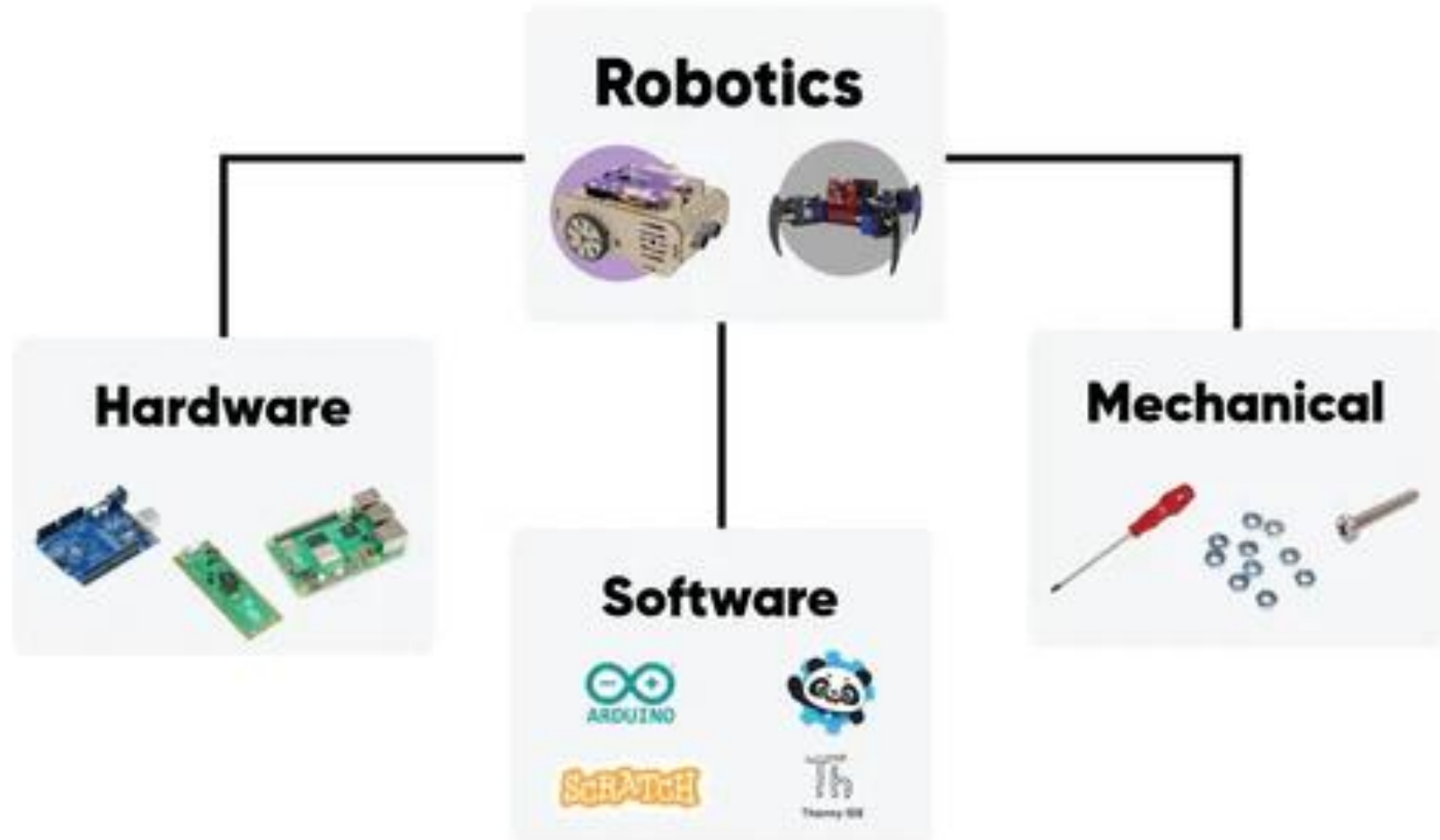


CORSO AVANZATO DI INFORMATICA E ROBOTICA

LEZIONE 8: recap e nuovi argomenti

Hardware vs Software

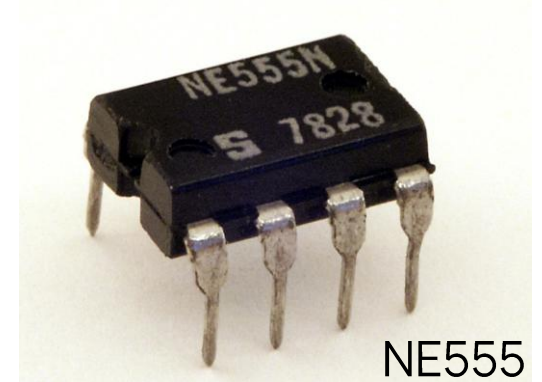


Hardware e Software... perché usare Arduino?

Con la sola elettronica posso fare cose semplici e fisse.

Con hardware + software posso fare cose intelligenti, adattabili e creative.

Arduino serve proprio a questo: dare un cervello all'elettronica.



