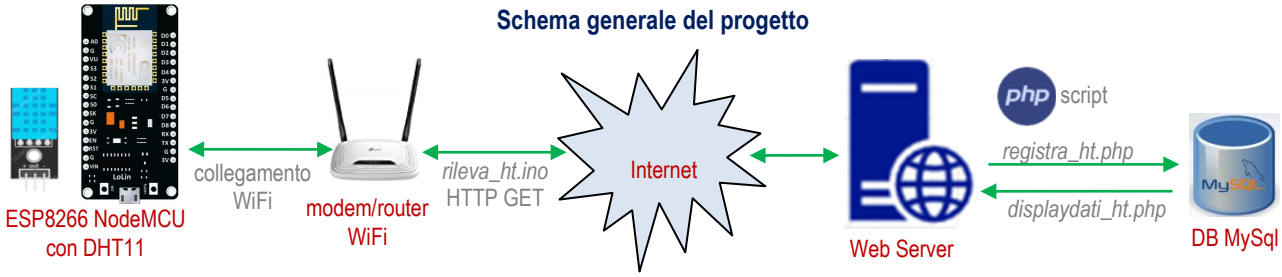
ESP8266 e DHT11: monitoraggio di umidità e temperatura e invio dei dati ad un DB MySql remoto

Esempio applicativo di Internet of Things (IoT).

Il progetto realizza una stazione di monitoraggio dell'umidità e della temperatura basata su una scheda ESP8266 NodeMCU, interfacciata con il sensore DHT11. La scheda, collegata ad Internet tramite il modulo WiFi integrato, invia i dati rilevati ad un database MySql remoto.



- La scheda, ad intervalli regolari di tempo (ad esempio ogni 15 minuti), tramite il sensore DHT11 rileva l'umidità H e la temperatura T ed invia i dati, con un richiesta HTTP GET, ad uno script PHP del server remoto (applicazione Web Client Arduino Ide *'rileva_ht.ino'*)
- Sul server, lo script PHP destinatario della richiesta recupera i valori inviati da ESP8266 NodeMCU e li inserisce in un database MySql remoto (*'registra_ht.php'*)
- Una pagina web sul server remoto, accessibile da qualsiasi parte del mondo, legge i dati memorizzati nel database e li presenta sotto forma di tabelle e grafici (*'displaydati_ht.php'*)

Ho eseguito il test del progetto sia con un server locale WAMP che con un server remoto e nei codici di programmazione, che fanno riferimento al test con WAMP, ho inserito i commenti che spiegano come utilizzare il server remoto.

Con il server WAMP ho utilizzato:

a) un database MySql amministrato con phpMyAdmin
b) lo script 192.168.1.3/prove/registra_ht.php (192.168.1.3 è l'IP del mio Pc con Wamp Server)
c) la pagina web 192.168.1.3/prove/displaydati_ht.php

Con il server remoto (dominio www.maurodeberardis.it) ho utilizzato:

a) un database MySql
b) lo script www.maurodeberardis.it/ CodiciPHP /registra_ht.php
c) la pagina web www.maurodeberardis.it/CodiciPHP/displaydati_ht.php

Componenti necessari

- Scheda ESP8266 NodeMCU collegata ad un PC Windows X con cavo micro USB
- Breadboard con cavetteria
- Sensore DHT11
- 2 Diodi led e 2 Resistenze di limitazione 220 Ohm

