Termostato Smart

Damiano Bellucci

Sommario

Il termostato smart è un concetto in voga nel panorama internet of things. I relativi prodotti in commercio hanno però un problema principale: la gestione del dispositivo avviene tramite server della casa produttrice ed all'utente viene richiesta una registrazione di un account, che associa la sua identità al dispositivo. In questo progetto è stato quindi realizzato un termostato smart, che non necessita di servizi terze parti (almeno all'interno della rete locale) e che può funzionare anche senza la parte smart (ad esempio in caso di mancata connessione) in quanto l'interfaccia hardware mette a disposizione tutte le operazioni possibili.

1 Introduzione

L'idea di questo progetto è stata quella di creare un termostato casalingo che di base può essere utilizzato anche senza la parte smart, quindi la mancata connessione non pregiudica in nessun modo l'utilizzo del dispositivo, ma lo rende solo non utilizzabile da remoto.

Le operazioni possibili sono quelle di settaggio della temperatura ambientale desiderata, le ore del giorno in cui il termostato può attivarsi (parametri orari) ed il cambio d'orario.

Il progetto consiste di due parti principali hardware: il termostato ed un server locale per la gestione della parte smart.

Nelle sezioni 2 e 3 vengono spiegati nel dettaglio hardware e software rispettivamente del termostato e del server locale, mentre nel capitolo 4 si parla delle conclusioni e dei possibili sviluppi futuri di questo progetto.

In figura 1 viene mostrato uno schematizzazione del progetto per quanto riguarda la parte smart: il server locale ed il termostato sono nella stessa rete locale. Il server locale ha al suo interno un modulo che espone delle API che permettono ai clients all'interno della rete locale di interfacciarsi con il termostato, non direttamente ma tramite il server locale. Per permettere l'utilizzo delle funzioni smart anche al di fuori della rete locale, il server locale ospita la gestione di un Bot Telegram, che permette ai clients di interagire con il termostato tramite chat direttamente dall'app Telegram.

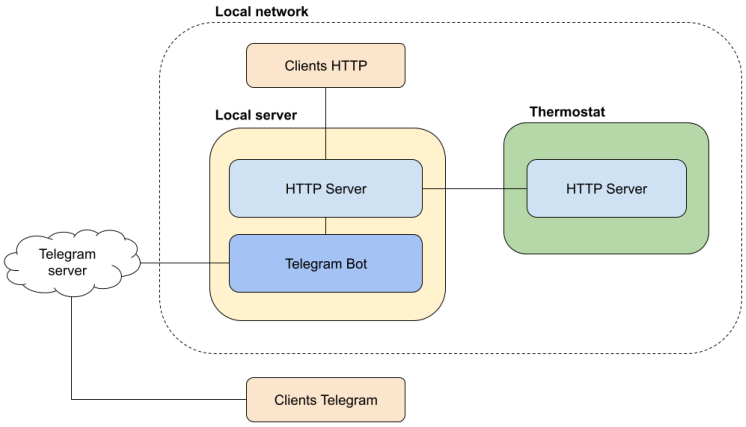

Figura 1: Schema del progetto, parte smart

2 Termostato

In questa sezione vengono spiegate nel dettaglio le scelte effettuate per interfaccia, l'hardware e software relative al termostato.

2.1 Interfaccia

L'interfaccia del termostato consiste in un display e quattro pulsanti. In caso di illuminazione dello schermo data dalla pressione di uno dei tasti, oppure quando si esce dai menù di impostazione dei parametri orari/temperatura ambientale/orario, viene mostrato sul display un riepilogo dello stato attuale del termostato.

Ci sono tre possibili stati raggiungibili tramite pulsanti: riepilogo, configurazione parametri frequenti (parametri orari e temperatura ambientale) e configurazione orario.

Un pulsante menù è adibito all'accesso della configurazione dei parametri con settaggio "frequente", cioè parametri orari e parametro della temperatura ambientale desiderata. Ci sono poi altri due tasti che servono per aumentare/diminuire i valori una volta che si è sul menù di modifica di un settaggio. Questi due tasti, se premuti assieme mentre si è nella stato di riepilogo, permettono di accedere al menù di settaggio dell'orario del termostato. Un ultimo tasto permette di spostarsi a destra/sinistra quando si è nella configurazione dell'orario.