Molte cose si potrebbero fare solo elettronicamente... MA:

Solo elettronica (HW)	HW + Software
Rigido	Flessibile
Difficile da modificare	Basta cambiare il codice
Ogni funzione = nuovi componenti	Stesso hardware, mille funzioni
Poco scalabile	Estremamente scalabile

Programma Arduino

Un programma Arduino è **una lista di ordini eseguiti dall'alto verso il basso**, con una parte che gira **una volta** (il setup) e una che gira **per sempre** (il loop).

Il software consente di agire sui componenti hardware



con la giusta **sintassi, librerie e skill** potete utilizzare qualsiasi componente hardware vi capiti!

es: smontate una stampante?

```
1 void setup() {  
2  
3 }  
4  
5 void loop() {  
6  
7   //Do first...  
8   //Do this next...  
9   //Do this too...  
10  
11 }
```

Runs once

Runs over and over and over...

Librerie

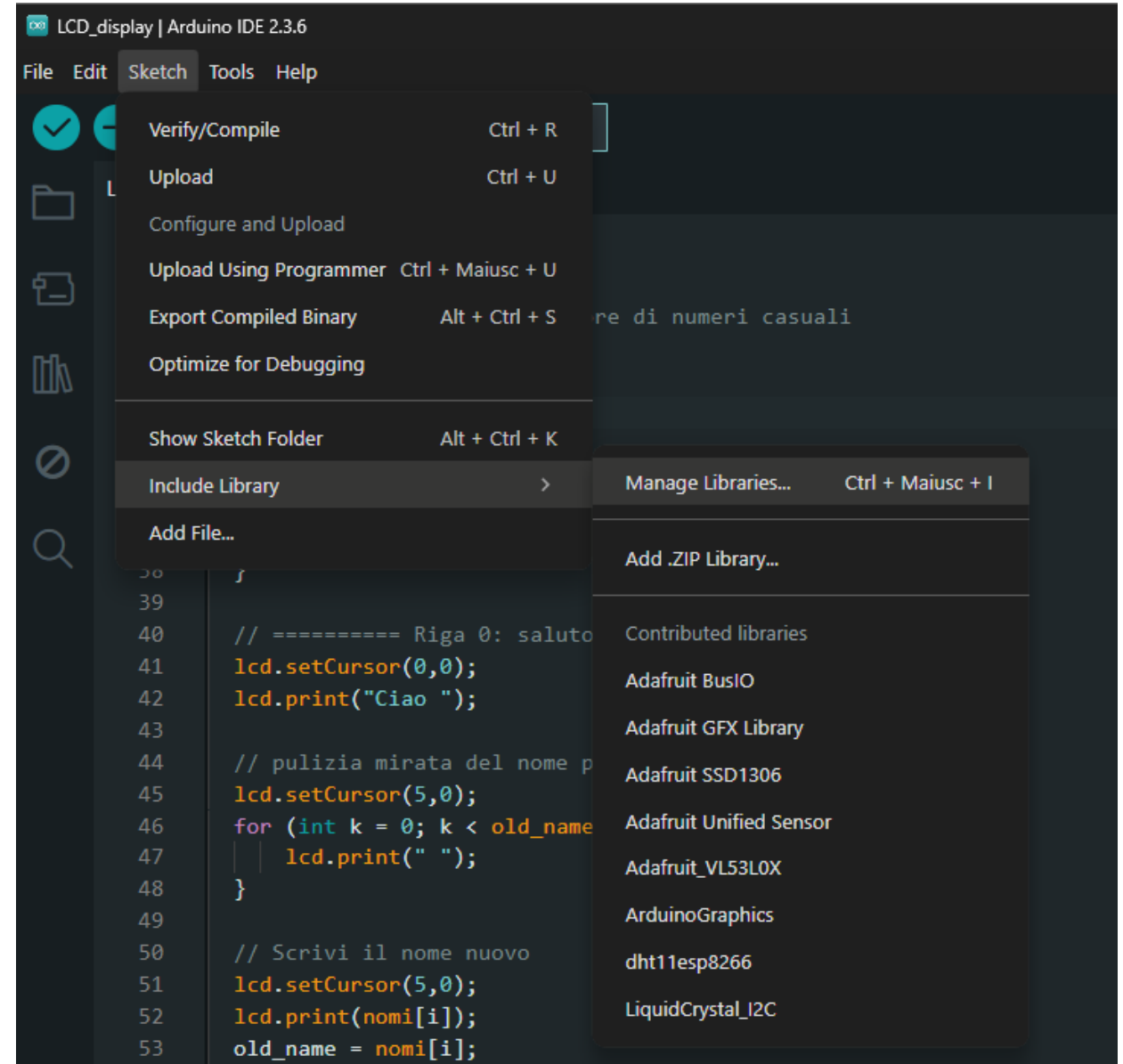
Una libreria è un insieme di funzioni già pronte che servono per usare facilmente qualcosa di specifico.

Senza librerie, ogni volta si dovrebbe:

- capire il funzionamento del sensore/attuatore da integrare
- gestire segnali elettrici dell'interfaccia
- scrivere decine o centinaia di righe di codice

Sarebbe come costruire una macchina partendo dai bulloni!