Sensore DHT11

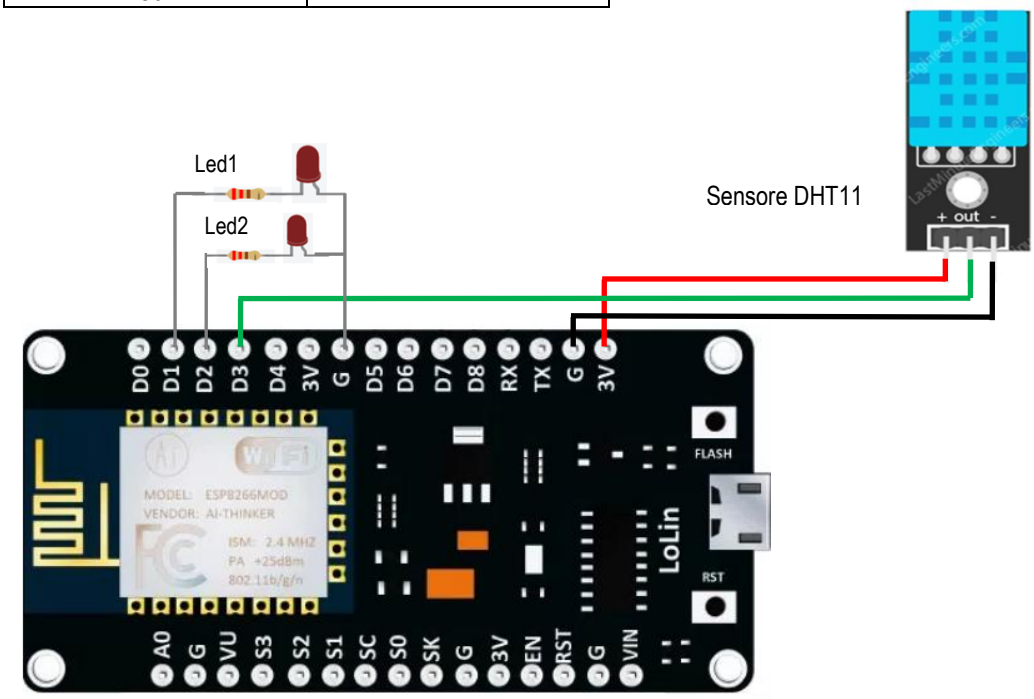
Il DHT11 è un sensore digitale di umidità e temperatura dell'aria costituito da una parte resistiva che si occupa della rilevazione dell'umidità e da un NTC che rileva la temperatura. Il sensore viene fabbricato in due configurazioni: a 3 o a 4 pin. In entrambi i casi i pin VCC, GND e OUT (DATA), che devono essere collegati ad ESP8266 NodeMCU, sono indicati chiaramente. Il pin NC, nel caso della configurazione a 4 pin, non viene collegato. Tra la linea del segnale OUT e VCC (3V) è necessaria una resistenza di pull-up da 4.7K Ohm.
Nel caso di configurazione a 3 pin, utilizzata in questo progetto, la resistenza di pull-up è già montata.

Schema dei collegamenti

Lo schema è molto semplice. Il diodo led sul pin D1 serve ad indicare che il collegamento WiFi è attivo, il diodo led sul pin D2 si accende per qualche secondo quando la scheda legge i valori di umidità e temperatura.

Sensore DHT11	ESP8266 NodeMCU
VCC (+)	3V
DATA (out)	D3
GND(-)	G

Diodi Led	ESP8266 NodeMCU
Led 1	D1
Led 2	D2



Scheda
ESP8266 Node MCU

Applicazione Web Client Arduino Ide 'rileva_ht.ino'

