341_CYAN,        /*   0, 255, 255 */
  ILI9341_MAGENTA,     /* 255,   0, 255 */
  ILI9341_RED,         /* 255,   0,   0 */
  ILI9341_ORANGE,      /* 255, 165,   0 */
  ILI9341_YELLOW,      /* 255, 255,   0 */
  ILI9341_GREEN,       /*   0, 255,   0 */
  ILI9341_CYAN,        /*   0, 255, 255 */
  ILI9341_MAGENTA,     /* 255,   0, 255 */
  ILI9341_RED,         /* 255,   0,   0 */
  ILI9341_ORANGE      /* 255, 165,   0 */
};
char sbuff[256];
const char *ssid = "AP.smtr10";
const char *password = "admsmtr";  // not used by initialization
```

```
const int channel=10;  // WiFi channel
ESP8266WebServer server(80);   // to launch the WEB server
char page[2048];
int c,i,s;

void affichage()
{
  memset(sbuff,0,256);// argument analysis and parameter setting
  for(int i=0; i<server.args() ;i++)
    {
    if(server.argName(i)=="submit") break;   // page refresh  no arguments
    if(server.argName(i)=="mess")   server.arg(i).toCharArray(sbuff,128);
    }
    snprintf ( page, 2048,
  "<html>\
  <head>\
    <meta name='viewport' content='width=device-width, user-scalable=no'>\
    <title>ESP8266 Box</title>\
    <style type='text/css'>\
     body { background-color:green; font-size: 12pt; font-family: Arial,
Helvetica, Sans-Serif; Color: black; }\
    </style>\
  </head>\
  <body>\
  <br>\
  <h2>ETN-SMTR : afficheur des messages</h2> \
     <hr>Ecrivez votre message:<br>\
   <form action='/affichage' method='get' target='_self'>\
   <textarea name='mess' rows=8 cols=26 >Votre message a
afficher</textarea><br>\
   <input type='submit' name='submit' value='envoyer'/>\
   </form>\
   <hr>\
   Message evoye:<br>%s\
   <hr>\
   </body>\
   </html>",sbuff);
   Serial.println(sbuff);
  tft.fillScreen(ILI9341_BLACK);
  tft.setTextSize(2);
  tft.setTextColor(ILI9341_WHITE, ILI9341_RED);
  tft.setCursor(0, 0);
  tft.print(" ETN-SMTR ");
  tft.setTextColor(ILI9341_WHITE, ILI9341_GREEN);
  tft.print(" WiFi ");
  tft.setTextColor(ILI9341_WHITE, ILI9341_BLUE);
  tft.print(" Afficheur");
  //tft.fillScreen(ILI9341_BLACK);
  tft.setCursor(0, 32);
  tft.print(sbuff);
```

```
server.send (200,"text/html", page );
}

void handleRoot() {
server.send(200, "text/html", "<h1>You are connected to SMTR display
device</h1>");
}

void setup() {
delay(1000);
  wifiManager.resetSettings();
  delay(1000);
  Serial.begin(9600);
  tft.begin();
Serial.print("Configuring access point...");
Serial.println(ssid);  Serial.println(password);
// we use channel parameter and no password  NULL
WiFi.softAP(ssid,NULL,channel);
IPAddress myIP = WiFi.softAPIP();
Serial.print("AP IP address: ");Serial.println(myIP);
server.on("/", handleRoot);
server.on("/affichage", affichage);
server.begin();
Serial.println("HTTP server started");
  delay(200);
tft.setRotation(3);
   // init banner
  tft.fillScreen(ILI9341_BLACK);
  tft.setTextSize(2);
  tft.setTextColor(ILI9341_WHITE, ILI9341_RED);
  tft.setCursor(0, 0);
  tft.print(" ETN-SMTR ");
  tft.setTextColor(ILI9341_WHITE, ILI9341_GREEN);
  tft.print(" WiFi ");
  tft.setTextColor(ILI9341_WHITE, ILI9341_BLUE);
  tft.print(" Afficheur");
}