- ✓ Una libreria è codice già scritto
- ✓ Serve per usare componenti o funzionalità complesse
- ✓ Ti fa programmare più velocemente
- ✓ Nasconde i dettagli difficili



Esempio: la libreria <LiquidCrystal_I2C.h>

Spesso le librerie sono scritte utilizzando la **programmazione ad oggetti**

Esempio:

Repository della libreria (aggiungibile da «Manage Libraries» all'interno di Arduino IDE!):

https://github.com/johnrickman/LiquidCrystal_I2C

Ma come si usa?

...

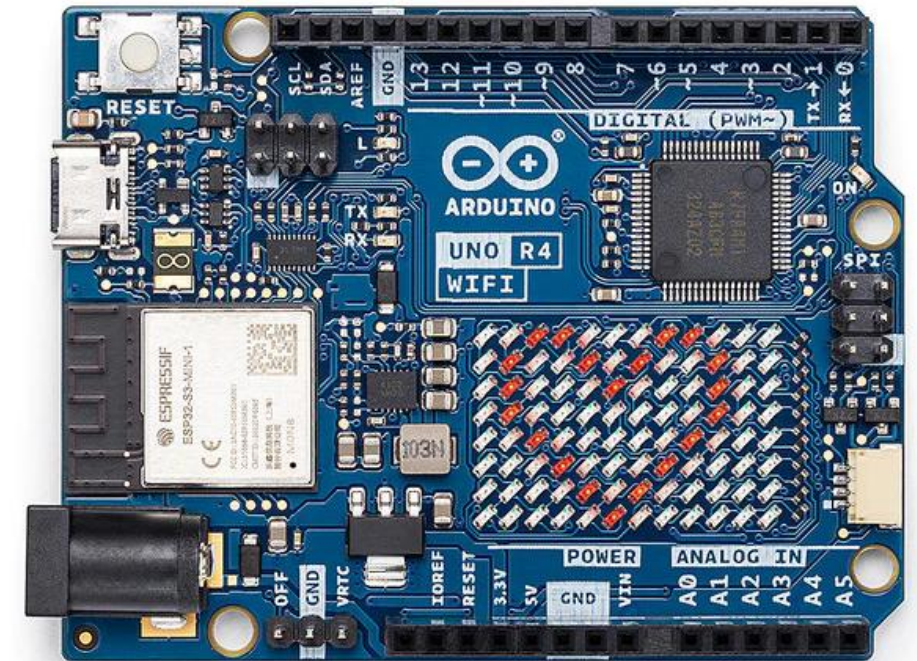
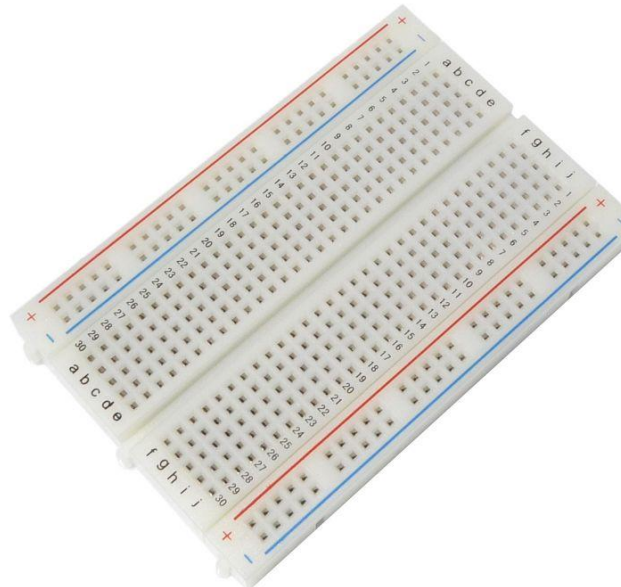
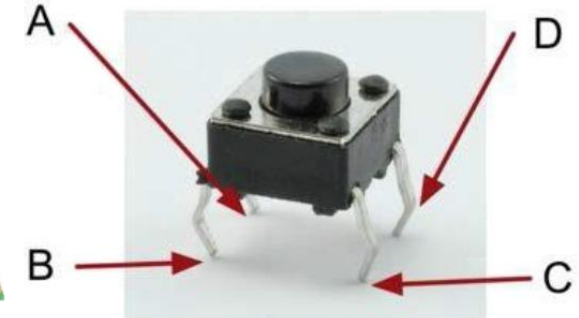
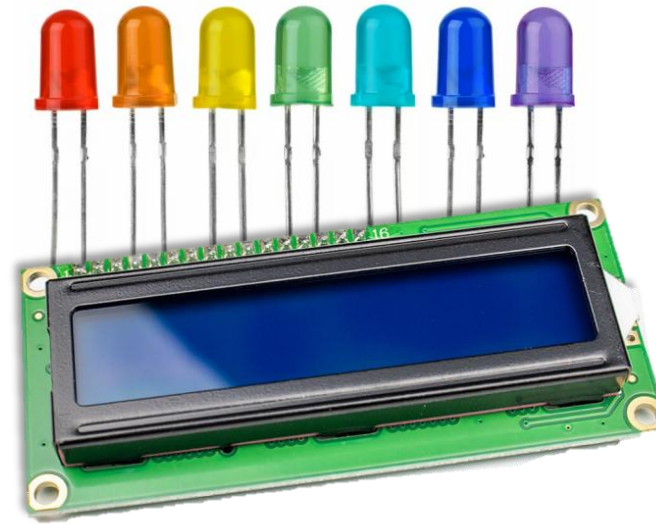
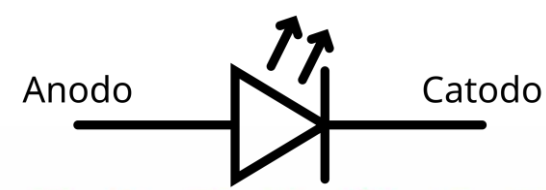
```
55  class LiquidCrystal_I2C : public Print {  
56      public:  
57          LiquidCrystal_I2C(uint8_t lcd_Addr,uint8_t lcd_cols,uint8_t lcd_rows);  
58          void begin(uint8_t cols, uint8_t rows, uint8_t charsize = LCD_5x8DOTS );  
59          void clear();  
60          void home();  
61          void noDisplay();  
62          void display();  
63          void noBlink();  
64          void blink();  
65          void noCursor();  
66          void cursor();  
67          void scrollDisplayLeft();  
68          void scrollDisplayRight();  
69          void printLeft();  
70          void printRight();  
71          void leftToRight();  
72          void rightToLeft();  
73          void shiftIncrement();  
74          void shiftDecrement();  
75          void noBacklight();  
76          void backlight();  
77          void autoscroll();
```

```
4  #include <LiquidCrystal_I2C.h>
5
6  LiquidCrystal_I2C lcd(0x27,20,4); // set the LCD address to 0x27 for a 16 chars and 2 line display
7
8  void setup()
9  {
10     lcd.init();                // initialize the lcd
11     // Print a message to the LCD.
12     lcd.backlight();
13     lcd.setCursor(3,0);
14     lcd.print("Hello, world!");
15     lcd.setCursor(2,1);
16     lcd.print("Ywrobot Arduino!");
17     lcd.setCursor(0,2);
18     lcd.print("Arduino LCM IIC 2004");
19     lcd.setCursor(2,3);
20     lcd.print("Power By Ec-yuan!");
21 }
22
23
24 void loop()
25 {
26 }
```

```
55  class LiquidCrystal_I2C : public Print {
56  public:
57     LiquidCrystal_I2C(uint8_t lcd_Addr,uint8_t lcd_cols,uint8_t lcd_rows);
58     void begin(uint8_t cols, uint8_t rows, uint8_t charsize = LCD_5x8DOTS );
59     void clear();
60     void home();
61     void noDisplay();
62     void display();
63     void noBlink();
64     void blink();
65     void noCursor();
66     void cursor();
67     void scrollDisplayLeft();
68     void scrollDisplayRight();
69     void printLeft();
70     void printRight();
71     void leftToRight();
72     void rightToLeft();
73     void shiftIncrement();
74     void shiftDecrement();
75     void noBacklight();
76     void backlight();
77     void autoscroll();
```


