Corso di RETI 2: Sperimentazioni  
Relazione su progetto

HomeFort

HomeFort è un sistema di gestione del comfort all'interno di un'abitazione.

L'obiettivo preposto è stato quello di creare un sistema automatico di regolazione di alcuni aspetti del comfort casalingo, permettendo l'interazione con l'utente, il quale può impostare i parametri del comfort e gestire direttamente alcuni componenti.

L'utente finale del prodotto ha la facoltà di accendere o spegnere le luci e impostarne la luminosità, può chiedere informazioni sulle condizioni interne della casa (luminosità, umidità, temperatura e movimento) e sulle condizioni meteo.

L'interazione con l'utente è realizzata attraverso un assistente conversazionale che si appoggia su un bot telegram, in modo che si possa gestire il sistema comodamente anche fuori casa, ma anche sul divano senza doversi muovere.

Per quanto riguarda la configurazione del sistema automatico, l'utente può impostare gli orari in cui le luci dovranno essere sempre accese e la temperatura ideale che si vuole mantenere nell'abitazione.

Il sistema di controllo automatico gestisce quindi luci e temperatura interna. Le prime vengono accese nel range d'orario impostato precedentemente dall'utente, ma solamente se in casa viene rilevato del movimento. HomeFort raccoglie informazioni sulla luminosità interna e imposta la luminosità delle luci intorno ad un valore ideale, se richiesto dall'utente attraverso la modalità automatica.

La temperatura viene regolata automaticamente attorno al valore di comfort impostato dall'utente.

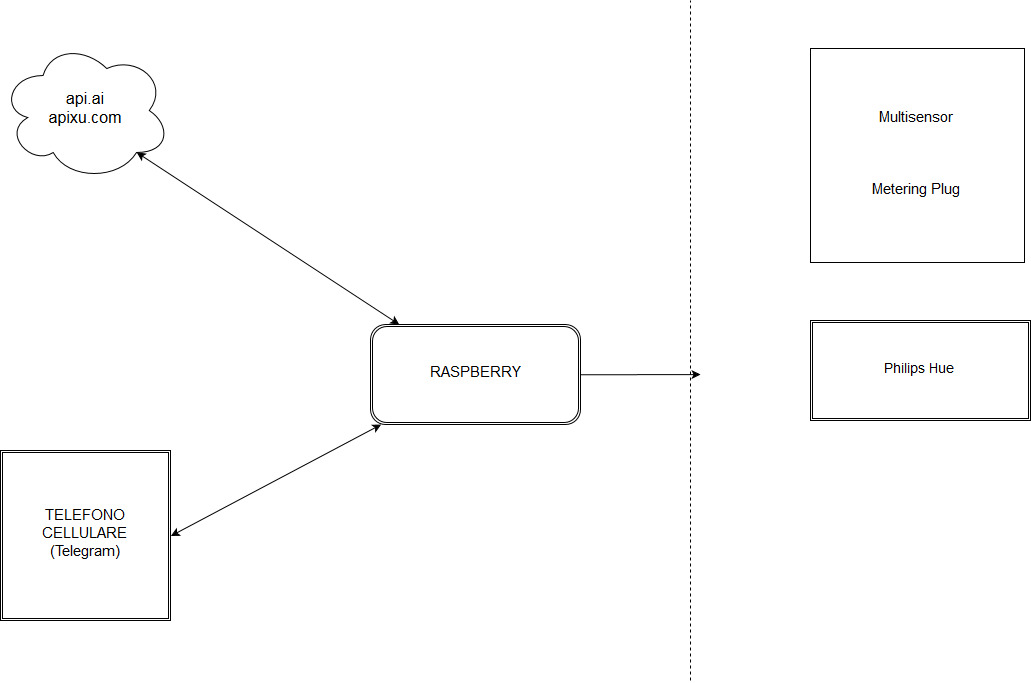
Nel dettaglio, i servizi forniti dal sistema sono:

* Accensione delle luci
* Impostazione della luminosità delle luci
* Spegnimento luci
* Impostazione di range orari per la gestione automatica delle luci
* Richiesta previsioni meteo su 3 giorni
* Richiesta per le condizioni meteo attuali
* Impostazione della temperatura ideale
* Possibilità di attivare/disattivare la gestione automatica delle luci (orario e luminosità)
* Possibilità di controllare se qualcuno si trova nella casa
* Richiedere le condizioni attuali della casa in termini di umidità, luminosità, temperatura e movimento

ARCHITETTURA

\*\*DISEGNO\*\*

Servizi Cloud si terze parti: si è scelto di usare il servizio API offerto da Apixu.com per la gestione delle previsioni meteo e il servizio di creazione di assistenti conversazionali e webhook offerto da API.ai.

Dispositivi Utente: l’utente può interagire col sistema tramite l’uso di un qualsiasi telefono cellulare connesso a internet, previa installazione dell’applicazione Telegram.

Nodo Computazionale: il programma che riceve e interpreta le richieste dell'utente e che gestisce il controllo del comfort è eseguito su Raspberry Pi.

Come si può osservare guardando la struttura del progetto, si è scelto di dividere il sistema in 6 moduli che interagiscono tra loro. Il più importante di questi è quello denominato “httpServer” all’interno del quale si possono trovare i sotto-moduli che gestiscono le richieste/risposte HTTP tramite WebHook.

Il sotto-modulo “ApiAiListener” è quello principale di tutto il sistema, questo, infatti, contiene il Main che: inizializza alcuni parametri (convertitore JSON, parametri ZWave), spegne tutti i dispositivi connessi alla rete (presa e Hue), fa partire i threads necessari per controllare luminosità e luci, ed infine aspetta l’arrivo di comandi, richiama il metodo doWebHook del sotto-modulo “Command Executer”, il quale riceve come parametri un AIResponse (tipo di dato fornito dalle librerie di API.ai) e un FulFillment, e ritorna il Fulfillment stesso, modificato, in forma di JSON.

“CommandExecuter” è il sotto-modulo che esegue i comandi che arrivano dall’utente e imposta i messaggi di ritorno relativi ad ogni azione. All’interno del metodo “doWebHook” vengono fatte le seguenti operazioni, i cui nomi sono riconducibili alle Actions definite nei rispettivi Intents dell’assistente conversazionale creato in API.ai:

* “lightsOn”: azione che permette all’utente di accendere autonomamente le luci. Quest’azione esegue le operazioni necessarie all’accensione delle lampadine Philips Hue provando a:
  + chiamare il metodo corrispondente (“lightsOn”)[[1]](#footnote-2);
  + controllare se la modalità automatica è attiva, nel caso, la disattiva e scrive un messaggio per l’utente, avvisando della disattivazione della modalità automatica;
  + impostare il messaggio di ritorno per l’utente

Se qualcuna di queste operazioni fallisce, genera un’eccezione corredata da rispettivo messaggio d’errore. In ogni caso, alla fine dell’azione verrà inviato un messaggio all’utente.