Utilizzo dell'IDE di Arduino per programmare ESP8266 NodeMCU

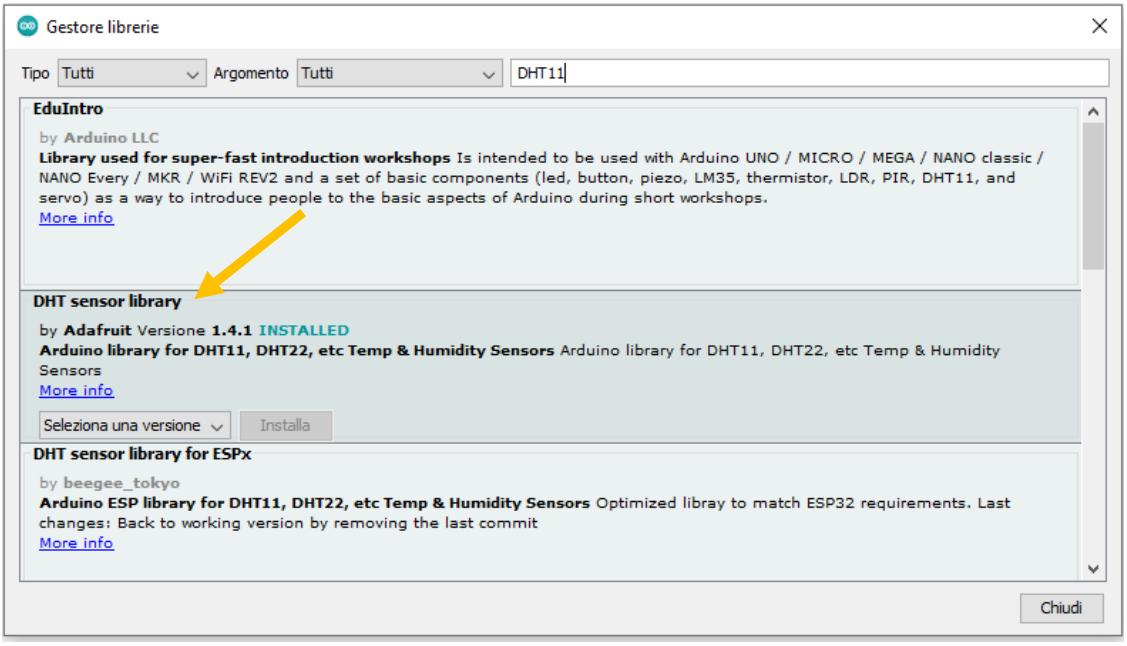
Chi legge questo tutorial, probabilmente è in grado di programmare la scheda ESP8266 NodeMCU con l'IDE di Arduino. Se non è cosi, può far riferimento al progetto *'Web Server ESP8266 NodeMCU'* o al progetto *' ESP8266, RC522 e MySql: Sistema di validazione remota di un tag RFID'* pubblicati nella sezione "Arduino" dell'Area Download del sito www.maurodeberardis.it

Programmiamo la scheda ESP8266 utilizzando l'IDE di Arduino: ricordo che è necessario che il componente aggiuntivo ESP8266 sia installato nell'IDE di Arduino e che sia selezionata la porta giusta.

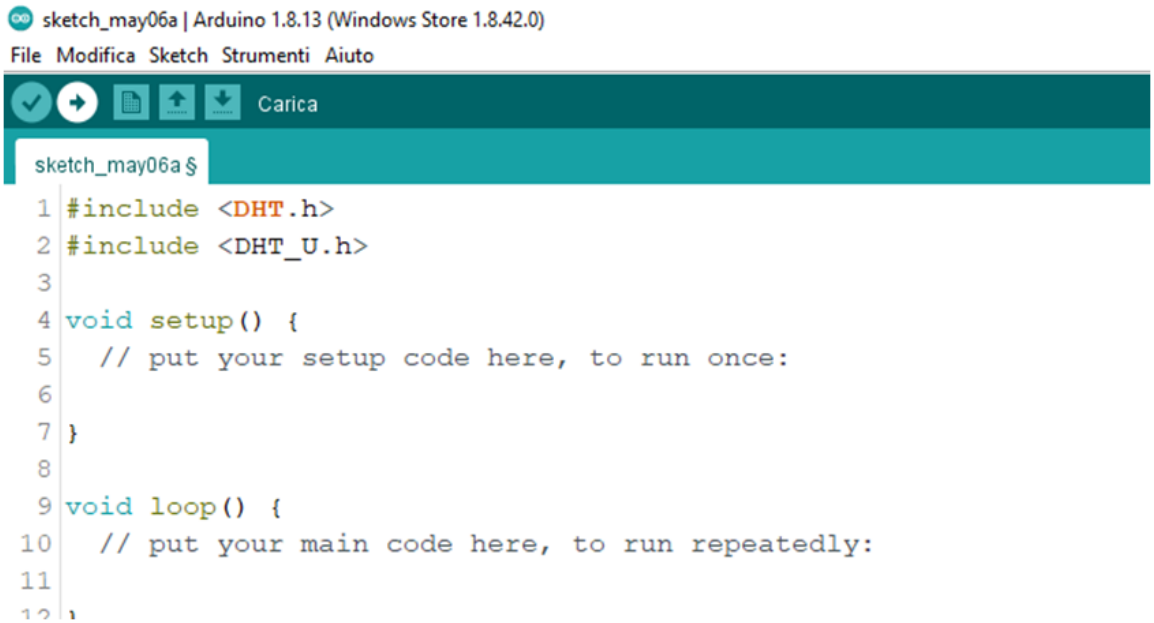
Prima di scrivere il codice, bisogna installare, e quindi includere nello sketch, la libreria necessaria per poter utilizzare il sensore DHT11