Compendi Hardware

- Come (non) fare un cortocircuito ed evitare di invertire la polarità dei componenti
- Struttura generale di ARDUINO
- La breadboard e collegamenti dei componenti
- Resistori, potenziometri e fotoresistori
- Funzionamento di un LED, di matrici di LED
- Funzionamento di un interruttore / bottone
- LCD screen
- Servo motore
- Interfaccia wi-fi di Arduino UNO R4
- ...



Documentazione e Datasheet

<https://docs.arduino.cc/>

Arduino Documentation

Browse through all our documentation to learn everything for your Arduino journey.

Hardware

The vital pieces of hardware documentation you need to start your Arduino projects.

[BROWSE HARDWARE →](#)



Cloud

Arduino Cloud is an online platform that allows you to create, deploy and monitor IoT projects.

[DISCOVER CLOUD →](#)



Software

Learn about the IDEs, Web Editor, CLI and all the software tools that you need to get your hands dirty.

[DISCOVER SOFTWARE →](#)



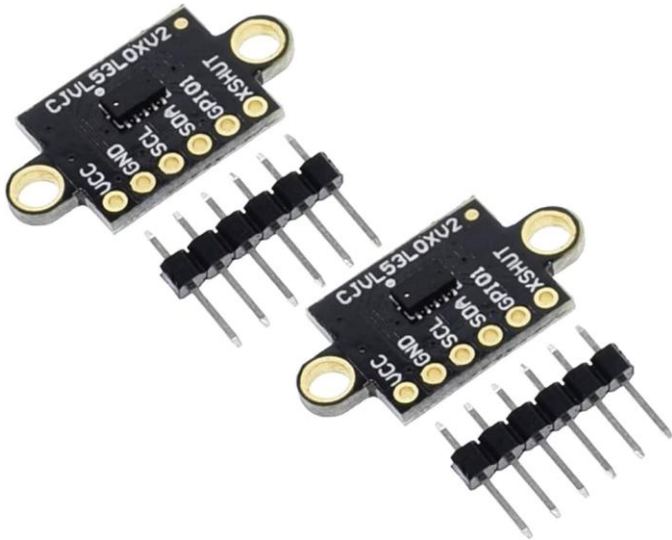
Programming

All you need to know to program with Arduino, including library documentation.

