	Istituto Tecnico Industriale Statale "C. Zuccante" Venezia-Mestre	Allievo: Lorenzo De Luca Data: 8/5/2019 Classe: 4ID

LABORATORIO DI SISTEMI E RETI

Titolo esperienza: Stazioni meteo remote con server per archiviare/mostrare i risultati

IDE utilizzati:

ARDUINO → Arduino 1.8.8
Server Java → Apache Netbeans 11
Sito WEB→ Visual Studio Code

LINGUAGGI DI PROGRAMMAZIONE UTILIZZATI:

Java → Server java
Arduino/c++ → Stazione meteo(Arduino)
PHP→ sito web per mostrare i risultati(HTML, CSS, JavaScript)

COMPONENTI UTILIZZATI(indicazioni generiche di montaggio, vedere sotto schemi Fritzing):

Arduino UNO R3,
Shield Ethernet per Arduino,
Display LCD(2*16) → adattatore I2C Arduino
Fotoresistore(3V + resistenza 10K) → pin A0 Arduino
Termoresistore(3V) → pin A1 Arduino
Barometro(3V)→ Arduino I2C
Sensore di umidità(3V) → pin A2 Arduino
Dispositivo host per il server java
Server web(Apache 2 + PHP 7) per il sito web(interfaccia grafica) → XAMPP portable

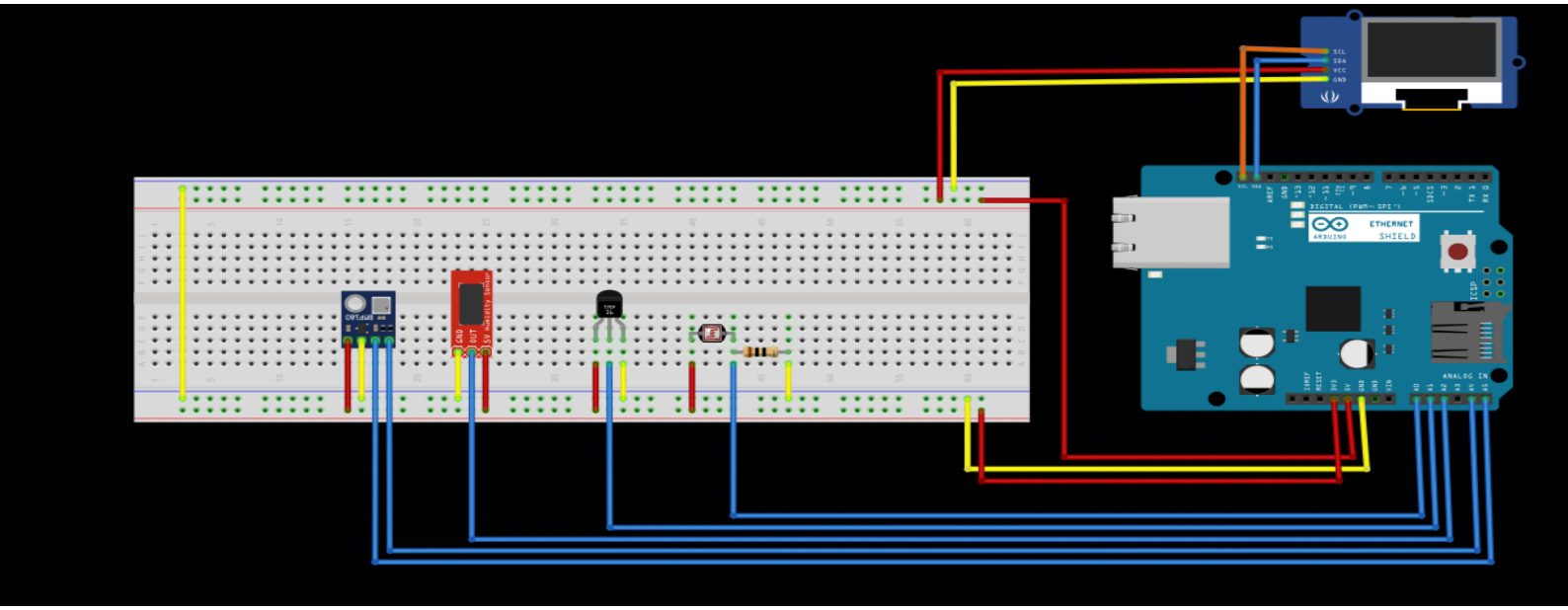
DESCRIZIONE DELL'ESPERIENZA

Si vuole realizzare un sistema di acquisizione dati in rete capace di memorizzare delle misure raccolte da delle stazioni dotate di sensori vari installate in punti diversi della rete. Ogni stazione, realizzata con una scheda Arduino uno e uno Shield Ethernet o con Arduino YUN (rete Wifi), acquisisce le informazioni dai sensori (es. temperatura, umidità, pressione e luminosità) e dopo averli memorizzati localmente li invia con apposito protocollo al server di acquisizione che li elabora e li archivia su un file CSV. Il protocollo applicativo consente il dialogo tra il server e i sistemi di acquisizione, opererà secondo una configurazione client – server e dovrà essere abbastanza flessibile da garantire più stazioni meteo collegate in rete. Infine il sistema deve prevedere un interfaccia(sito web) in grado di elaborare e visualizzare i risultati in modo opportuno all'utente.

