

Relazione finale per l'esame di Progettazione e Produzione Multimediale

Giovanna Scaramuzzino, Lorenzo Pisaneschi

Firenze, Settembre 2018

1 Introduzione: il progetto myThermostat

In questo documento presentiamo il progetto myThermostat, un semplice meccanismo che permette di monitorare la temperatura di ambienti chiusi. Ne presentiamo le componenti fisiche, le tecnologie software utilizzate, l'implementazione, l'architettura e l'uso finale.

2 Target di utilizzo

myThermostat nasce dalla necessità di avere sempre sotto controllo la temperatura e l'umidità. myThermostat è volto all'utilizzo in ambienti chiusi, come uffici e stanze domestiche. E' possibile utilizzare più sensori per controllare più stanze e, mediante una applicazione web, accedere ad una pagina di controllo dei termostati, come vedremo in seguito.

3 Il sensore

Dopo questa breve introduzione, cominciamo a parlare della vera e propria implementazione del sistema, partendo dal sensore e dalle componenti hardware e software utilizzate per le misurazioni. Per ottenere un oggetto il più completo ed integrato possibile, è stato scelto di utilizzare il software Arduino, un IDE basato sulla tecnologia Java, che permette la programmazione per la gestione di sensori.

Dal punto di vista hardware, myThermostat si basa sulle seguenti componenti:

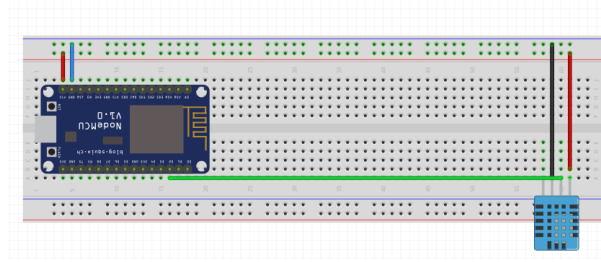


Figura 1: il circuito costituente il dispositivo myThermostat.

- **Breadboard**, per la creazione del circuito elettrico;
- **NodeMCU**, una piattaforma Open Source che monta il modulo WI-FI **ESP8266**, di cui parleremo nel seguito, in quanto componente fondamentale per la trasmissione dei dati via internet dal sensore;
- **DHT11**, il sensore di temperatura ed umidità che funziona grazie ad un processore ad 8 bit single chip.

Descritta la parte hardware, possiamo adesso occuparci della parte software, partendo innanzitutto da Arduino, per poi continuare la nostra trattazione verso la Web Application associata a myThermostat, concludendo con la grafica finale. Per completezza ed ordine, elenchiamo qua le tecnologie software utilizzate:

- **Arduino**, per la configurazione del sensore.
- **mySQL**, il famoso linguaggio di programmazione per Database; nello specifico sono state usate le librerie Flask-SQLAlchemy ed SQLAlchemy.
- **SQLite**, per la creazione del Database.
- **Json**, per lo scambio di dati Client/Server.
- **Flask**, un framework in Python per la creazione di pagine web, sul modello MVC.
- **HTML5**, **CSS**, **Bootstrap**, per lo stile e l'impaginazione finale dell'interfaccia grafica.
- **Jinja2**, template engine per Python.
- **JQuery**, per rendere interattiva la pagina web.

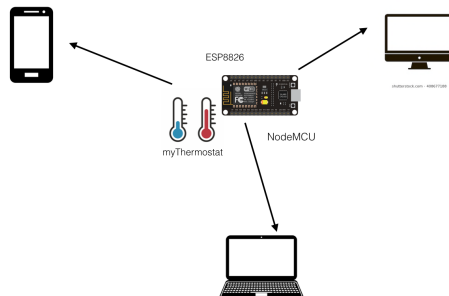


Figura 2: Comunicazione myThermostat.

3.1 Programmazione ed implementazione

In questa sotto sezione vengono descritte nel dettaglio le strategie implementative per poi passare all'interfaccia grafica per l'utente.