[START CODING →](#)



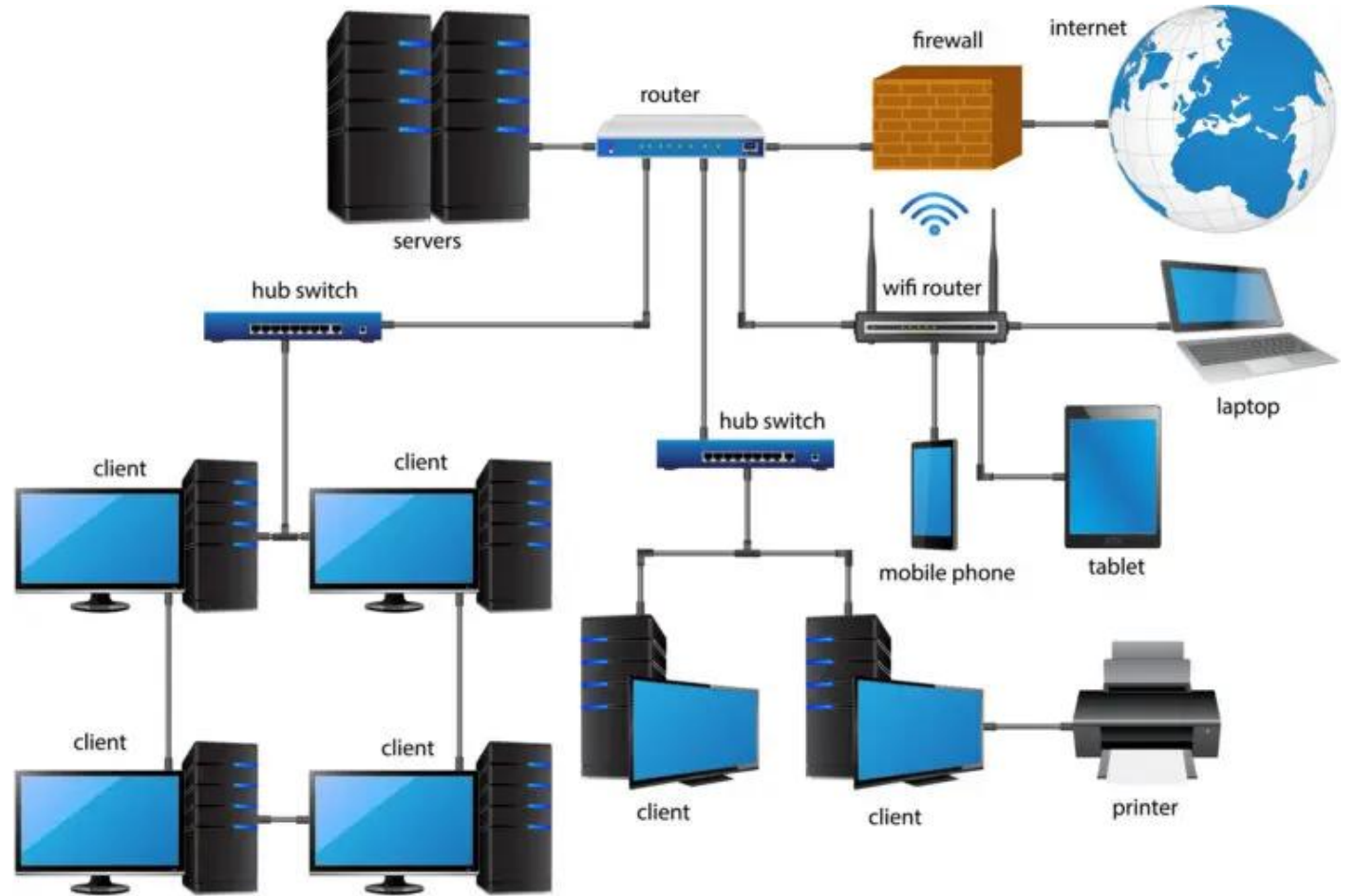
Altri datasheet (es. di sensori o attuatori) è fruibile online andando a cercare il codice del componente, es:



→ CJVL53L0X V2 → Time-of-Flight Distance Sensor

Concetti di Reti e Internet

Prossima lezione!