* “lightsOff”: azione che permette all’utente di spegnere autonomamente le luci. Il funzionamento di quest’azione è sostanzialmente identico a quello dell’azione sopra descritta, l’unica differenza si trova nel metodo chiamato (“lightsOff”)1.
* “lightsPower”: azione che permette all’utente di impostare autonomamente la luminosità delle luci. Il funzionamento, ancora una volta, è molto simile ai precedenti, le differenze si possono vedere nel metodo chiamato (“lightsPower”)1 e nella gestione del valore di percentuale dato in input dall’utente, il quale non dovrà essere al di sotto dello 0 o al di sopra del 100.
* “time”: azione che permette all’utente di chiedere data e ora del giorno. Quest’azione richiede, semplicemente, l’ora di sistema del Raspberry su cui sta girando il sistema e lo ritorna all’utente
* “luminosity”: azione che permette all’utente di ricevere informazioni riguardanti la luminosità dell’ambiente. Quest’azione richiama il metodo (“getLuminosity”)[[2]](#footnote-3) e ritorna il valore in Lux della luminosità.
* “humidity”: azione che permette all’utente di ricevere informazioni sull’umidità interna della casa. Quest’azione richiama il metodo (“getHumidity”)2 e ritorna il valore in % d’umidità.
* “motion”: azione che permette all’utente di ricevere informazioni sul movimento all’interno dell’abitazione. Quest’azione richiama il metodo (“getMotion”)2 e ritorna semplicemente vero o falso.
* “temperature”: azione che permette all’utente di ricevere informazioni sulla temperatura della casa. Quest’azione richiama il metodo (“getTemperature”)2 e ritorna il valore in °C.
* “sensorInfo”: azione che permette all’utente di raggruppare tutte le informazioni precedenti in un unico messaggio
* “getWeather”: azione che permette all’utente di ricevere informazioni sulle condizioni meteo attuali oppure su 3 giorni. Questa richiama, a seconda della scelta dell’utente, (“getExternalWeather”)[[3]](#footnote-4) oppure (“getForecast”)3 e ritorna all’utente le previsioni meteo.
* “setLights”: azione che permette all’utente di impostare gli orari in cui accendere le luci in modo automatico. Quest’azione fa uso di un thread che memorizza su file di testo gli orari per l’accensione automatica delle luci
* “setLightsReset”: azione che permette all’utente di eliminare le regole impostate precedentemente. Quest’azione fa uso di un thread per resettare il file di testo di cui prima
* “autoModeOn”: azione che permette all’utente di attivare la modalità automatica di gestione del sistema di accensione luci
* “autoModeOff”: azione che permette all’utente di disattivare la modalità automatica di gestione del sistema di accensione luci
* “setComfort”: azione che permette all’utente di impostare la temperatura secondo lui ideale da mantenere all’interno dell’abitazione

Come citato più volte nella spiegazione precedente, sono stati usati dei Threads. Questi si trovano all’interno del package “controlThreads”, e ognuno è indipendente dall’altro.

Il thread “ComfortControl” si occupa della gestione della temperatura interna e della luminosità. Per quanto riguarda la gestione della luminosità \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*SONO QUI\*\*\*\*\*\*

Per poter utilizzare al meglio questo servizio è necessario avere:

* 1 Lampadina Philips Hue
* 1 Stufetta Elettrica
* 1 Multisensore Z-Wave
* 1 Presa Z-Wave
* 1 Portalampade
* 1 Raspberry Pi (RPi 06)

Purtroppo questo servizio ha alcune importanti limitazioni:

* è ridotto ad 1 luce e 1 presa, perciò non gestisce effettivamente una casa
* non è possibile attivare e disattivare manualmente la presa collegata alla stufetta
* la soluzione finale non comprende la gestione di comandi vocali
* sarebbe necessario collegarsi ad un termostato, il quale però non espone nessun servizio API
* l’uso della stufa elettrica, per come è stato studiato il sistema, porta a consumi molto elevati, in quanto la temperatura viene mantenuta intorno al valore ideale anche quando non c’è nessuno in casa

SCELTE?????????

-scelte fatte : esempio si è scelto di far girare comfort control ogni 500 0 3000 millisec anche se a temp non varia di cosi tanto in così poco tempo, ma solo perchè lui controlla anche intensità luci. (Quindi esempio scelta fatta stata : non teneretroppo conto dei consumi e delle "prestazioni" o consumo di memoria)

altro non mi viene in mente, da vedere quali sono i "buchi" o le cose poco chiare una volta finita la relazione e metterle qua

1. Metodi del sotto-modulo di “HueController”, Hue [↑](#footnote-ref-2)
2. Metodo del sotto-modulo di “ZwaveController”, Zwave [↑](#footnote-ref-3)
3. Metodo del sotto-modulo di “weatherApi”, WeatherGetter [↑](#footnote-ref-4)