3.1.1 Arduino e la comunicazione WiFi

Una volta fatti i collegamenti, attraverso Arduino è possibile collegarsi ad una rete WiFi locale, grazie al modulo ESP8266 montato sul NodeMcu e alla libreria associata. Ad ogni sensore viene associato un IP statico, in modo da rendere più semplice la comunicazione con il Client e la loro connessione e scambio di dati. Dal server corrispondente all'IP associato al sensore, è possibile leggere i dati di temperatura ed umidità comunicati; qui entrano in gioco Python e Json.

3.1.2 Python e Json: collezione dei dati dei sensori.

Per gestire i dati letti dal sensore viene utilizzato Python, dove vengono usate la libreria urllib3, modulo che fornisce un'interfaccia di alto livello per estrarre dati attraverso il World Wide Web, e la tecnologia Json. In particolare, dal momento che abbiamo la necessità di leggere i dati collezionati da più sensori, la funzione PoolManager mantiene in ordine le sessioni di comunicazioni HTTP tra Client e Server. Json, dall'altro lato, decodifica in UTF-8 i dati letti dal sensore e attraverso la struttura del dizionario (basato sulle chiavi "Temp" ed "Humidity", etichette derivanti dal programma Arduino), rende più facile la fase di memorizzazione degli stessi all'interno di un opportuno Database, che è oggetto della successiva fase di sviluppo che andiamo a trattare.

3.2 Database

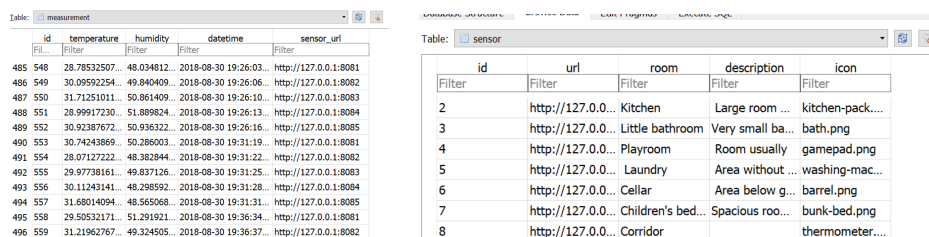
Per avere sotto controllo le temperature e le umidità lette dai sensori, abbiamo integrato la nostra applicazione anche con un database, costruito con SQLite, libreria software scritta in C. La base di dati è costituita da due tabelle:

Measurements, che è provvista dei seguenti attributi:

- ID, numero identificativo del sensore.
- TEMP, temperatura misurata.
- HUM, umidità misurata.
- TIME, data e ora della misurazione.
- IP, indirizzo associato al sensore.

Sensors, che è provvista dei seguenti attributi:

- ROOM, che indica la stanza in cui è posizionato il sensore.
- URL, indirizzo associato al sensore.
- DESCRIPTION, che è una breve descrizione del sensore.
- ICON, icona grafica rappresentante il tipo di sensore.



The image shows two screenshots of a SQLite database viewer. The left screenshot displays the 'measurement' table, which contains columns for id, temperature, humidity, datetime, and sensor_url. The right screenshot displays the 'sensor' table, which contains columns for id, url, room, description, and icon. Both tables have filter buttons above their respective column headers.

id	temperature	humidity	datetime	sensor_url
485	28.78532507...	48.034812...	2018-08-30 19:26:03...	http://127.0.0.1:8081
486	30.09592254...	49.840409...	2018-08-30 19:26:06...	http://127.0.0.1:8082
487	31.71251011...	50.861409...	2018-08-30 19:26:10...	http://127.0.0.1:8083
488	28.99917230...	51.889824...	2018-08-30 19:26:13...	http://127.0.0.1:8084
489	30.92387672...	50.936322...	2018-08-30 19:26:16...	http://127.0.0.1:8085
490	30.74243869...	50.286003...	2018-08-30 19:31:19...	http://127.0.0.1:8081
491	28.07127222...	48.382844...	2018-08-30 19:31:22...	http://127.0.0.1:8082
492	29.97738161...	49.837126...	2018-08-30 19:31:25...	http://127.0.0.1:8083
493	30.11243141...	48.298592...	2018-08-30 19:31:28...	http://127.0.0.1:8084
494	31.68014094...	48.565068...	2018-08-30 19:31:31...	http://127.0.0.1:8085
495	29.50532171...	51.291921...	2018-08-30 19:36:34...	http://127.0.0.1:8081
496	31.21962767...	49.324505...	2018-08-30 19:36:37...	http://127.0.0.1:8082