LAN Network Diagram

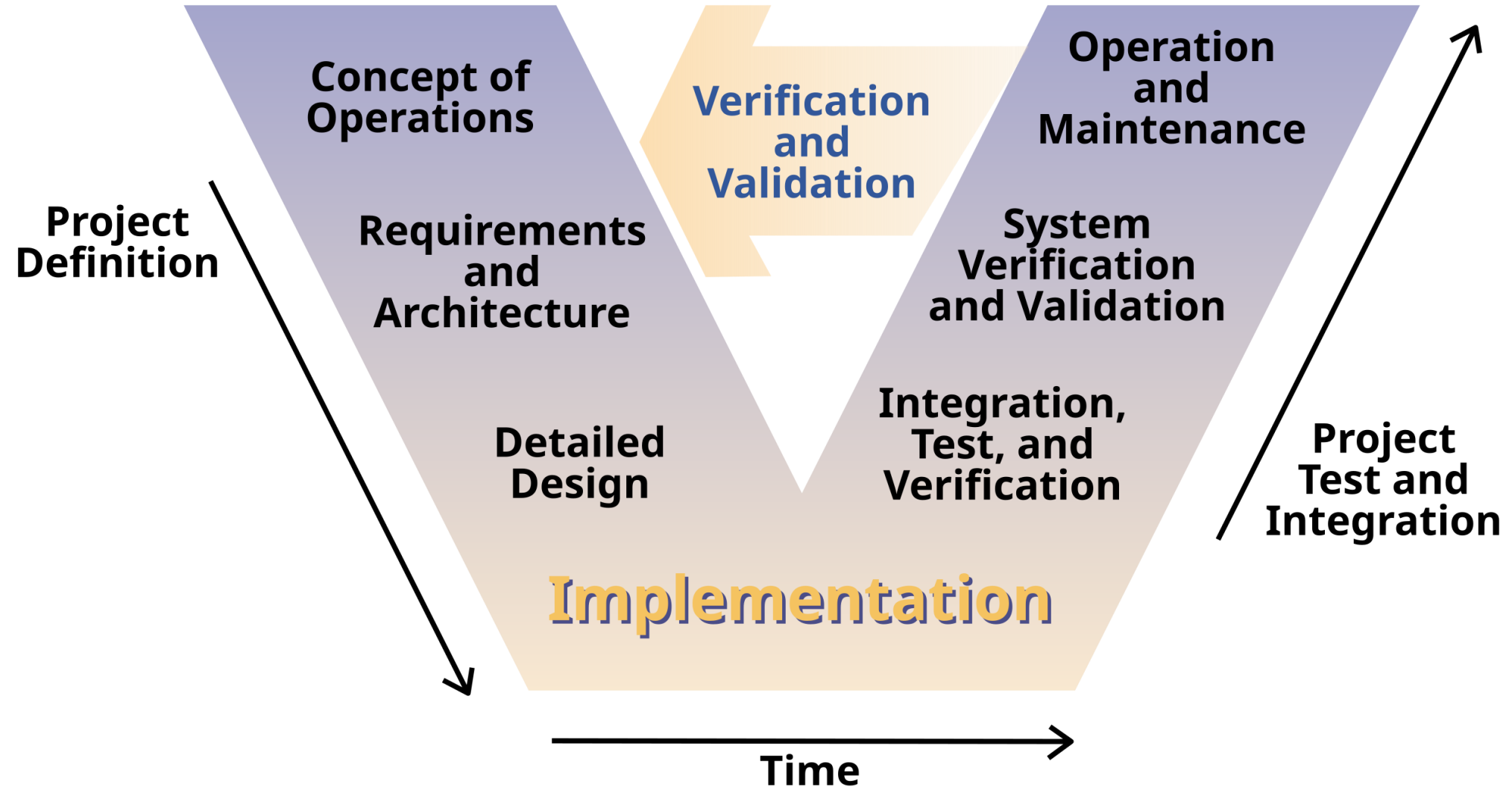


Mini-progettino su Arduino!