FOTO

Nella foto si può vedere la stazione meteo rappresentata dall' Arduino che mostra sul display LCD connesso con il protocollo I2C i dati che raccoglie tramite i suoi sensori. Sotto all'Arduino si trova un computer che opera da server e archivia i dati che l'Arduino gli passa tramite Ethernet.

BREADBOARD FRITZING



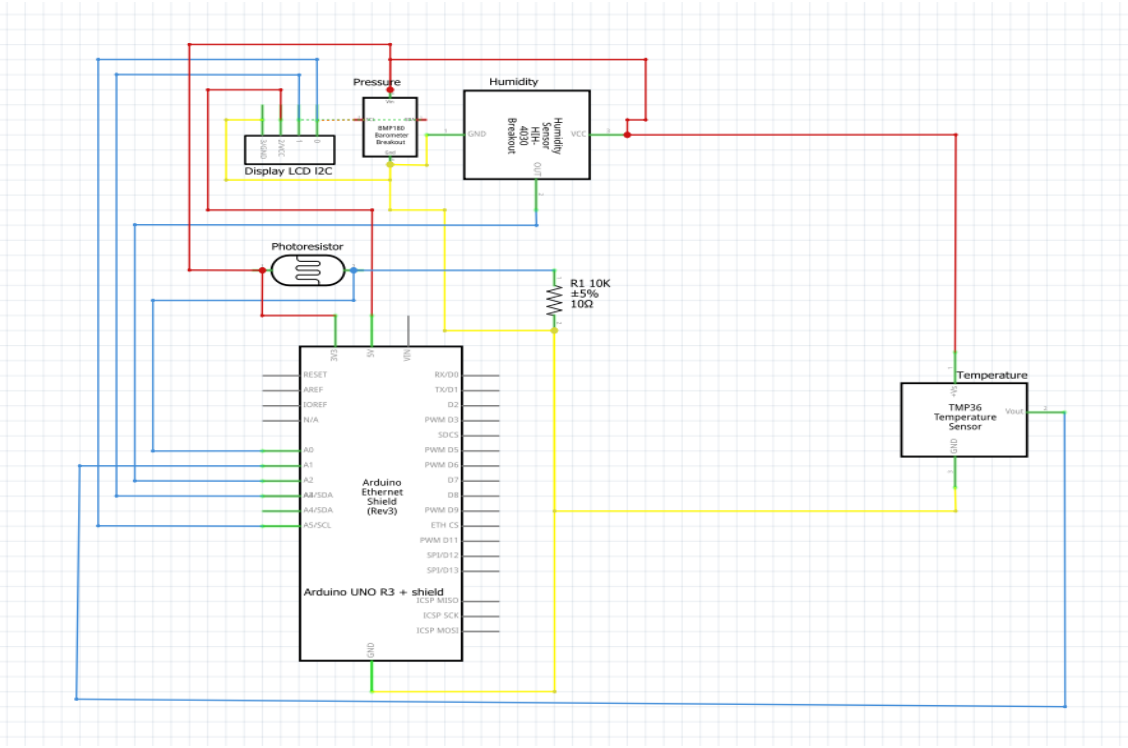
LEGENDA:

GND: CAVI BIANCHI E GIALLI

VCC: CAVI ROSSI

DATI INPUT RX / OUTPUT TX: BLU/AZZURRO

SCHEMA FRITZING



CODICE ARDUINO

Il codice per Arduino può essere visto come se fosse suddiviso in tre parti principali:

Nella prima parte vengono incluse le librerie e dichiarate le variabili che vengono usate da tutto il progetto. Le variabili vengono tutte inizializzate globali ad inizio progetto per evitare gli errori principali dovuti alla allocazione della memoria di Arduino(es. memory overflow), facilitati dal fatto che arduino non ha molta memoria e non ha un memory manager che ne gestisce l'allocazione ad alto livello.