- Con l'IDE Arduino in esecuzione, scegliere Strumenti-Gestione librerie…

- Quando si apre la finestra del Gestore librerie, nel campo di ricerca inserire "DHT11" e tra le diverse librerie proposte trovare, selezionare e installare "DHT sensor library" (by Adafruit)



- Aprire lo sketch che si vuole programmare, scegliere l'opzione *Sketch-#include libreria* e selezionare "DHT Sensor Library" e nello sketch vengono incluse due librerie (DHT_U.h non è necessaria e se si vuole si può cancellare)



Codice dello sketch sull'ESP8266 NodeMCU: rileva_ht.ino

```
/* Prof. Mauro De Berardis 2021
 * ------------------------------------------------------------------------------------
 * Il progetto realizza una stazione di monitoraggio basata su una scheda ESP8266 NodeMCU
 * interfacciata con un sensore  DHT11. La scheda, collegata ad Internet tramite il modulo
 * WiFi integrato al suo interno,  ad intervalli regolari di tempo rileva  umidità H
 * e temperatura T  e invia i valori letti ad un database MySql remoto.
 */
#include <ESP8266WiFi.h> //permette la connessione della scheda  alla rete WiFi
#include <ESP8266HTTPClient.h>

#include <DHT.h>      //libreria necessaria per utilizzare il sensore DHT11
#define sensorePin D3
// pin di ESP8266 NodeMCU collegato all'uscita (pin OUT) del sensore DHT11
#define tipoDHT DHT11
/* definisce il tipo di sensore della famiglia DHT:  in questo caso viene selezionato
   DHT11, ma esistono sono altri sensori quali DHT21 e DHT22 */
DHT dht(sensorePin,tipoDHT);  //Istanzia l'oggetto dht della classe DHT
float h;          //umidità
float t;          //temperatura
#define Led1 D1       // Led verde
#define Led2 D2       // Led rosso
//***Configurazione WiFI: impostazione delle credenziali di rete
const char* ssid = "mySSID";         //Inserire qui l'SSID
const char* password = "myPassword"; //Inserire qui la password WiFi
// definisco le variabili per gestire l'intervallo tra una rilevazione e l'altra
unsigned long time_last = 0;        // time ultima rilevazione
unsigned long time_current;         // time corrente
const long interval = 900000;
// fisso l'intervallo tra una rilevazione e l'altra ad esempio a 15 minuti
// 900000ms = 9000s=15 minuti


void setup()
{
  delay(1000);

  pinMode(Led1,OUTPUT);
  pinMode(Led2,OUTPUT);

  // apre una connessione seriale. A scopo di debug inviamo messaggi al monitor seriale
  Serial.begin(115200);
  WiFi.begin(ssid, password);   //Si connette al router WiFi
  Serial.println("Connessione al WiFi in corso..");
  while (WiFi.status() != WL_CONNECTED)
  {
      delay(500);
      Serial.print(".");
  }
  Serial.println("");
  Serial.print("ESP8266 NodeMCU connessa alla rete WiFI: ");
  Serial.print(ssid);
  Serial.print("  con il numero IP : ");
  Serial.println(WiFi.localIP());//l'IP di ESP8266 viene assegnato dal DHCP
  Serial.println();