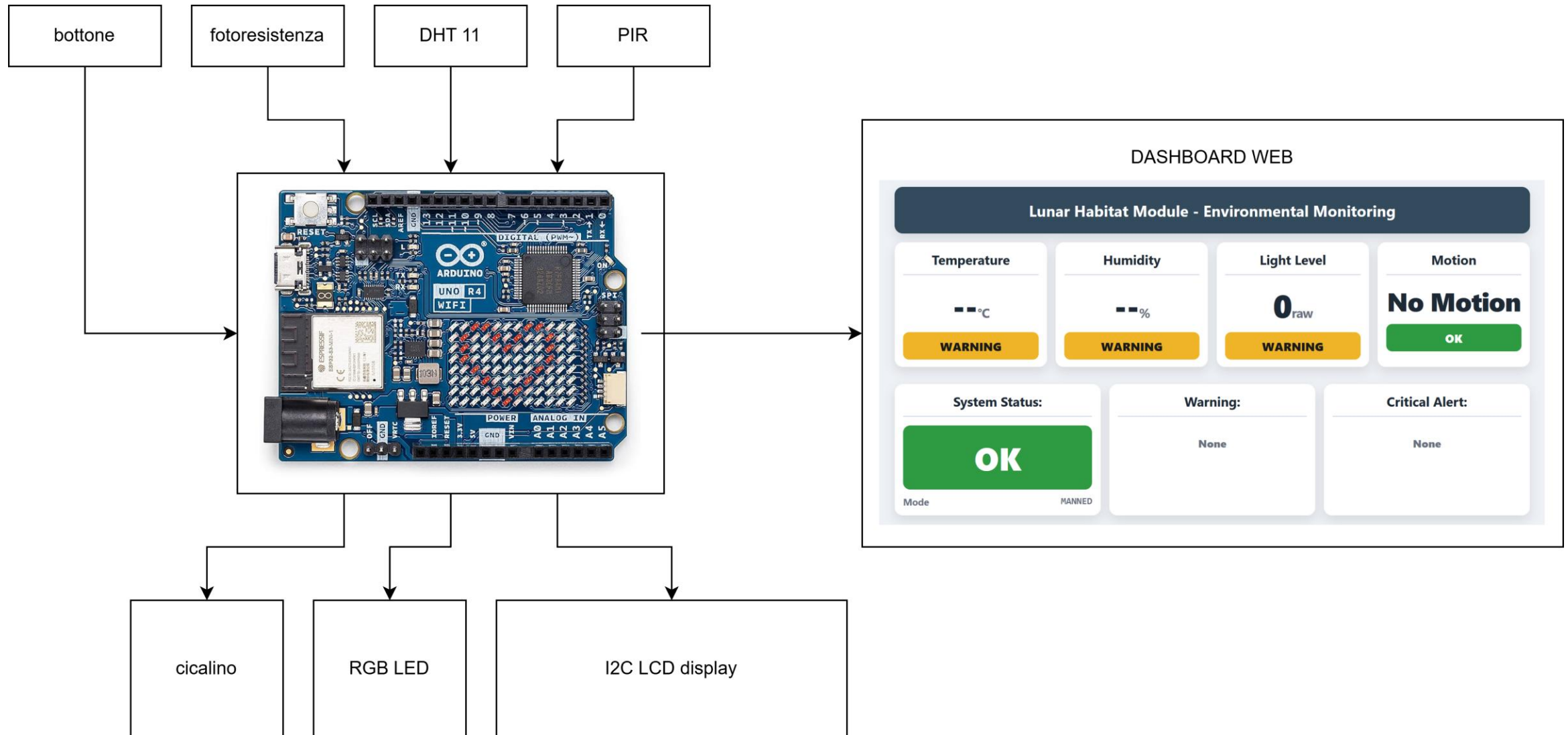
Lunar Habitat Module – Environmental Monitoring System