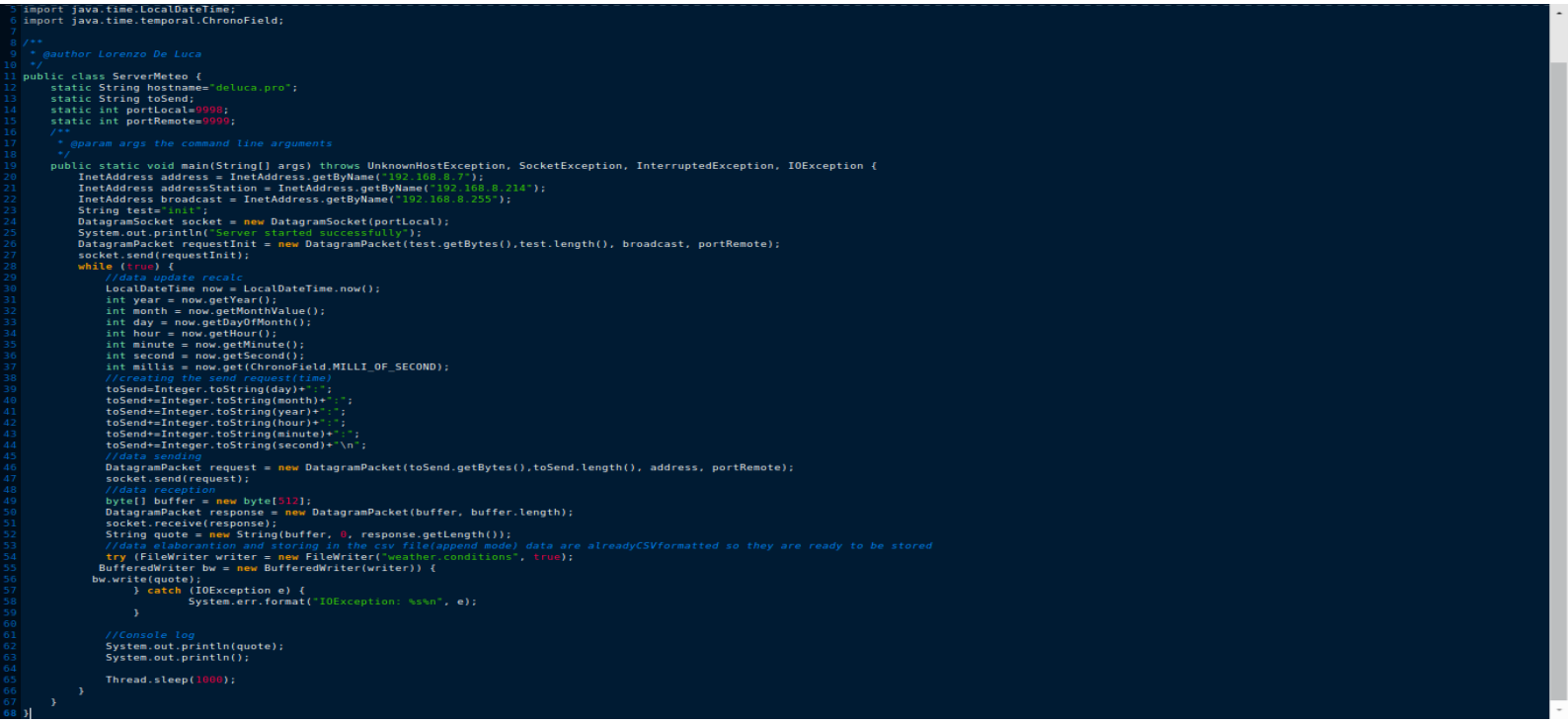
```
void setup() {
  // Open serial communications and wait for port to open:
  Serial.begin(9600);
  while (!Serial) {
    ; // wait for serial port to connect. Needed for native USB port only
  }
  // print your local IP address:
  Serial.print("My IP address: ");
  Ethernet.begin(mac,myIp);
  Serial.println(Ethernet.localIP());
  // start UDP
  Udp.begin(localPort);
  //Udp.beginPacket(serverIp, serverPort);
  //Udp.write("New Arduino station");
  //Udp.endPacket();
}
```