Per evitare sovrapposizioni di configurazione dei settaggi, nel momento in cui il termostato è in uno stato che non è quello di riepilogo, non è momentaneamente possibile utilizzare le funzioni smart per la modifica dei parametri, ma solo le funzioni di lettura di questi ultimi.

Al display viene tolta la retroilluminazione se non ci sono più interazioni per 10 secondi e si torna allo stato di riepilogo, questo per evitare che l'utente rimanga su uno stato di menù che comporta la disattivazione della parte smart, oltre che per evitare che il display rimanga acceso in maniera indesiderata.

Nelle figure 3 4 5 6 sono presentate i vari stati in cui si può trovare il display (e come è possibile arrivarci premendo i pulanti, in base allo schema in figura 7), di seguito elencanti:

- Figura 3, schermata di riepilogo: questo stato è quello di default e quello in cui si ritorna quando si esce dai menù di impostazione dei parametri. Vengono mostrati a partire dalla riga in alto da sinistra l' orario, stato on/off (circuito chiuso/aperto della caldaia) e temperatura corrente, mentre nella nella riga sottostante vengono mostrati i parametri orari: ogni rettangolo corrisponde ad un'ora della giornata, in maniera ordinata, da 0 a 23, se il rettangolo è spostato verso l'alto vuol dire che il parametro orario è su on e quindi se in quell'orario corrispondente la temperatura ambientale è sotto quella desiderata il circuito della caldaia viene chiuso

- Figura 4, menù di impostazione dei parametri orari: raggiungibile dallo stato di riepilogo premendo il bottone 1. Da questo stato è possibile tenere premuto il bottone 2 per scorrere un cursore che punta ad un singolo parametro orario, mentre per cambiare lo stato del parametro orario si dovrà premere il bottone 3 mentre è attivo il cursore

- Figura 5, menù di impostazione della temperatura desiderata: raggiungibile dallo stato di riepilogo premendo due volte il bottone 1. Viene mostrato il valore di temperatura desiderata. E' possibile cambiare il valore aumentandolo premendo il bottone 2 oppure diminuendolo premendo il bottone 3.

- Figura 6, menù di impostazione orario: raggiungibile dallo stato di riepilogo premendo i bottoni 2 e 3 contemporaneamente. Viene mostrato l'orario attuale del termostato (fornito dal modulo RTC) secondo la sintassi ore:minuti, giorno del mese, mese, anno. E' possibile shiftare l'orario di ore, minuti, giorni selezionando uno dei tre tramite il cursore sottostate, che può essere fatto scorrere premendo il bottone 4, mentre per cambiare il valore bottone 2 per aumentarlo e bottone 3 per diminuirlo.

In figura 2 viene mostrata una schematizzazione dei possibili flussi di esecuzione tra i vari stati del display. Le etichette numeriche sulle sulle frecce di transizione corrispondono al numero dei pulsanti in figura 7.

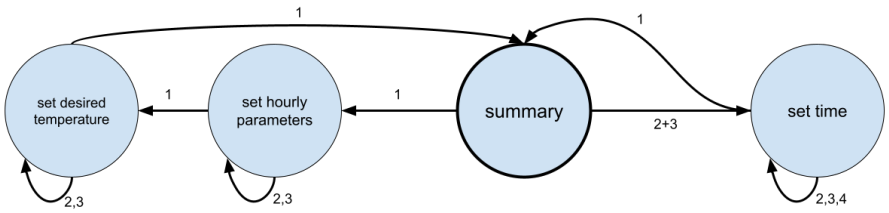

Figura 2: Schermata del display: schema stati possibili e movimenti in base ai pulsanti premuti