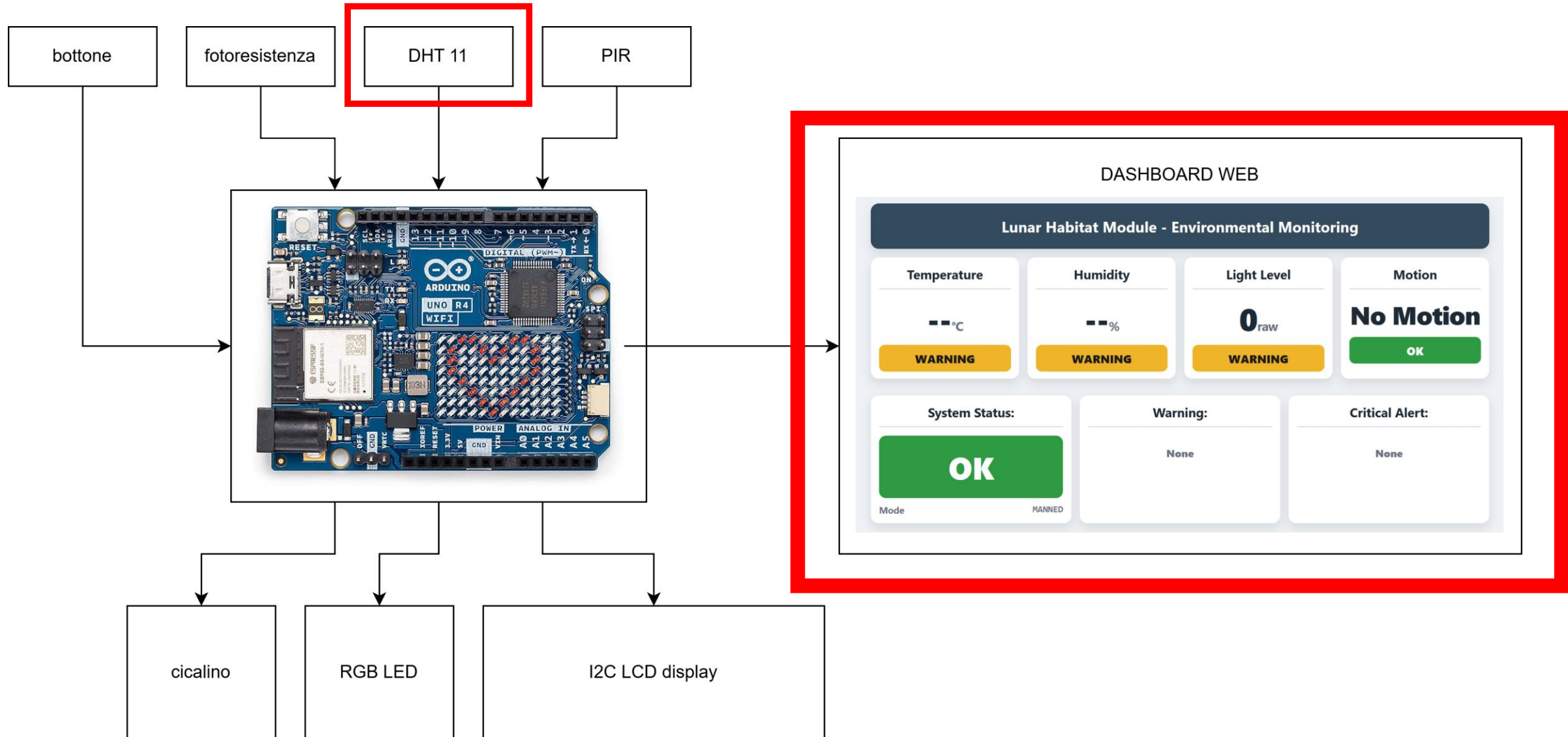
Come procedere



Analisi Funzionale e Architettura

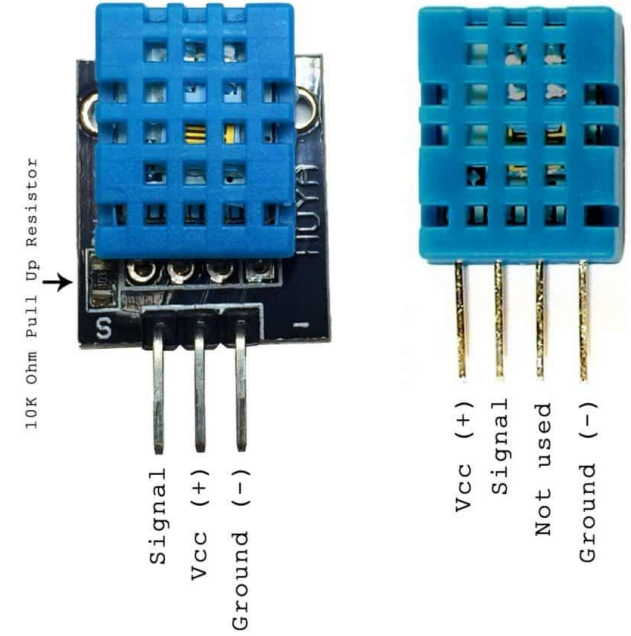
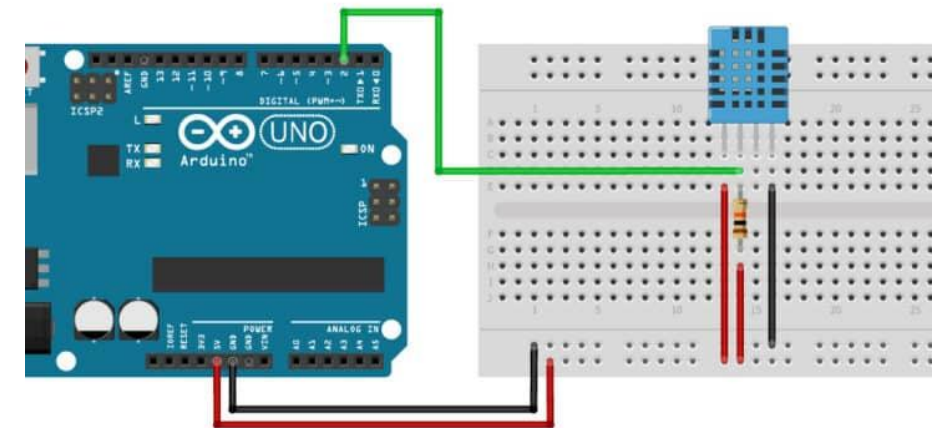
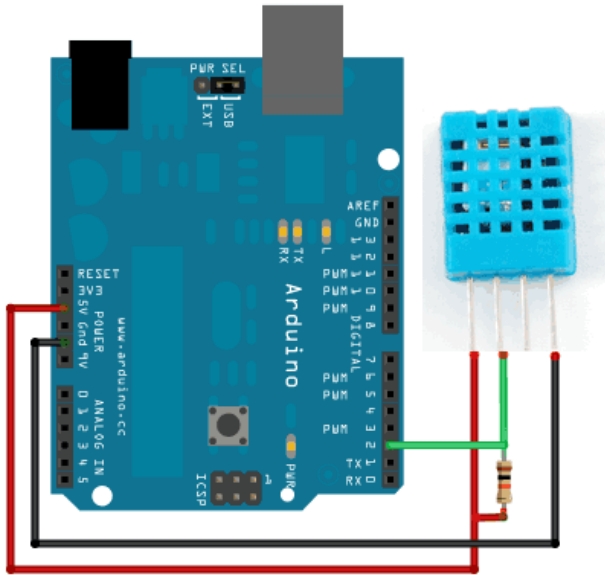


Analisi Funzionale e Architettura



Sensore DHT

<https://github.com/adafruit/DHT-sensor-library>



Sensore DHT – codice

```
1 #include <DHT.h>  
2
```

Interfaccia WEB



Domande, Dubbi, Perplessità?