  digitalWrite(Led1, 1); // si accende il led 1 che indica che il Wifi è attivo
  delay(500);
  rileva_invia();
  // chiamata alla funzione che rileva umidità H e temperatura T  e invia i dati allo script
  // PHP  del server remoto con una richiesta HTTP GET

} //----chiude setup()-------------------------------------------------------------------
```

```
void loop()
{
  time_current = millis();
  if (time_current - time_last >= interval) // se è trascorso il tempo impostato in interval
  {
    time_last = time_current; // aggiorno il time ultima rilevazione
    rileva_invia();
    // chiamata alla funzione che rileva umidità H e temperatura T  e invia i dati allo script
    // PHP  del server remoto con una richiesta HTTP GET

    // il led 2 si accende per 2 secondi  segnalando che è stata fatta una lettura
      digitalWrite(Led2, 1);
      delay(2000);
      digitalWrite(Led2, 0);
  }
}//----chiude loop()----------------------------------------------------------------

void rileva_invia()
{
   // faccio 3 letture in modo che i valori letti dal sensore DHT11 , che non è velocissimo,
   // si stabilizzino
   for(int i=0;i<3;i++)
    {
        h = dht.readHumidity();     // Lettura dell'umidità
        t = dht.readTemperature();  // Lettura della temperatura in gradi Celsius
        delay(300);
    }
    if (isnan(h) || isnan(t))  //Verifica se  si presenta un errore di lettura
    {
          Serial.println("Errore di lettura...");
          h=0.0;
          t=0.0;
    }
    String s="Umidità= "+String(h)+"     Temperatura="+String(t);
    Serial.println(s);
    String s2=String(h)+"#"+String(t)+"#";
    HTTPClient http;    //Dichiara l'oggetto http della classe HTTPClient
    String  richiestaHTTP, link,risposta;
    richiestaHTTP = "?h=" + String(h)+"&t="+String(t);
    // querystring da inviare nell'URL della richiesta HTTP GET

    // utilizzo un server di hosting remoto ad esempio www.maurodeberardis.it
    // link= "http://www.maurodeberardis.it/CodiciPHP/registra_ht.php"+richiestaHTTP;

    // utilizzo un server WAMP su un pc con IP 192.168.1.3. Al posto del numero Ip non posso
    // scrivere localhost
    link = "http://192.168.1.3/prove/registra_ht.php"+richiestaHTTP;
    http.begin(link);           //Specifica la destinazione della richiesta HTTP GET
    int httpCode = http.GET();  //Invia la richiesta HTTP GET

    if (httpCode==HTTP_CODE_OK)
      {
          risposta= http.getString(); //Ottiene la risposta dal Server
          Serial.print("HTTP code: ");
          Serial.println(httpCode);
          Serial.println("Dati salvati correttamente sul DB remoto");
      }
      else
      {
          Serial.println("Errore");
          Serial.print("HTTP code: ");
          Serial.println(httpCode);
          Serial.println(link);
          Serial.println();
      }

}//----chiude rileva_invia()--------------------------------------------------------
```