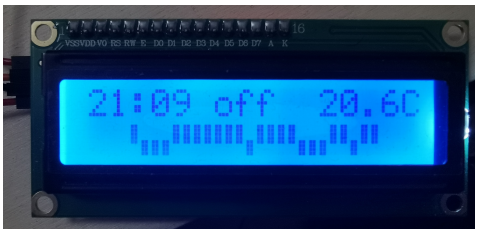

Figura 3: Schermata del display: riepilogo

Figura 4: Schermata del display: settaggio parametri orari

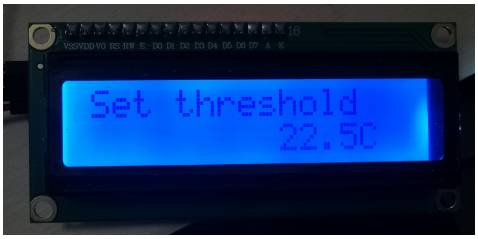

Figura 5: Schermata del display: settaggio parametro temperatura desiderata

Figura 6: Schermata del display: settaggio dell'orario

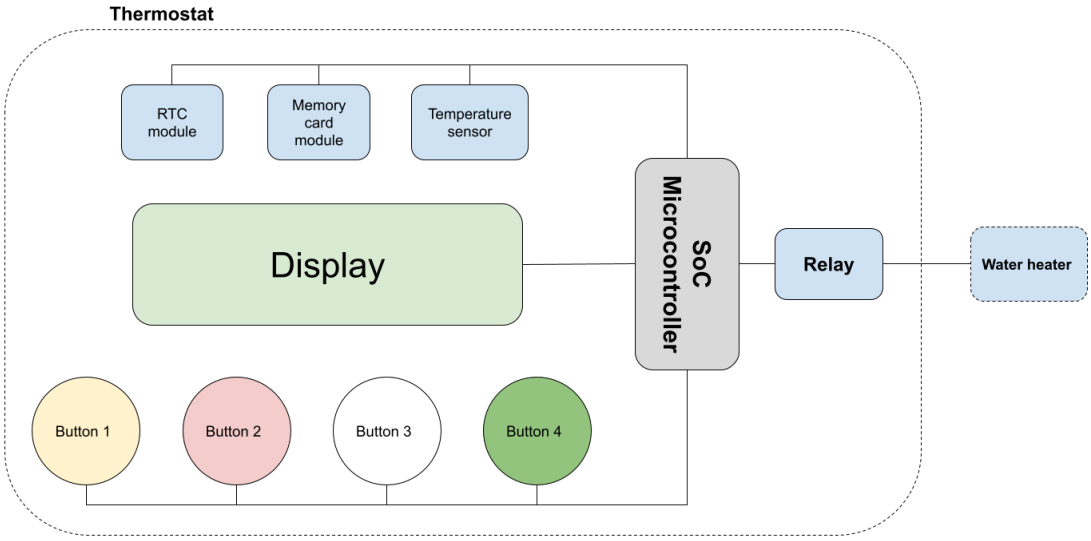

Figura 7: Schema del progetto: termostato

2.2 Hardware

In figura 7 una schematizzazione dell'hardware del termostato e di seguito la lista dei componenti hardware utilizzati, con le motivazioni che hanno portato alla scelta di questi componenti:

- SoC microcontroller: è stata utilizzata una ESP32, equivalente per le sue caratteristiche alla serie Arduino Nano IoT e con stesso ambiente di sviluppo e librerie. E' stata scelta questa scheda in quanto ha modulo WiFi integrato e per le sue dimensioni ridotte, che si adattano bene ad un dispositivo come un termostato.
- Modulo memory card: comprende una microSD card reader SPI per l'interfacciamento con il microcontrollore ed una scheda microSD. Questo modulo ha il compito di salvare le impostazioni di temperatura desiderata e parametri orari, per mantenere le impostazioni di ultimo settaggio nel caso di mancata alimentazione del termostato. In alternativa a questo modulo si sarebbe potuto anche usare la memoria interna del microcontrollore, ma non è una buona scelta a lungo termine, in quanto le memorie interne dei microcontrollori hanno una vita breve in termini di operazioni di lettura e scrittura.
- RTC module: comprende un un RTC module DS1307 I2C con batteria. Questo modulo è adibito a tenere l'orario del termostato (secondi, ore, giorni, mesi, anni) anche in caso di mancata alimentazione del microcontrollore grazie alla sua alimentazione indipendente a batteria. La libreria per la sua gestione fornisce API che permettono di impostare l'orario e di eseguire shift temporali.
- Temperature sensor: il sensore di temperatura è un DHT22, che è un sensore di temperatura ed umidità digitale. La libreria per la sua gestione fornisce le API per ottenere informazioni su temperatura ed umidità. In questo progetto è stata presa in considerazione solo la temperatura.
- Display: il display è un I2C 16x2. La libreria usata per la sua gestione è la LiquidCrystal I2C.
- Relay: il relay è stato utilizzato per fare da switch circuito aperto/chiuso della caldaia a seconda del segnale in uscita dal microcontroller. Il relay è normalmente aperto, quindi se non arriva il segnale dal microcontrollore il circuito della caldaia non viene chiuso.
- Pulsanti e relative resistenze: sono stati utilizzati 4 interruttori a singolo polo a 4 contatti, normalmente aperti, con relative resistenze di pull-up da 1k ohm.