id	url	room	description	icon
2	http://127.0.0...	Kitchen	Large room ...	kitchen-pack...
3	http://127.0.0...	Little bathroom	Very small ba...	bath.png
4	http://127.0.0...	Playroom	Room usually	gamepad.png
5	http://127.0.0...	Laundry	Area without ...	washing-mac...
6	http://127.0.0...	Cellar	Area below g...	barrel.png
7	http://127.0.0...	Children's bed...	Spacious roo...	bunk-bed.png
8	http://127.0.0...	Corridor		thermometer...

Figura 3: Database.

Le funzionalità delle due tabelle sono evidenti: nella tabella **Measurements** sono raccolti i dati di tutti i sensori che sono collegati alla rete locale; ogni lettura è associata ad un ID identificativo e, soprattutto, ad un IP specifico che funziona da chiave primaria. Nella tabella **Sensors** sono memorizzati tutti i sensori che sono attivi. I dati memorizzati costituiranno lo storico delle misurazioni eseguite dei sensori, e saranno disponibili sulla pagina web associata all'applicazione che stiamo descrivendo. In seguito, parliamo, quindi, di come è stato integrato il Database con la WebPage sviluppata e come è stata ideata quest'ultima.

3.2.1 WebPage

La WebPage è stata pensata per aggiungere, eliminare, leggere i sensori che sono presenti in abitazioni o uffici, basandosi proprio sul paradigma dell'IoT. Per sviluppare la pagina web, è stato utilizzato Flask, un microframework leggero ed estensibile basato su Python. Con Flask è stato possibile integrare il Database di cui abbiamo parlato sopra, grazie alle librerie SQLAlchemy e Flask-SQLAlchemy, che hanno reso molto semplice la gestione delle queries, grazie alla tecnica di programmazione associata basata sull'**ORM**. Per ORM (Object-Relational-Mapping), si intende un metodo di programmazione che rende possibile trattare dei modelli relazionali come degli oggetti della Object Oriented Programming, e quindi renderli molto più portabili e gestibili: infatti, una tabella del Database può essere intesa come un oggetto, con tutti i vantaggi del caso.

3.2.2 MVC: Model View Controller

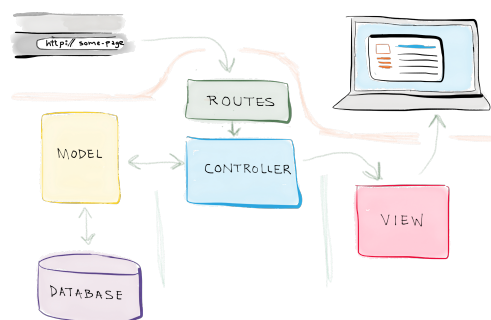


Figura 4: L'architettura Model-View-Controller.

Prima di proseguire, è utile approfondire l'architettura MVC. L'architettura MVC è stata pensata per la gestione di applicazioni Web, riconoscendo tre componenti principali:

- **Model:** è il modello seguito dall'applicazione per gestire le richieste provenienti dagli utenti;
- **View:** è l'oggetto finale che viene presentato all'utente in seguito alla richiesta.
- **Controller:** è l'interfaccia tra Model e View. Infatti è proprio il Controller che, seguendo il Model e basandosi sulle richieste degli utenti, è delegato a costruire la View.