Nella seconda parte(Setup) viene inizializzato lo shield ethernet con gli indirizzi ip e mac definiti prima e viene inizializzata la comunicazione seriale e ethernet udp.

```
void loop() {
  //read and send of the data to the server
  //gestione richieste
   packetSize = Udp.parsePacket();
  if (packetSize) {
    Serial.print("Received packet of size ");
    Serial.println(packetSize);
    Serial.print("From ");
    IPAddress remote = Udp.remoteIP();
    for (int i=0; i < 4; i++) {
      Serial.print(remote[i], DEC);
      if (i < 3) {
        Serial.print(".");
      }
    }
    Serial.print(", port ");
    Serial.println(Udp.remotePort());
    // read the packet into packetBufffer
    Udp.read(packetBuffer, UDP_TX_PACKET_MAX_SIZE);
    Serial.println("Contents:");
    Serial.println(packetBuffer);
    if(packetSize==sizeOfTimePacket||true){//tutti i pacchetti che il server java invia all'arduino sono per aggiornare l'ora
      justUpdated=true;
      //lettura ora formato gg:mm:aaaa:hh:mm:ss
      millisecOnTimeInit=millis();
      day=String(packetBuffer)[0];
      day[1]+=String(packetBuffer)[1];
      month=String(packetBuffer)[3];
      month+=String(packetBuffer)[4];
      year=String(packetBuffer)[6];
      year+=String(packetBuffer)[7];
      year+=String(packetBuffer)[8];
      year+=String(packetBuffer)[9];
      ora=String(packetBuffer)[11];
      ora+=String(packetBuffer)[12];
      minuti=String(packetBuffer)[14];
      minuti+=String(packetBuffer)[15];
      sec=String(packetBuffer)[17];
      sec+=String(packetBuffer)[18];
    }
    // send a reply to the IP address and port that sent us the packet we received(Acknowledge)
```

Infine nella terza parte (loop) viene prima aggiornata dal server/localmente l'ora e poi vengono letti i dati dai sensori e vengono inviati al server Java(in figura la prima parte del loop).

CODICE JAVA



```java
import java.time.LocalDateTime;
import java.time.temporal.ChronoField;

/**
 * @author Lorenzo De Luca
 */
public class ServerMeteo {
    static String hostname="deluca.pro";
    static String toSend;
    static int portLocal=9998;
    static int portRemote=9999;
    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) throws UnknownHostException, SocketException, InterruptedException, IOException {
        InetAddress address = InetAddress.getByName("192.168.8.7");
        InetAddress addressStation = InetAddress.getByName("192.168.8.214");
        InetAddress broadcast = InetAddress.getByName("192.168.8.255");
        String test="init";
        DatagramSocket socket = new DatagramSocket(portLocal);
        System.out.println("Server started successfully");
        DatagramPacket requestInit = new DatagramPacket(test.getBytes(),test.length(), broadcast, portRemote);
        socket.send(requestInit);
        while (true) {
            //data update recalc
            LocalDateTime now = LocalDateTime.now();
            int year = now.getYear();
            int month = now.getMonthValue();
            int day = now.getDayOfMonth();
            int hour = now.getHour();
            int minute = now.getMinute();
            int second = now.getSecond();
            int millis = now.get(ChronoField.MILLI_OF_SECOND);
            //creating the send request(time)
            toSend=Integer.toString(day)+":";
            toSend+=Integer.toString(month)+":";
            toSend+=Integer.toString(year)+":";
            toSend+=Integer.toString(hour)+":";
            toSend+=Integer.toString(minute)+":";
            toSend+=Integer.toString(second)+"\n";
            //data sending
            DatagramPacket request = new DatagramPacket(toSend.getBytes(),toSend.length(), address, portRemote);
            socket.send(request);
            //data reception
            byte[] buffer = new byte[512];
            DatagramPacket response = new DatagramPacket(buffer, buffer.length);
            socket.receive(response);
            String quote = new String(buffer, 0, response.getLength());
            //data elaborantion and storing in the csv file(append mode) data are alreadyCSVformatted so they are ready to be stored
            try (FileWriter writer = new FileWriter("weather.conditions", true);
           BufferedWriter bw = new BufferedWriter(writer)) {
          bw.write(quote);
                } catch (IOException e) {
                        System.err.format("IOException: %s%n", e);
                }

            //Console log
            System.out.println(quote);
            System.out.println();

            Thread.sleep(1000);
        }
    }
}
```

Il codice del server è anche esso diviso in tre parti, nella prima parte vengono inizializzate le variabili tra cui gli indirizzi ip delle macchine con cui comunicherà il server. Nella seconda parte viene inviata se richiesta l'ora alla stazione Arduino per permetterle di aggiornarsela localmente e infine nell'ultima parte viene ricevuta la stringa dall'Arduino contenente i dati dei sensori che ha raccolto e la stringa viene a sua volta trascritta su un file CSV per consentire all'interfaccia web di prelevare i dati.

IL CSV:

Il comma-separated values (abbreviato in CSV) è un formato di file basato su file di testo utilizzato per l'importazione ed esportazione (ad esempio da fogli elettronici o database) di una tabella di dati.
Non esiste uno standard formale che lo definisca, ma solo alcune prassi più o meno consolidate. In questo formato, ogni riga della tabella (o *record* della base dati) è normalmente rappresentata da una linea di testo, che a sua volta è divisa in campi (le singole colonne) separati da un apposito carattere separatore, ciascuno dei quali rappresenta un valore.