2.3 Software

L'intero software è stato sviluppato tramite Arduino IDE.

Nella parte di setup vengono caricati dalla memory card i parametri di settaggio (parametri orari e temperatura ambientale desiderata), effettuata la connessione alla rete ed inizializzato l'http async web server per rendere il termostato disponibile alle operazioni smart. Nel caso di mancata connessione, l'esecuzione procede e si ritenta nel loop. Nel caso di perdita di connessione all'interno del loop viene ritentata la connessione ad ogni ciclo.

Ogni volta che vengono modificati i parametri di settaggio frequenti, sia dal termostato sia tramite le operazioni smart, vengono scritti in memory card i nuovi settaggi (i file relativi ai vecchi parametri vengono sostituiti dai nuovi). Nelle fasi di lettura di questi parametri invece, per evitare l'overhead di lettura in memoria, questi vengono direttamente letti dalle variabili dinamiche. Per quanto riguarda la modifica dell'orario, questo viene gestito direttamente dal modulo RTC, che si occupa di mantenere nella sua memoria l'orario.

L'async web server espone due API: una di GET dall'endpoint /currentstate ed una di POST dall'endpoint /setparameters. Questi endopoint verranno raggiunti solo dal server locale che fa da bridge tra il termostato e l'utente che utilizza funzioni smart.

Dall'endpoint /currentstate si possono recuperare le informazioni correnti del termostato, in base ai parametri di query della richiesta http GET, queste informazioni possono essere richieste singolarmente o insieme. Le informazioni reperibili sono le seguenti:

- temperatura corrente
- temperatura ambientale desiderata
- orario
- parametri orari
- stato on/off (circuito della caldaia chiuso o aperto)

Dall'endpoint /setparameters si possono aggiornare tutti i parametri di settaggio del termostato, in base ai parametri nel body della richiesta http POST. Anche in questo caso, l'API permette di poter configurare più parametri assieme (da uno a tutti). I parametri modificabili sono i seguenti:

- temperatura ambientale desiderata
- shift dell'orario: si possono aggiungere/togliere secondi, minuti, ore, giorni
- parametri orari: per ogni ora del giorno si può impostare on/off

Operazione di GET è disponibile sempre in caso di connessione, mentre l'operazione di POST è disponibile solo quando il termostato è nello stato di riepilogo, per evitare sovrapposizioni delle modifiche nel settaggio dei parametri.

3 Server locale

Il server locale è un bridge tra le operazioni smart del termostato ed il termostato stesso. E' un raspberry pi con sistema operativo raspbian headless, raggiungibile dall'esterno tramite protocollo SSH.

Il raspberry fa girare un http server che si comporta da proxy, gestisce le richieste per n eventuali clients, ma è l'unico poi ad interagire con il termostato.

Inoltre, sul raspberry è in esecuzione il Bot Telegram, che permette di fare interagire i clients, anche al di fuori della rete locale, tramite l'app di messaggistica Telegram in maniera user friendly.

3.1 HTTP server

L'http server, sviluppato in nodejs, fornisce gli stessi endpoint del termostato. Quello che fa però è gestire le richieste controllando la validità di queste, fornire risposte adeguate al client in caso di mancata disponibilità sulla rete del termostato, ecc... Il suo compito è rendere la gestione delle richieste efficiente per il web server nel termostato, che dovrà occuparsi solo di soddisfare le richieste una volta arrivate.