Un componente non citato nel nome dell'architettura, ma comunque molto importante, è costituito dalle **Routes** della WebPage, la mappatura delle varie pagine, i percorsi che vengono seguiti dal Controller per rendere le diverse Views all'User.

Vediamo adesso brevemente qual è il naturale percorso che viene seguito da una applicazione partendo dalla richiesta utente al servizio finale della stessa.

- L'utente compie una HTTP Request inserendo una URL nel Browser;
- Una volta fatta la richiesta, si segue la relativa Route che ci aspettiamo, ovviamente, esista;
- In caso positivo, il Controller prende in mano le redini della situazione e, basandosi sul Model, prende dal Database tutte le informazioni necessarie a servire la richiesta dell'User;
- A questo punto, i dati memorizzati in una struttura come dizionario o una lista, vengono processati in HTML e mostrati all'utente.

Nella nostra Web Application, questi passaggi vengono tutti gestiti da Flask: quando l'utente fa una richiesta, Flask la analizza e la serve come descritto, interfacciandosi con il Database, fino ad utilizzare Jinja per le Views finali.

4 User Interface and Design

Una volta creata la pagina grazie a Flask ed averla appoggiata al Database, è finalmente stato possibile lavorare sull'interfaccia grafica per l'utente. Chiaramente, è stato necessario costruire una pagina che fosse responsiva, interattiva e adattiva. Inoltre, l'idea era quella di creare una HomePage, dove fosse possibile gestire i sensori (eliminarli e aggiungerli), collegandosi ad un'altra pagina appositamente disegnata, e una pagina per la lettura del sensore, ovvero che mostrasse un grafico dello storico dei dati raccolti e la situazione attuale di temperatura ed umidità.

Vediamo quali sono state le tecnologie usate per creare quanto detto sopra.

- **Bootstrap.** E' un insieme di strumenti liberi che permette di creare pagine web in maniera molto semplice. Nel nostro caso è stato utilizzato per avere un tema fisso, gestito e modificato con opportuni file CSS ed HTML, e soprattutto responsivo, in grado di adattarsi a qualsiasi schermo e a qualsiasi risoluzione.
- **Jinja2,** è un motore di template attrezzato per il linguaggio Python e per questo il più utilizzato da Python. Viene utilizzato da framework per applicazioni web, proprio come Flask. La parte più potente di Jinja è l'ereditarietà del template che permette di costruire uno scheletro di base del template, contenente tutti gli elementi comuni e di definire blocchi che le pagine "figlio" possono ereditare. Volendo costruire un sito di più pagine graficamente basate sullo stesso tema, Jinja2 risulta essere uno strumento adatto.
- **Javascript.** Questa tecnologia è stata utilizzata per rappresentare la temperatura e l'umidità attuale, con dei calibratori, e lo storico dei dati, attraverso un grafico, come detto in precedenza.

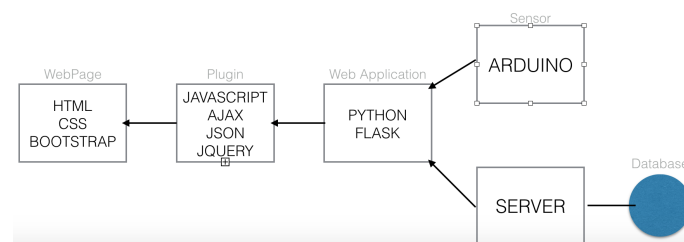


Figura 5: L'applicazione descritta.