CODICE SITO WEB(HTML 5, PHP 7)

```
<!-- IT/Venice,9,5,2019,14,5,23,100,19,54,1018 -->
<thead>
    <tr><th>Day/Night</th><th>Location</th><th>Time</th><th>brightness</th><th>temperature</th><th>umidity</th><th>pressure</th></tr>
</thead>
<tbody>
    <?php
        //retriving get params
        $day_filter="";
        $month_filter="";
        $year_filter="";
        $loc_filter="";
        if(!empty($_GET['f-day']))$day_filter=$_GET['f-day'];
        if(!empty($_GET['f-month']))$month_filter=$_GET['f-month'];
        if(!empty($_GET['f-year']))$year_filter=$_GET['f-year'];
        if(!empty($_GET['f-loc']))$loc_filter=$_GET['f-loc'];
        $filename = '../server/ServerMeteoDeLuca/weather.conditions';
        // The nested array to hold all the arrays
        $weather = [];
        // Open the file for reading
        if (($h = fopen("{$filename}", "r")) !== FALSE) {
            // Each line in the file is converted into an individual array that we call $data
            // The items of the array are comma separated
            while (($data = fgetcsv($h, 1000, ",")) !== FALSE)
            {
                // Each individual array is being pushed into the nested array
                $weather[] = $data;
            }
            // Close the file
            fclose($h);
        }
        //Building the table with the data extracted from the weather csv file
        $build="";
        foreach($weather as $row){
            if(//filters
                ($day_filter=="" || $day_filter==$row[1])&&
                ($month_filter=="" || $month_filter==$row[2])&&
                ($year_filter=="" || strstr($row[3],$year_filter))&&
                ($loc_filter=="" || strstr($row[0],$loc_filter))
            ){
                $build .= '<tr>';
                if($row[4]>6&&$row[4]<19){ // day
                    if($row[10]>1000){/* rainy or sunny based on the pressure */
                        $build .= '<td><img src="day.png"></td>';
                    }else {
                        $build .= '<td><img src="day-cloudy.png"></td>';
                    }
                }else{ //night
                    if($row[10]>1000){    /* rainy or sunny based on the pressure */
                        $build .= '<td><img src="night.png"></td>';
                    }else{
                        $build .= '<td><img src="night-cloudy.png"></td>';
                    }
                }
                $build .= "<td>{$row[0]}</td>"; /* location */
                $build .= "<td>{$row[1]}/{$row[2]}/{$row[3]} {$row[4]}:{$row[5]}:{$row[6]}</td>"; /* time */
                $build .= "<td>{$row[7]}</td>"; /* light */
                $build .= "<td>{$row[8]}</td>"; /* temperature */
                $build .= "<td>{$row[9]}</td>"; /* umidity */
                $build .= "<td>{$row[10]}</td>"; /* pressure */
                $build .= "</tr> \n";
            }
        }
        echo ($build);
```

Il codice delle pagine web è scritto usando la sintassi classica con gli elementi dello standard html 5. Nella pagina in cui sono mostrate le previsioni meteo è presente una tabella che viene generata dal preprocessore di hypertext PHP7. Il PHP si occupa di leggere dal file scritto dal server Java i dati salvati in formato csv. Per leggerli utilizza una funzione predefinita del php per essere il più efficiente possibile. Poi gli elabora calcolando le previsione e li restituisce nella forma di tabella html.

HTML 5:
L'**HTML5** è un linguaggio di markup per la strutturazione delle pagine web, pubblicato come W3C Recommendation da ottobre 2014.

PHP7:
PHP (acronimo ricorsivo di "**PHP**: Hypertext Preprocessor", preprocessore di ipertesti) è un linguaggio di scripting interpretato, originariamente concepito per la programmazione di pagine web dinamiche.