Script registra_ht.php

Lo script php lato server recupera i dati inviati da ESP8266 NodeMCU lato client con una richiesta HTTP GET ed esegue una query di inserimento nella *tabella "misure_ht"* del *database "stazioni":* il database viene utilizzato ovviamente per memorizzare le letture del sensore, in modo che sia possibile accedervi in un secondo momento. È un database elementare con una sola tabella che memorizza un id, la data e l'ora, il valore dell'umidità H e il valore della temperatura T.

Query di creazione del database MySql "stazioni" e della tabella "misure_ht"

```
CREATE DATABASE stazioni;

CREATE TABLE IF NOT EXISTS misure_ht (
id INT(10) PRIMARY KEY AUTO_INCREMENT,
dataora datetime,
umidita double(8,4),
temperatura double(8,4))
```

#	Nome	Tipo	Codifica caratteri	Attributi	Null	Predefinito	Extra
1	**id**	int(10)			No	*Nessuno*	AUTO_INCREMENT
2	**dataora**	datetime			Sì	*NULL*	
3	**umidita**	double(8,4)			Sì	*NULL*	
4	**temperatura**	double(8,4)			Sì	*NULL*	

Codice dello script: http://192.168.1.3/prove/registra_ht.php

```
<?php
if(isset($_GET['h']))
  {
    $sh=$_GET['h'];
    $st=$_GET['t'];
    $h=floatval($sh);
    $t=floatval($st);
    $ora = date("Y-m-d H:i:s");
    // $conn=mysqli_connect("myhost","myusername","mypassword","stazioni_ht");
    // stringa di connessione
    $conn=mysqli_connect("localhost","myusername","mypassword","stazioni");
    $s="Insert into misure_ht (dataora,umidita,temperatura)
        values ('$ora',$h,$t)";
    $q=mysqli_query($conn,$s);
    mysqli_close($conn);
  }
?>
```

Script displaydati_ht.php

Lo script visualizza i dati memorizzati nella tabella "misure_ht" del database "stazioni". La pagina, volutamente molto semplice ed essenziale, stampa in una tabella le misure eventualmente filtrate per data.

```
<!DOCTYPE html>
<html>
<head><title>ESP8266-DHT11-MySql </title></head>
<style> td,th{border:1px solid #aaaaaa;text-align:center}</style>
<body>
<?php
 if(!isset($_GET["submit"]))
 {  ?>
   <form name="form1">
        <h2>ESP8266 e DHT11: monitoraggio di umidità e temperatura</h2>
        <label>Data (data vuota: tutte le letture)</label><br/>
        <input type="date" id="data_let" name="data_let"> <br/><br/>
        <input type="submit" name="submit" value="Display dati">
        <input type="reset" value="Reset">
    </form>
<?php
}
else
{
   $data_let= $_GET['data_let'];
   $aa=substr($data_let,0,4);
   $mm=substr($data_let,5,2);
   $gg=substr($data_let,8,2);
   $data=$gg."/".$mm."/".$aa;
   $aa=(int)$aa; $mm=(int)$mm;    $gg=(int)$gg;
   $titolo1="ESP8266 e DHT11: monitoraggio di umidità e temperatura";
   $titolo2="Data rilevamento: ".$data;
   if($aa=="")$titolo2="Data rilevamento: ";
   $conn=mysqli_connect("myhost","myusername","mypassword","stazioni");
   //    se utilizzo Wamp
    //   $conn=mysqli_connect("192.18.1.3","myusername","mypassword","rfid");
    //   192.18.1.3 è l'indirizzo IP del Server Wamp utilizzato*/
   if($aa==""){
        $s="Select * from misure_ht order by dataora Desc";
   }
   else
   {
        $s="Select * from misure_ht where
        Year(dataora)=$aa And Month(dataora)=$mm And Day(dataora)=$gg
        order by dataora Desc";
   }

   $q=mysqli_query($conn,$s);
   $nr=mysqli_num_rows($q);
   if ($nr==0){
        echo "<h2>$titolo1 </h2>";
        echo "<br><br>Nessun dato trovato<br/><br/>";
   }
   else
   {
        echo "<h2>$titolo1 </h2>";
        echo "<h3>$titolo2</h3>";
        echo "<table width=80%>";
        echo "<tr><th>Nr</th><th>Id</th>
        <th>Data-ora lettura</th><th>Umidit&agrave;</th>
        <th>Temperatura</th>  </tr>";
```

```
        while($r=mysqli_fetch_array($q))
        {
                $timemisura=substr($r[1],11,8);
                if($aa=="")$timemisura=$r[1];
                echo "<tr><td>$k</td><td>$r[0]</td>
                <td>$timemisura</td><td>$r[2]</td><td>$r[3]</td>";
                echo"</tr>";
        }
        echo"</table>";
        }
        mysqli_close($conn);
        echo"<br/><br/><a href='displaydati_ht.php'>Scegli un'altra data</a>";
}
?>
</body>
</html>
```