void loop() {
server.handleClient();
}
```

4.5 Point d'accès et serveur WEB pour l'affichage des donnés captées

L'exemple suivent montre comment consulter les données captées par un capteur de Température et Humidité sur un smartphone connecté au point d'accès WiFi activé sur la carte Wemos D1.

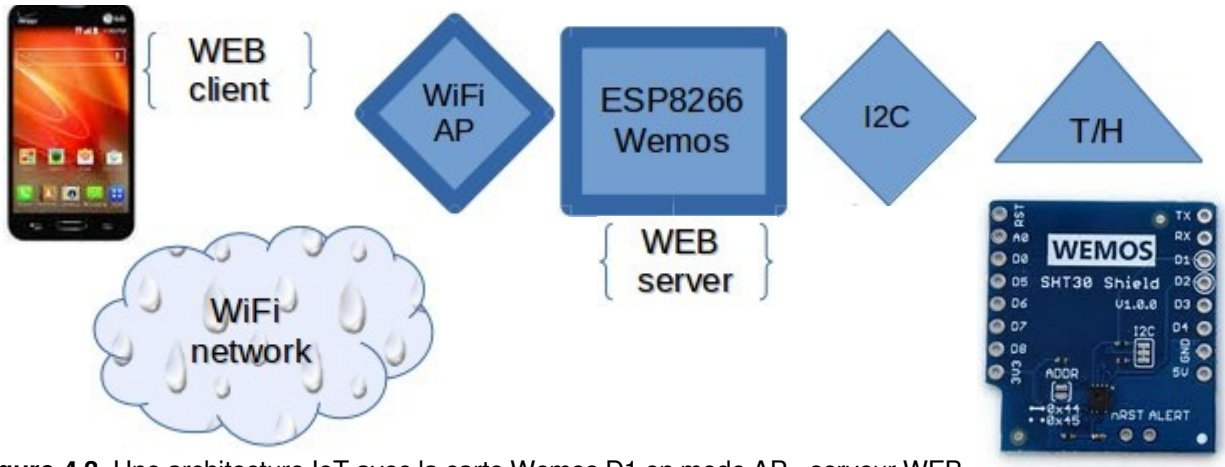

Figure 4.3. Une architecture IoT avec la carte Wemos D1 en mode AP - serveur WEB permettant envoyer les réponses aux requêtes HTML de client avec les données captés sur la carte.

```
#include <ESP8266WiFi.h>
#include <WiFiClient.h>
#include <ESP8266WebServer.h>
#include <WiFiManager.h>
#include <WEMOS_SHT3X.h>
SHT3X sht30(0x45);
WiFiManager wifiManager;
const char *ssid = "SCLab12";
const char *password = NULL;
int channel=12 ;
ESP8266WebServer server(80); // default IP : 192.168.4.1

void handleRoot() {
      server.send(200, "text/html", "<h1>You are connected</h1>");
}

void getsht30() {
  char buffer[512];
  sht30.get();
  sprintf(buffer,"<h1>Data from SHT30</h1><hr><br>Temperature:%2.2f<br>Humidity:
%2.2f<br><hr>",sht30.cTemp,sht30.humidity);
  server.send(200, "text/html", buffer);
}

void setup() {
      Serial.begin(9600);
      Serial.println();
      wifiManager.resetSettings();
      delay(1000);
      Serial.print("Configuring access point...");
      WiFi.softAP(ssid , NULL ,channel);
```

```
      IPAddress myIP = WiFi.softAPIP();
      Serial.print("AP IP address: ");
      Serial.println(myIP);
      server.on("/", handleRoot);
      server.on("/sht30", getsht30);
      server.begin();
      Serial.println("HTTP server started");
}

void loop() {
      server.handleClient();
}
```

Exercices:

L'envoi de données GPS sur le serveur ThingSpeak.
Affichage des données GPS sur l'écran TFT.
Récéption des données GSP à partir du serveur ThingSpeak et leur affichage sur l'écran TFT.
Mise en ouvre d'un point d'accès et d'un serveur WEB permettant de recevoir les données GPS de la carte.