Nelle figure 8 9 10 11 12 13 vengono mostrati degli esempi di richieste agli endpoint /currentstate e /setparameters.

In figura 8 un esempio di richiesta corretta all'endpoint /currenstate, con tutti i parametri di query possibili (è possibile settare a piacimento i parametri di query per ottenere una o più informazioni). In risposta alla richiesta si avrà un json in cui per ogni chiave c'è una informazione richiesta relativa allo stato attuale del termostato:

- temperatura rilevata
- status: con uno indica che il circuito della caldaia è chiuso, zero in caso contrario
- knights: un json che raccoglie il settaggio dei parametri orari, con zero corrispondente ad off ed uno ad on
- threshold: temperatura desiderata
- time: un json che indica per ogni campo le informazioni relative all'orario del termostato, con "hh" che equivale ad ore, "mm" minuti, "ss" secondi, "dd" giorno del mese, "mo" mese, "yy" anno

Come si può notare, lo status è zero anche se la temperatura desiderata (threshold) è sopra la temperatura rilevata. Questo è dato dal fatto che il parametro orario relativo alle ore 16 (orario al momento della richiesta) è impostato su 0, cioè su off.

In figura 9 un esempio di richiesta corretta all'endpoint /setparameters. Nell'esempio sono stati settati solo alcuni parametri, ma è possibile settare uno o più dei seguenti parametri nella stessa richiesta, che sono i seguenti:

- t: temperatura desiderata, cioè la threshold
- 1,...,23: parametri orari, ognuno corrispondente ad un'ora del giorno
- hh: shift delle ore
- mm: shift dei minuti
- ss: shift dei secondi
- dd: shift dei giorni

In questa richiesta viene impostata una temperatura desiderata a 21.5, parametri orari relativi alle 14 ed alle 15 su on mentre su off il parametro orario delle 16, inoltre all'orario corrente de termostato vengono aggiunti 5 minuti.

Nella figura 10 un esempio di richiesta dove il termostato non è connesso alla rete. In questo caso la risposta contiene un codice di errore 503 che corrisponde a "servizio non disponibile".