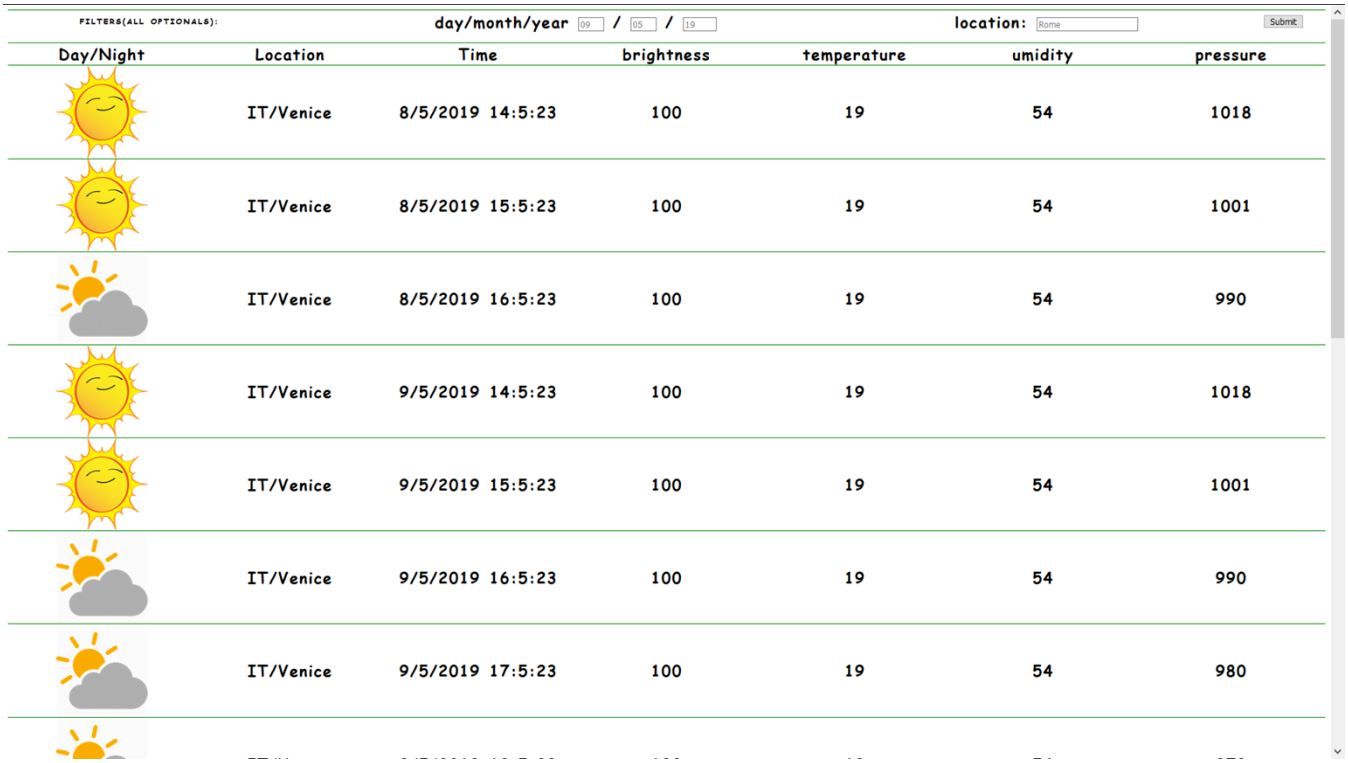
RELAZIONE

Lo scopo di questo progetto era quello di realizzare una rete di stazione

meteorologiche connesse a internet che raccolgono dei dati attraverso dei sensori(luminosità, umidità, pressione e temperatura). Queste stazioni devono attraverso l'utilizzo dello shield ethernet inviare queste informazioni attraverso la rete al server Java che le elabora e le salva su un file scritto con la sintassi CSV(comma separated values). Infine c'è un sito web scritto utilizzando i linguaggi HTML, CSS, JavaScript e PHP versione 7 che legge i dati dal file e li elabora mostrandoli all'utente. Il sito web elabora anche i dati per ***mostrare all'utente una previsione delle condizioni meteorologiche(sereno-pioggia)*** e se con quei dati è più probabile che piova o che ci sia il cielo sereno. Per poter funzionare questo sistema necessita che tutti i componenti siano collegati tra di loro in rete e bisogna configurare nei vari codici gli indirizzi ip degli host per consentire alle macchine di comunicare correttamente. Per provare il progetto basta comporre il circuito hardware per l'Arduino e farlo partire, fare partire il programma server e aprire su un host il sito dell'interfaccia grafica.

L'INTERFACCIA GRAFICA(sito web html js php)

3 possibile di visualizzazione dei valori!!!
MONDO-ITALIA-SINGOLE CITTA'
selezionabili cliccando nel punto desiderato



FILTERS(ALL OPTIONALS): day/month/year 09 / 05 / 19 location: Rome Submit

Day/Night	Location	Time	brightness	temperature	umidity	pressure
	IT/Venice	8/5/2019 14:5:23	100	19	54	1018
	IT/Venice	8/5/2019 15:5:23	100	19	54	1001
	IT/Venice	8/5/2019 16:5:23	100	19	54	990
	IT/Venice	9/5/2019 14:5:23	100	19	54	1018
	IT/Venice	9/5/2019 15:5:23	100	19	54	1001
	IT/Venice	9/5/2019 16:5:23	100	19	54	990
	IT/Venice	9/5/2019 17:5:23	100	19	54	980

RISULTATO DELL'ELABORAZIONE

N.B.: L'immagine a sinistra nella prima colonna della tabella indica le previsioni metereologiche calcolando se il tempo in quel momento è sereno o nuvoloso/piovoso basandosi sulla pressione rilevata.