L'immagine sopra descrive tutto il lavoro svolto dai singoli elementi e la loro composizione; adesso ci concentriamo proprio sulla **WebPage**, analizzando proprio HTML e CSS utilizzati, oltre le funzionalità di Bootstrap; questo aspetto è molto importante perché ovviamente un'applicazione, anche se complessa, deve essere facilmente accessibile da qualsiasi utente. Il sito web associato all'applicazione myThermostat si basa sulle seguenti pagine:

- **Homepage.** La pagina principale, dove è possibile avere una visione di insieme di tutti i sensori attivi, aggiungerne altri od eliminarli dalla lista.
- **Add sensor.** Cliccando sul bottone "+", si apre la pagina con la form relativa, dove si chiede la stanza, la URL e una breve descrizione del nuovo dispositivo; terminata la procedura, si aggiorna la tabella del database e si viene reindirizzati alla Home.
- **View.** Dalla Home si accede a "View", che apre la Dashboard del sensore.
- **Graphics.** Dalla pagina di View, è possibile vedere la situazione del giorno, della settimana, del mese passato o di un periodo a scelta tramite un grafico.
- **Overview.** E' disponibile anche una rapida panoramica di tutta la rete myThermostat attraverso la pagina "Overview".

Finalmente è possibile descrivere l'applicazione facendo unicamente riferimento all'interfaccia grafica ed andando ad analizzare i singoli componenti e le singole pagine, astraendo da tutta la parte implementativa finora descritta. myThermostat è una applicazione pensata per essere facile da utilizzare, molto intuitiva e con la possibilità di indicare facilmente all'utente tutti gli errori con cui è possibile scontrarsi, da un inserimento di un IP sbagliato del sensore, al tentativo di inserire un sensore già esistente. Nelle sezioni successive sarà possibile apprezzare queste strategie implementative direttamente per via grafica.

4.1 Home

Una volta acceduto il sito, la Home si presenta così (per comodità si mostrano le schermate da smartphone):

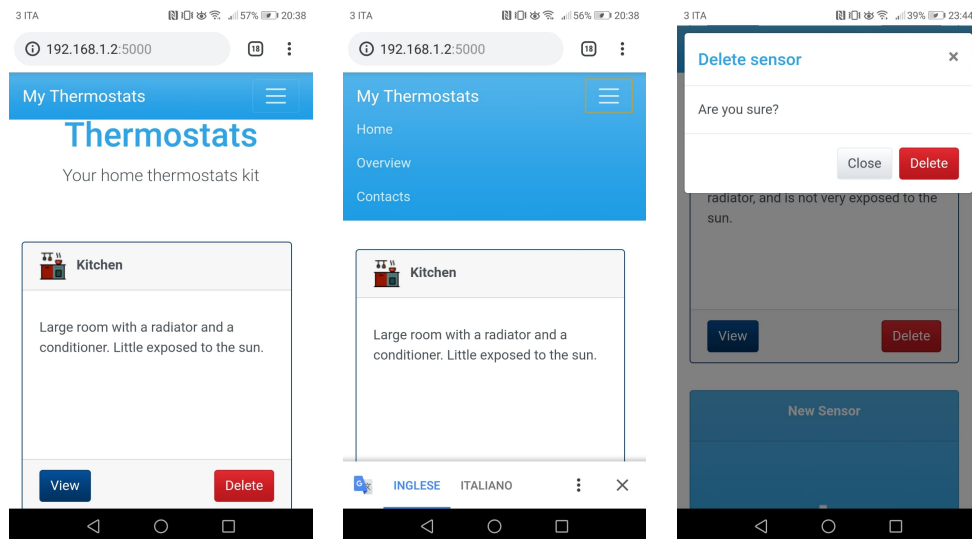


Figura 6: Home page.

Come si può notare, è possibile fare le seguenti azioni dalla pagina principale della Web Application myThermostat:

- Vedere un sensore: attraverso il pulsante "View" è possibile accedere alla Dashboard giornaliera che mostra tutte le ultime misurazioni di temperatura ed umidità;
- Eliminare un sensore, attraverso il tasto Delete: nel caso questo venga premuto, apparirà un alert che permette all'utente di confermare o annullare la scelta precedente di eliminazione.
- Aprire il menù a tendina per accedere ai Contatti degli sviluppatori (mail e cellulare) o alla pagina Overview, che dà una visione generale dell'edificio in cui è installata la rete myThermostat.