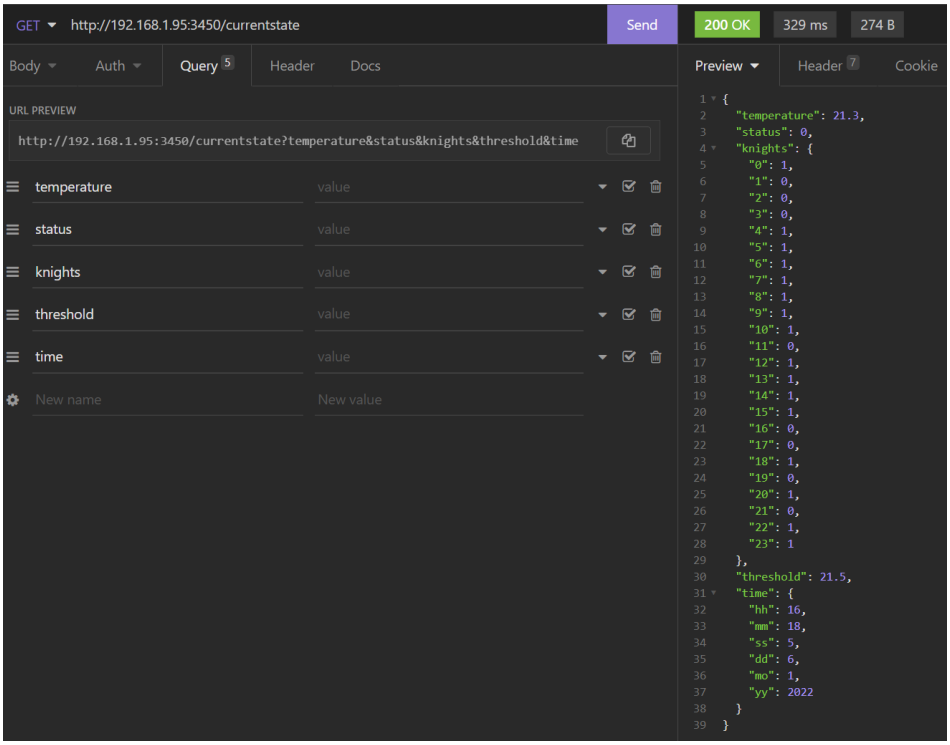

Figura 8: Esempio di GET currentstate

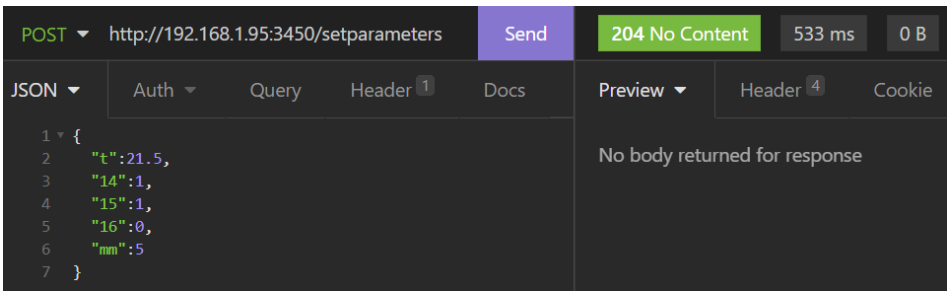

Figura 9: Esempio di POST setparameters

Nella figura 11 un esempio di richiesta post mentre nel termostato si è in un menù per la configurazione de parametri. Anche in questo caso la risposta è un codice di errore 503 con testo dell'errore che indica che il termostato è in uno stato di configurazione dei parametri.

Ogni richiesta http (sia per l'endpoint /currentstate che per l'endpoint /setparameters) viene validata prima di essere inoltrata al termostato. La validazione comprende controllo di vincoli su parametri possibili e sui loro valori corrispondenti. In caso di richiesta non valida, invece di inoltrare la richiesta al termostato, viene data una risposta relativa all'errore commesso nella richiesta. In figure 12 13 degli esempi di richieste mal formulate.

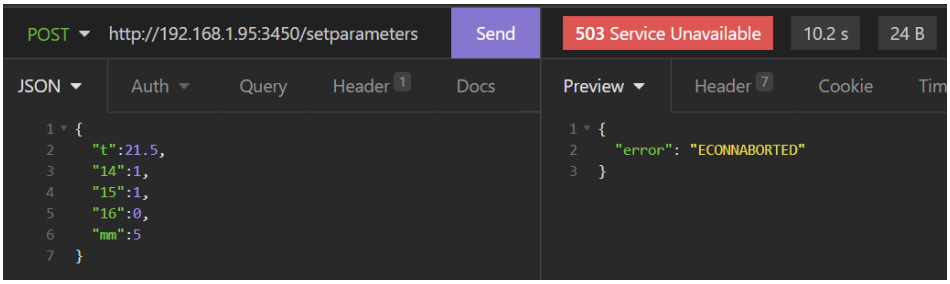

Figura 10: Esempio di richiesta in caso in cui termostato non è connesso alla rete

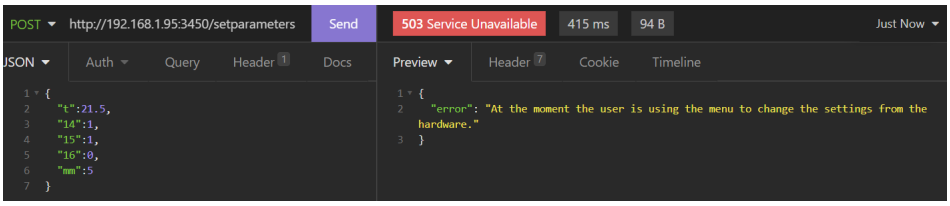

Figura 11: Esempio di POST request contemporanea a termostato in menù in configurazione parametri