L'interfaccia grafica è costituita da un sito web, lo ho scritto interamente io senza utilizzare librerie o framework esterni. Il sito web è multipiattaforma, responsive e visualizzabile su tutti i browser attualmente più utilizzati. Offre la possibilità di filtrare i dati per locazione(città/stato) e orario di registrazione. Le pagine sono HTML con del codice js per gestire gli eventi e degli script PHP per elaborare il file csv da cui preleva i dati.

DEFINIZIONE: L'**interfaccia grafica** (nota anche come GUI (dall'inglese Graphical User Interface), in informatica è un tipo di **interfaccia** utente che consente l'interazione uomo-macchina in modo visuale utilizzando rappresentazioni **grafiche** piuttosto che utilizzando una **interfaccia** a riga di comando.

COME UTILIZZARE L'INTERFACCIA: Per far funzionare l'interfaccia è necessario eseguire index.html in un web server(consiglio di usare Apache 2 + PHP) . Per ottenere velocemente un ambiente dove usare il programma se non si ha un server web già pronto su windows si può scaricare da internet XAMPP Portable e bisogna inserire la vartella dell'interfaccia eliminando i file già presenti nella cartella htdocs. Per avviare il web server è presente un programma intuitivo

insieme ad XAMPP… Basta avviare apache 2, php verrà eseguito automaticamente.
Poi per avviare l'interfaccia sarà sufficiente andare su un browser all' indirizzo: localhost(127.0.0.1), l'interfaccia è accessibile anche da un altro dispositivo host inserendo al post dell'ip dall'altro host l'indirizzo della macchina dove sta girando il web-server. Un web server alternativo utilizzabile potrebbe essere NGINX anche se è più complicato per un utente base da configurare.
Informazioni sulle risorse utilizzate(per utenti nuovi nel settore):
XAMPP è una piattaforma software multipiattaforma e libera costituita da Apache HTTP Server, il database MariaDB e tutti gli strumenti necessari per utilizzare i linguaggi di programmazione PHP e Perl. Il nome è un acronimo dei software sopra citati.
Apache HTTP Server, o più comunemente Apache, è il nome di un server web libero sviluppato dalla Apache Software Foundation. È la piattaforma server Web modulare più diffusa, in grado di operare su una grande varietà di sistemi operativi, tra cui UNIX/Linux, Microsoft Windows e OpenVMS.
HTTP: In telecomunicazioni e informatica l'HyperText Transfer Protocol è un protocollo a livello applicativo usato come principale sistema per la trasmissione d'informazioni sul web ovvero in un'architettura tipica client-server. Le specifiche del protocollo sono gestite dal World Wide Web Consortium.
PHP: PHP è un linguaggio di scripting lato server e un potente strumento per creare pagine Web dinamiche e interattive. PHP è un'alternativa ampiamente utilizzata, gratuita ed efficiente per concorrenti come Microsoft ASP.PHP 7 è l'ultima versione stabile. Questo tutorial utilizza PHP 7.2.10. Il 77% dei siti web al mondo è realizzato in PHP.
HTML è l'acronimo di *HyperText Markup Language* ("Linguaggio di contrassegno per gli Ipertesti") e ***non è un linguaggio di programmazione***. Si tratta invece di un **linguaggio di markup** (di 'contrassegno' o 'di marcatura'), che permette di indicare come disporre gli elementi all'interno di una pagina.

IL SERVER IN JAVA

Il codice del server in java che si occupa di ricevere i dati dal Arduino è stato composto usando l'ide di testo NETBEANS versione 11. Sviluppato da Apache(la stessa società del web server). Per ricevere i dati utilizzando il protocollo UDP non vengono utilizzate altro che le libreri java standard
java.io.* e java.net.*. Perciò il programma è particolarmente efficiente anche perché tutti i meccanismi sono fatti con istruzioni a basso livello molto veloci ed efficienti. Il programma è ingrado di inoltrare l'ora corrente alle stazioni Arduino

ed è ingrado di ricevere da loro i dati metereologici che raccolgono. Dopo che ha ricevuto i dati dalla stazione li salva in un file con sintassi CSV(weather.conditions). Per inviare e ricevere i dati il programma si basa sui DataGram appunto udp e gli oggetti DataSocket DataPacket e InetAddress per evalutare gli indirizzi ip e inviare/ricevere i dati.