4.2 Add Sensor

Sempre dalla Home, attraverso il pulsante ”+” è possibile aggiungere alla rete un nuovo sensore.

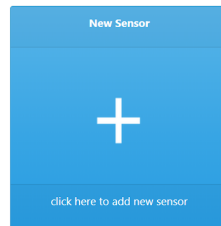


Figura 7: Tasto ”Add Sensor”.

Toccando o cliccando sul pulsante sopra mostrato sarà sufficiente compilare la form mostrata in figura per aggiungere un nuovo dispositivo alla rete myThermostat, indicando la stanza dove il nuovo termostato verrà aggiunto, l’indirizzo IP associato e, se si vuole, anche una breve descrizione, oltre ad una icona. Qui sotto viene illustrata la pagina ”Add Sensor”:

Figura 8: Schermate con la form per aggiungere sensori.

4.3 Dashboard

Una volta aggiunto un nuovo sensore alla rete, è possibile aprire due pagine diverse: una, chiamata Dashboard, dove sono presenti le temperature ed umidità medie e attuali registrate dal sensore selezionato, un'altra, chiamata "Graphic", che vedremo in seguito.

Qui sotto viene presentato un dettaglio della pagina "Today Dashboard":

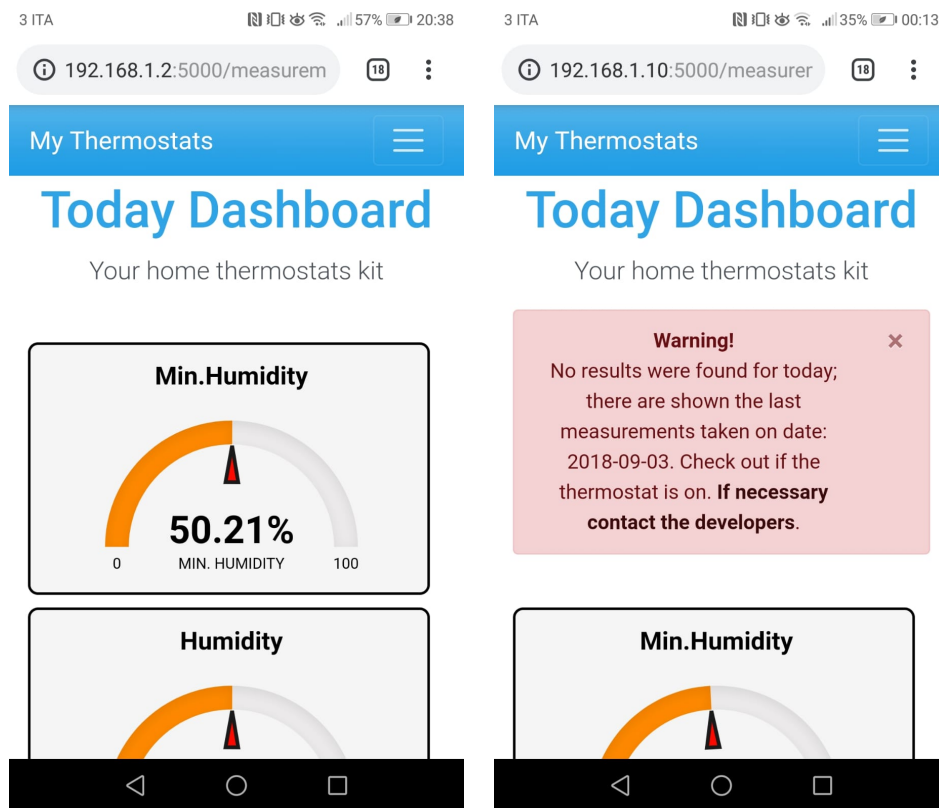


Figura 9: Overview.