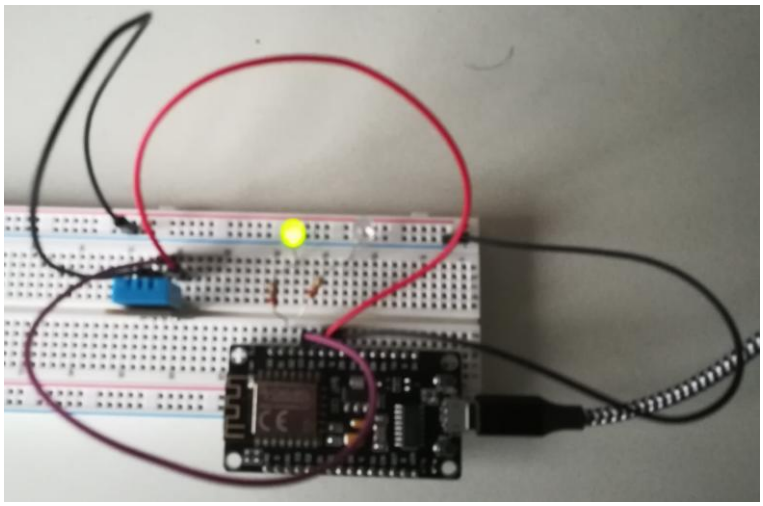
Prova del progetto

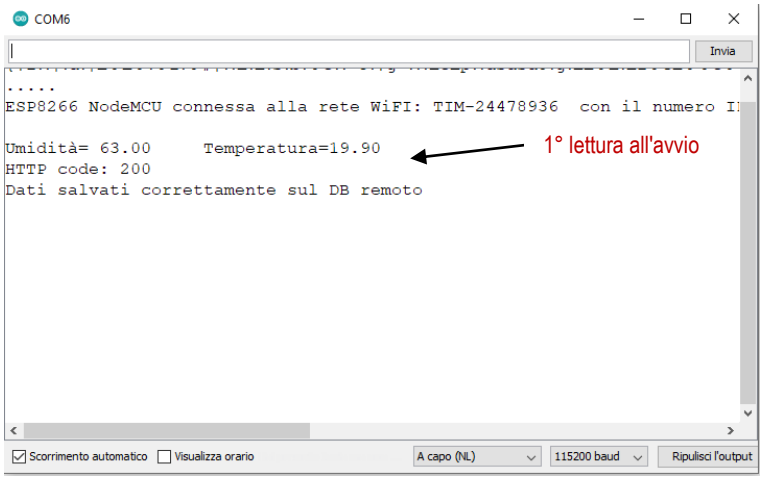
Il progetto è perfettamente funzionante sia con server di hosting remoto (www.maurodeberardis.it) sia con Wamp Server (192.168.1.3): qui vengono riportate le **prove fatte con Wamp Server.**

Rilevazione H e T e invio dei dati al DB MySql remoto

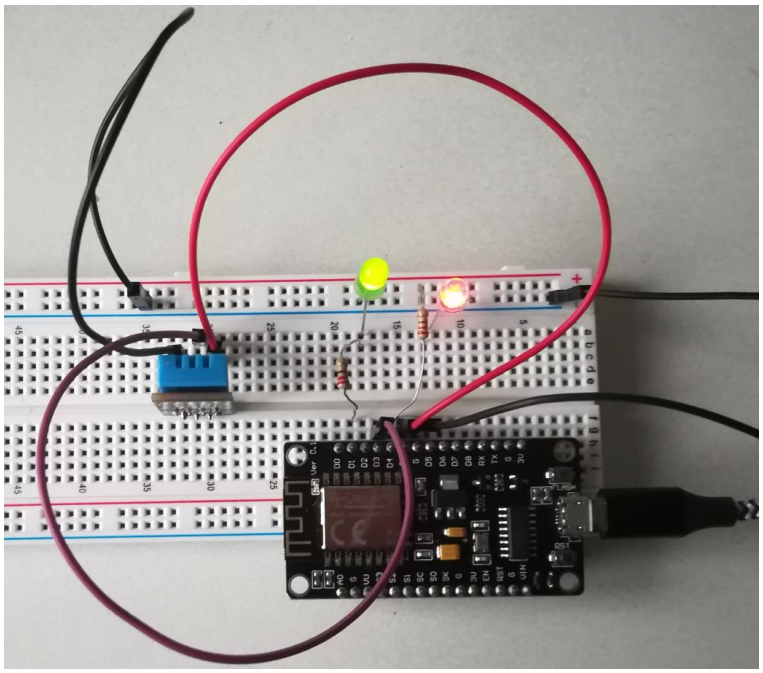
Resetto la scheda ESP8266 NodeMCU (premendo l'apposito pulsantino) e sull'IDE Arduino apro il monitor seriale (la velocità deve essere impostata a 115200 baud).