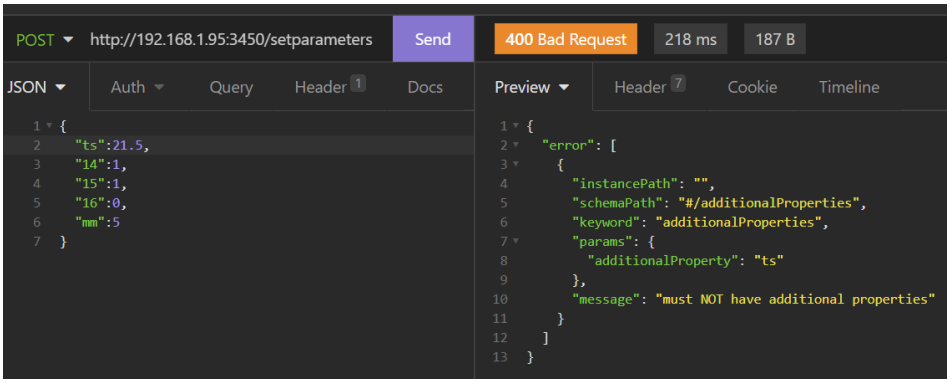

Figura 12: Esempio di POST con errore nella richiesta

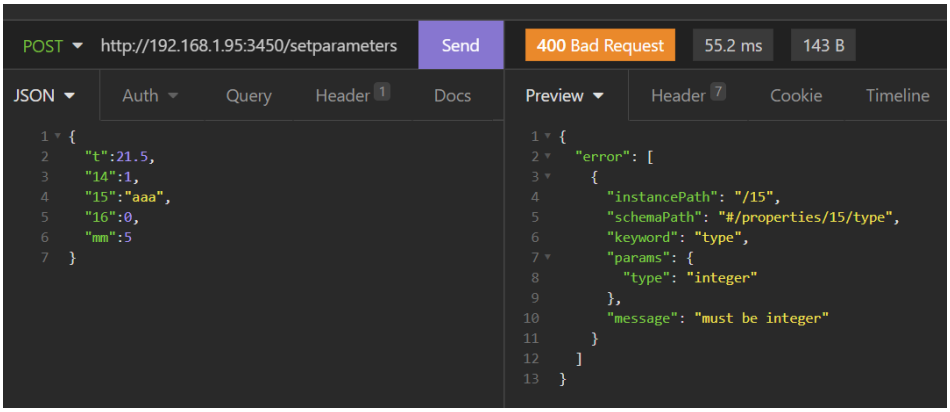

Figura 13: Esempio di POST con errore nella richiesta

3.2 Bot Telegram

Il bot Telegram, sviluppato in nodejs, si interfaccia con l'http server presente nel raspberry pi, utilizzando le sue API. Quello che si può fare tramite bot è equivalente alle funzioni disponibili tramite le API, solo con una interazione passo passo, con i messaggi del bot che suggeriscono all'utente quali sono le operazioni possibili, come farle ed in caso di errore risposte che spiegano con esempi come procedere per far si che la richiesta avvenga con successo.

In figura 14 viene mostrato un esempio di esecuzione del flusso tramite bot per aggiornare temperatura desiderata e aggiornare il parametro temporale di un'ora precisa. Si può notare come inizialmente viene richiesto lo stato generale del termostato, che mostra che lo status è off in quanto la temperatura registrata è sopra la temperatura desiderata. L'utente in seguito setta la temperatura desiderata a 22.5, ma lo stato del termostato rimarrebbe off in quanto si è nell'orario delle 19 ed il parametro orario 19 è su off. L'utente quindi setta il parametro orario 19 su on e si può notare che nella successiva richiesta current state lo stato del termostato è su on, in quanto vengono incontrate le condizioni su temperatura desiderata e parametro temporale.