Sotto sono anche presente una data ed un orario che evidenziano quando è avvenuta l'ultima misurazione del sensore. A proposito di questo, è necessario dire che, nel caso in cui non venga effettuata una misurazione giornaliera, nella pagina in questione viene presentata la Dashboard relativa all'ultima misurazione avvenuta da parte di quel determinato sensore.

4.4 Graphic

Dalla Dashboard si può accedere alla pagina dei grafici contenenti lo storico dei dati sopra citata, grazie al bottone "Show relative Graphic".

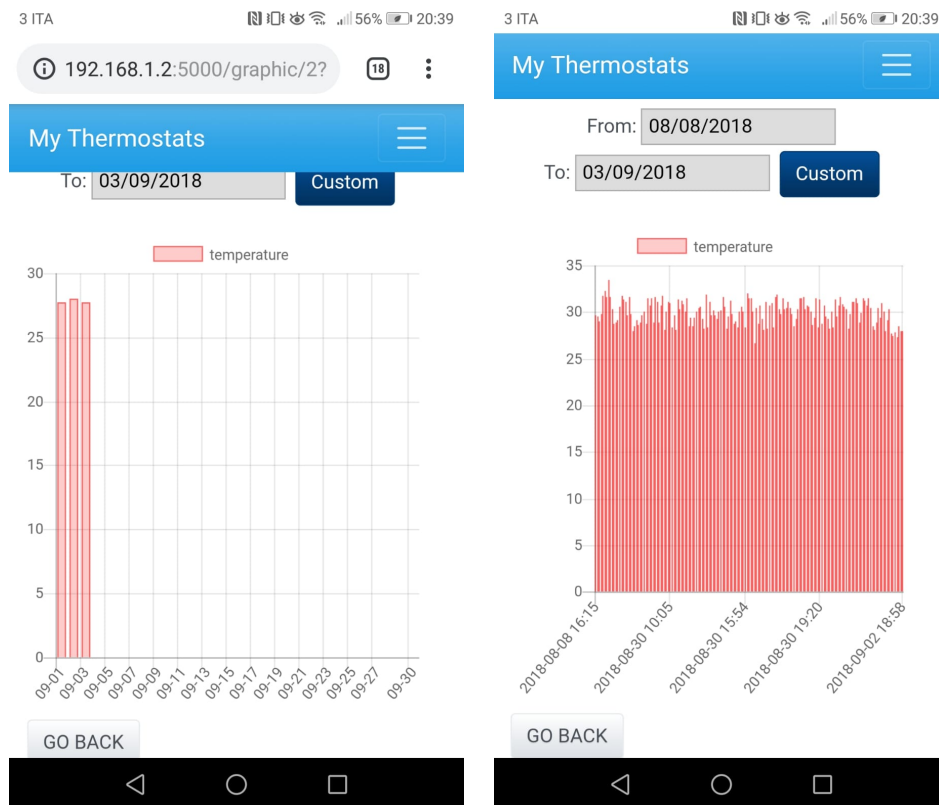


Figura 10: Schermata con esempio di grafico.

Alla pagina dei grafici si accede attraverso il pulsante "Show relative graphic" che si trova nella Dashboard di un sensore.

In questa pagina è possibile selezionare un grafico che mostra i dati raccolti per il giorno corrente, la settimana, il mese e l'anno o, alternativamente, c'è anche la possibilità di selezionare un periodo a scelta da parte dell'utente con l'apposita form.

Se in un periodo selezionato non sono state effettuate misure dal sensore scelto, allora verrà mostrato il grafico che considererà i dati a partire dall'ultima misurazione fino all'ultima disponibile nel periodo indicato. I sensori sono programmati per fare più misure nell'arco della giornata; per ogni giornata viene fatta la media sui dati raccolti nelle ore; per settimana, mese e anno, sui giorni.