Dettagli:

Lo **User Datagram Protocol** (**UDP**), nelle telecomunicazioni, è uno dei principali protocolli di rete della suite di protocolli Internet. È un protocollo di livello di trasporto a pacchetto, usato di solito in combinazione con il protocollo di livello di rete IP. A differenza del TCP, l'UDP è un protocollo di tipo connectionless, inoltre non gestisce il riordinamento dei pacchetti né la ritrasmissione di quelli persi, ed è perciò generalmente considerato di minore affidabilità. In compenso è molto rapido (non c'è latenza per riordino e ritrasmissione) ed efficiente per le applicazioni "leggere" o time-sensitive. Ad esempio, è usato spesso per la trasmissione di informazioni audio-video *real-time* come nel caso delle trasmissioni Voip.

Infatti, visto che le applicazioni in tempo reale richiedono spesso un bit-rate minimo di trasmissione, non vogliono ritardare eccessivamente la trasmissione dei pacchetti e possono tollerare qualche perdita di dati, il modello di servizio TCP può non essere particolarmente adatto alle loro caratteristiche.

Un datagramma è un'unità di trasferimento di base associata a una rete a commutazione di pacchetto. I datagrammi sono in genere strutturati nelle sezioni di intestazione e payload. I datagrammi forniscono un servizio di comunicazione senza connessione su una rete a commutazione di pacchetto(come le trasmissioni udp).

ARDUINO(Dettagli applicativi e protocollo di invio)

L'Arduino ha il compito di leggere i dati dai sensori e di inoltrarli al server Java che li elabora, scrive anche i dati su un display LCD per renderli immediatamente consultabili in tempo reale. Il display è collegato utilizzando il protocollo I2C con un adattatore per ottimizzare il numero di pin necessari sulla scheda per collegarlo(ne usa solo 2 oltre Vcc e gnd). Il codice è molto efficiente poiché invia i dati senza elaborarli troppo e quindi senza impiegare troppo tempo per editarli perché non fa altro che raggrupparli separandoli da un carattere separatore. Il vantaggio di questo protocollo è che i dati non sono

solo elaborati incredibilmente velocemente ma sono anche molto veloci poi da leggere per il server che li riceve che non deve spendere troppo tempo nel gestire il carico applicativo del pacchetto. Gli errori di trasmissione e di comunicazione sono gestiti dagli applicativi predefiniti forniti dal sistema operativo in collaborazione con la NIC(scheda di rete) quindo i due programmi stazione e server non si devono preoccupare di gestire eventuali errori di trasmissione. La stazione di Arduino è anche programmata per tenere l'orario aggiornato dopo che lo ha ricevuto sincronizzandosi con l'orario del server. Nel codice dell'Arduino è stata fatta particolare attenzione alla gestione delle variabili e alla ottimizzazione del codice.

Informazioni generali:

Il protocollo i2c è un sistema di comunicazione seriale bifilare utilizzato tra circuiti integrati. Il classico bus I²C è composto da almeno un *master* ed uno *slave* (letteralmente "capo, padrone" e "sottoposto, schiavo").
La situazione più frequente vede un singolo *master* e più *slave*; possono tuttavia essere usate architetture *multimaster* e *multislave* in sistemi più complessi.

In informatica la **scheda di rete** è a livello logico un'interfaccia digitale, costituita a livello hardware da una scheda elettronica inserita/alloggiata solitamente all'interno di un personal computer, server, stampante, router ecc., che svolge tutte le elaborazioni o funzioni necessarie a consentire la connessione dell'apparato informatico ad una rete informatica.

Il sistema di sincronizzazione dell'ora è un sistema **real-time** (in italiano "computazione in tempo reale") e cioè indica che le computazioni per le quali la correttezza del risultato dipendono non solo dalla correttezza logica ma anche dal tempo di risposta. Estendendo la definizione, possono essere anche aggiunti requisiti di affidabilità e interazione con l'ambiente.

DETTAGLI APPLICATIVI, PROBLEMI RISCONTRATI E SOLUZIONI ATTUATE

Non ho riscontrato problemi significativi nella realizzazione del sistema sul lato della composizione del circuito hardware e nell'interazione tra i vari componenti del sistema. L'unica difficoltà che ho dovuto affrontare è stata la mancanza di alcuni sensori che ho scelto di far utilizzare alla stazione meteo(Arduino) . Per risolvere il problema, come consigliato nel testo della consegna del progetto, le misurazioni per cui non disponevo della componentistica hardware le ho simulate sostituendo l'input al pin digitale con un generatore random tarato per dare per quei valori risultati vero simili.