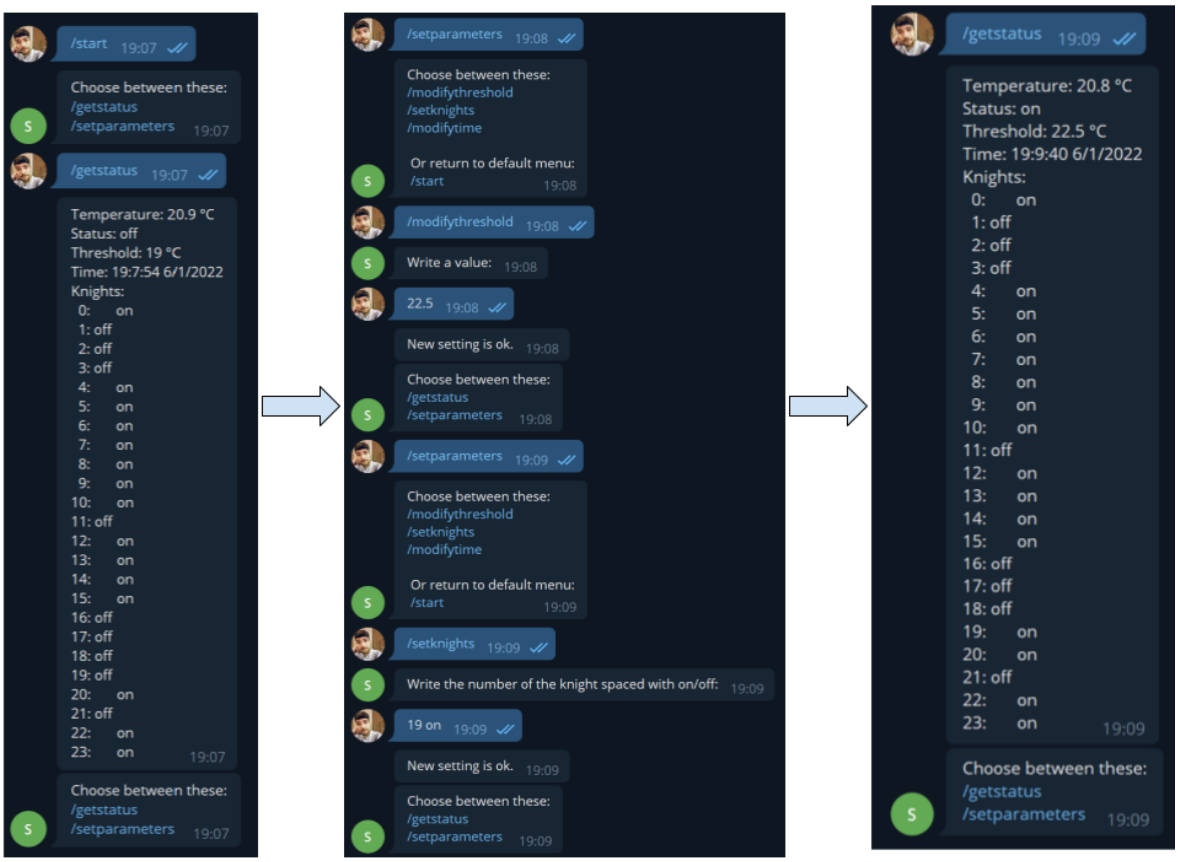

Figura 14: Esempio di flusso di esecuzione tramite bot Telegram

4 Conclusioni e sviluppi futuri

Tramite questo progetto è stato possibile sperimentare la prototipazione di un termostato smart, che sia utilizzabile anche senza connessione e che non richieda servizi esterni per poter funzionare all'interno della rete locale. L'utilizzo della parte smart al di fuori della rete locale fa affidamento ad un servizio esterno, che è Telegram. In questo caso i dati fluiscono nel server Telegram che fa da intermediario. Soluzioni a questo sono possibili, anche se non a costo zero e con una particolare attenzione da porre sulla sicurezza.

Possibili sviluppi futuri di questo progetto possono riguardare l'ampliamento della parte smart, infatti le API messe a disposizione dall'http server nel server locale sono state pensate proprio per poter fornire ai fruitori ampio margine di manovra sul termostato. Si potrebbe creare una piattaforma web ad hoc che comprende dashboard interattiva, possibilità di fare pianificazioni settimanali/mensili, gestione automatica dell'ora solare/legale, un modulo di raccolta e gestione dati per creare grafici di report e grafici in tempo reale ed utilizzare i dati per fare previsioni/studi basati sulle serie di dati temporali relativi alla temperatura.