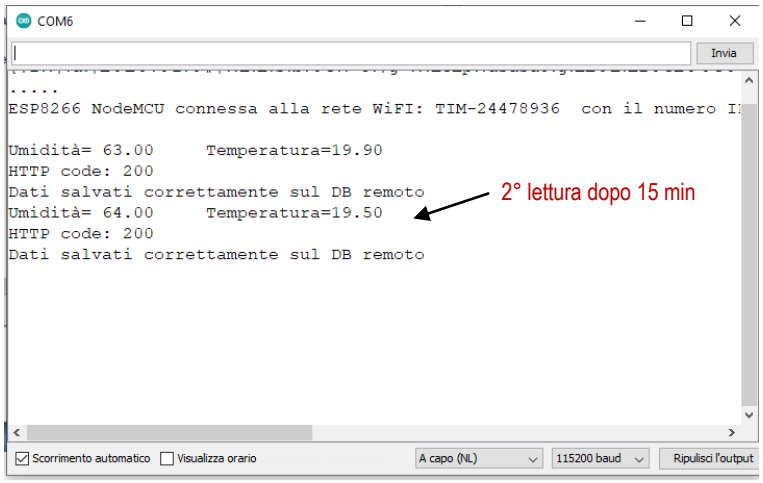
- La scheda si connette alla rete Wifi e il diodo led giallo si accende
- Effettua una lettura iniziale e invia i dati rilevati al database MySql remoto
- Resta quindi in attesa di fare una lettura ogni 15 minuti





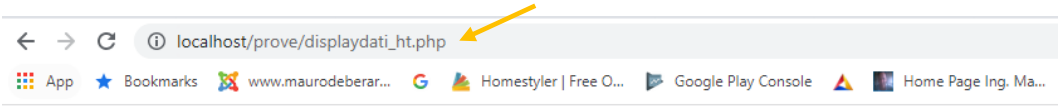
Trascorsi 15 minuti, la scheda effettua una seconda lettura con conseguente invio dei dati rilevati al database MySql remoto. La lettura viene segnalata dal diodo led rosso che resta acceso per due secondi





Le due letture effettuate risultano memorizzate nella tabella "misure_ht" del db MySql "stazioni"

Visualizzazione dei dati



localhost/prove/displaydati_ht.php?data_let=2021-11-16&submit=Display+dati

ESP8266 e DHT11: monitoraggio di umidità e temperatura

Data rilevamento: 16/11/2021

Nr	Id	Data-ora lettura	Umidità	Temperatura
1	36	16:31:40	64.0000	19.6000
2	35	16:16:40	64.0000	19.5000
3	34	16:01:44	63.0000	19.9000
4	33	15:47:57	63.0000	19.9000
5	32	15:32:57	63.0000	19.7000
6	31	15:17:57	66.0000	19.1000
7	30	15:02:57	68.0000	18.6000
8	29	14:47:57	68.0000	18.6000
9	28	14:33:11	67.0000	18.7000
10	27	14:17:57	67.0000	18.7000
11	26	14:02:57	67.0000	18.8000
12	25	13:47:57	67.0000	18.9000
13	24	13:32:57	68.0000	18.7000
14	23	13:17:57	67.0000	18.8000
15	22	13:02:58	67.0000	18.8000
16	21	12:47:58	66.0000	19.0000
17	20	12:32:56	66.0000	19.2000
18	19	12:17:56	64.0000	19.6000
19	18	12:02:55	62.0000	20.0000
20	17	11:47:55	64.0000	19.8000
21	16	11:32:55	62.0000	20.3000

Scegli un'altra data

Lascio agli studenti più volenterosi e motivati il compito di migliorare il progetto, specialmente per quel che riguarda la presentazione dei dati.

Suggerisco inoltre l'idea di espandere il progetto ad una rete di stazioni aggiungendo nella tabella "misure_ht" un campo "numerostazione" oppure utilizzando una tabella per ciascuna stazione: misure_ht1, misure_ht2, ecc.

Buon lavoro