4.5 Overview

La Dashboard e la pagina dei grafici permettono di vedere informazioni relativi a dati raccolti da un solo termostato che si trova nella rete myThermostat; nel caso in cui nell'ambiente siano presenti più termostati, è anche possibile visualizzare una "Overview", ovvero una mappa che indica temperature ed umidità in tempo reale di tutte le stanze coinvolte nelle misurazioni; ad ogni stanza è associato un colore relativo alla misurazione eseguita (per esempio, se nella stanza è presente una temperatura ritenuta adeguata si colora di arancione).

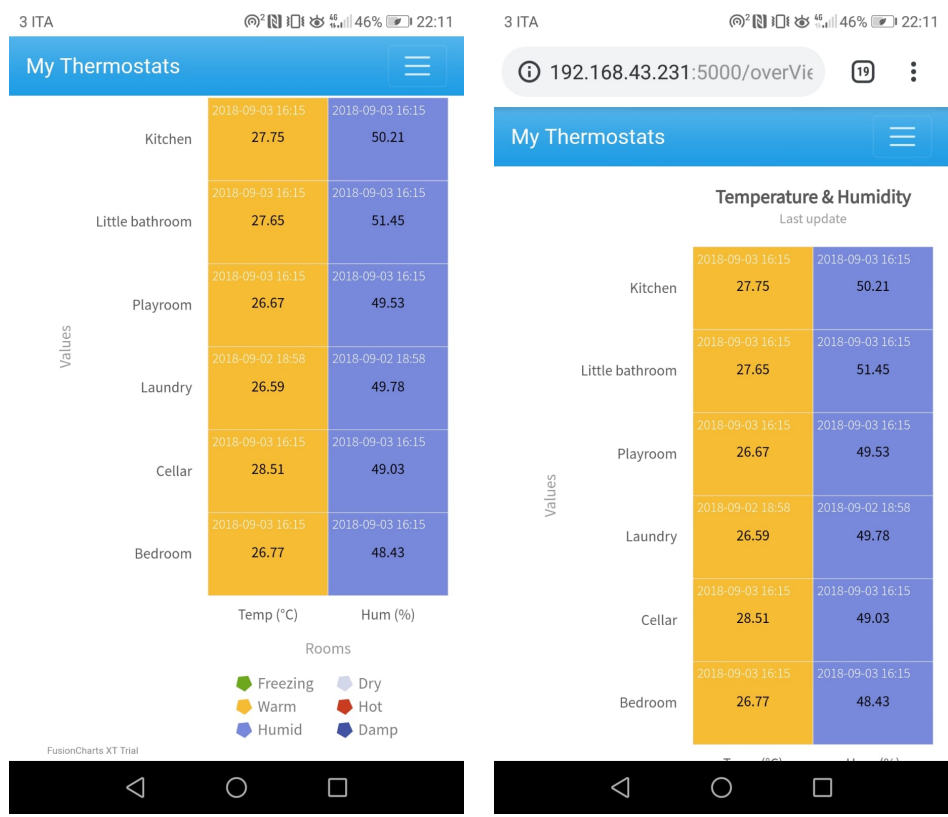


Figura 11: Overview.

5 Conclusioni

- myThermostat è stato testato con Windows 10 e iOS El Capitan; per quanto riguarda le versioni per smartphone, i test sono stati eseguiti con un Samsung e un OnePlus5.
- Il design della pagina web è stato scelto utilizzando un tema, per BootStrap, presente sul sito bootswatch.com, per la precisione il tema "Cerulean".
- L'utente può accedere a myThermostat da Smartphone, laptop o qualsiasi altro device dotato di connessione internet senza fili assicurandosi di essere connesso alla stessa rete del NodeMCU e digitando nel browser il corretto indirizzo IP della pagina relativa all'applicazione.
- La raccolta dei dati da parte di Flask sul sensore DHT11 avviene automaticamente ogni 60 minuti grazie ad uno script presente nel file `Sensor.py`, componente dell'applicazione.
- Le librerie principali usate sono: SQLAlchemy (per il Database, la sua scrittura e lettura); ESP8226.h e DHT11.h per la configurazione del modulo Wi Fi del NodeMCU e del sensore termico.