Sommario

Smart Street Lights

Introduzione

Il progetto si basa sulla realizzazione di un sistema IoT il cui scopo è la gestione automatica di lampioni stradali che si accendono e si spengono in risposta a determinati eventi.

Il funzionamento del sistema riguarda due modalità, quella automatica che gestisce il modo autonomo i lampioni, ovvero accende e spegne quest'ultimi in base alla luce esterna rilevata, inoltre regola l'intensità della luce, quando accesa, che aumenta al passaggio di veicoli in prossimità dei lampioni grazie all'attivazione dei sensori a infrarossi. L'altra modalità è quella manuale che sfrutta una pagina web dove un addetto può pilotare i vari lampioni in tre modalità: automatica, accesa alla luminosità massima oppure spenta.

Obiettivi e contributi del gruppo

L'obiettivo principale di questo sistema è il risparmio energetico, in quanto riduce i consumi nelle ore in cui le strade sono meno trafficate, mantenendo sempre una buona visibilità sulla strada. Le luci possono essere comandate singolarmente e quindi è possibile aumentare la loro intensità in caso di necessità, ad esempio per lavori in corso, attraversamenti pedonali, emergenze ecc.

Il progetto è stato implementato da Leonardo Scandino e Marco Tateo per lo più in collaborazione, ma concentrandosi alle volte singolarmente su vari aspetti del progetto, ovvero il primo sulla parte automatica del sistema e le sue parti fisiche (raspberry, sensori ecc.), e il secondo sulla parte web e la comunicazione tra quest'ultima e la parte fisica.

Materiali e strumenti utilizzati

- Raspberry pi 4

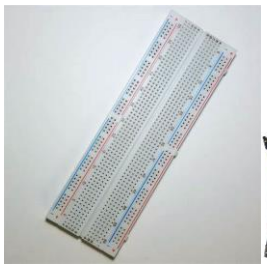
- Sensori
    - Due sensori a infrarossi IR

    - Fotoresistenza

- Due luci dimmerabili a 12v

- Alimentatore 10 Ampere 12V stabilizzato 220V 120W

- Breadboard, jumper e condensatore



- Due moduli mosfet IR520 per regolare le uscite PWM

- Flask

Collegamenti dei componenti

Il cuore del sistema è un raspberry pi 4, un computer a scheda singola progettata per ospitare sistemi operativi basati sul kernel Linux, utilizzato nel nostro caso per gestire tutti gli elementi del progetto, alle varie GPIO disponibili sono stati collegati tutti i sensori e tutte le luci.

I sensori a infrarossi IR presentano tre pin ovvero, VCC (collegata alla GPIO 3.3v), GND(collegato al GND) e il segnale OUT collegati rispettivamente alle due GPIO 22 e 23.

La fotoresistenza invece ha bisogno di un condensatore da 0.1 uf per calcolare il livello di luce e come per i precedenti ha bisogno di alimentazione e GND, il segnale passa per la GPIO 4.

La parte più complicata dei collegamenti sono le luci. Queste sono luci dimmerabili e devono essere alimentate a 12V quindi necessitano di un alimentatore, in questo caso di un alimentatore 10 ampere 12V stabilizzato 220V 120W e di un modulo mosfet IRF520 che permette di regolare le uscite PWM. Le due luci sono collegate ai due canali PWM presenti su Raspberry pi ovvero GPIO 13 e 18.

Per semplicità tutto quanto è stato collegato ad una breadboard tramite dei jumper.

Parte web

In questa fase, per sviluppare l'applicazione Python si è fatto uso di un Web framework Open Source di nome Flask, tramite le sue librerie Flask, render_templates e request.

L'implementazione di questo framework consiste in poche righe di codice, ma che permettono di controllare le luci a piacimento attraverso le richieste provenienti dalla pagina Web.

Flask riesce ad accedere ai vari templates e file attraverso il comando ***app= Flask(__name__)***, ***name*** si riferisce al modulo python dove l'istanza ***app*** è stata chiamata.

Abbiamo posto ***App.route(' ')*** al di sopra di alcune funzioni, il quale permette di eseguirle in base al percorso URL selezionato. Inserendo degli specifici parametri è possibile quindi eseguire delle specifiche funzioni di codice: come ad esempio ***('/')*** permette di eseguire la funzione ***main()*** all'apertura della pagina Web, caricando così nel template di main.html il dizionario delle luci; oppure ***('/light', methods = ['POST'])*** , dove, quando l'addetto seleziona le varie modalità dei lampioni (quindi: automatica, acceso o spento) tramite i pulsanti predisposti, verrà eseguita la funzione ***click(),*** la quale attiverà la modalità desiderata sulle specifiche luci.

Sviluppo del codice

Il codice è suddiviso in due file ovvero mainApp.py che è il nostro codice principale scritto in python e il main.html che è il codice html per l'applicazione web.

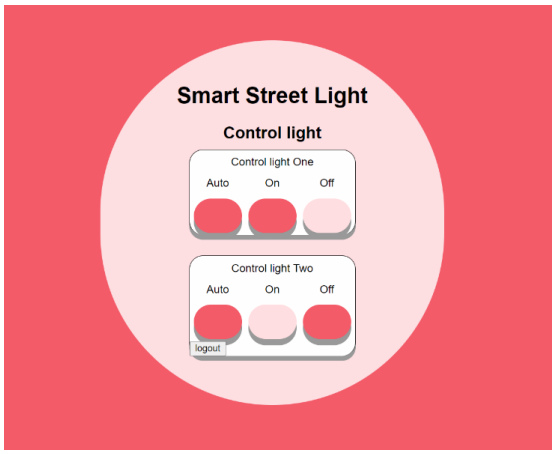
In mainApp.py vengono importate le librerie che serviranno nel corso del progetto ovvero RPi.GPIO, time, multiprocessing e Flask.

Per fare interagire la parte web e la parte automatica viene utilizzato un dizionario denominato lights dove sono inserite le informazioni più importanti ovvero: i pin delle luci, il pulsante selezionato nella pagina web, il pin del sensore a infrarossi, il pwm di riferimento e il processo.

A questo punto vengono settati tutti i sensori e tutte le luci tramite dei cicli e salvati nel dizionario.

Il funzionamento della parte automatica è essenzialmente sviluppato in due definizioni principale, ovvero dimmerLuce e valueFoto. La prima si occupa di leggere l' output di valueFoto e se il valore è alto accende le luci al 40%, dal momento che le luci sono accese e un veicolo passa in prossimità di un sensore a infrarossi la luce di riferimento, ovvero quella che la macchina deve raggiungere, si accenderà per due secondi al 100% per poi tornare al 40%. La seconda si occupa di leggere i dati prodotti dalla fotoresistenza e ricavarne il valore.

In seguito, c'è la parte di codice che si interfaccia anche alla parte web descritta in precedenza.



Come prima cosa viene passato il dizionario iniziale alla nostra pagina web che farà visualizzare lo stato iniziale, a questo punto viene inserita una definizione click che cambia lo stato del nostro dizionario e lo aggiorna.

Se si clicca su Auto viene modificato il pulsante in AUTO, viene spenta qualsiasi luce, in quanto nella modalità automatica se è giorno la luce non deve essere accesa, e infine attiva il processo che sfrutta dimmerLuce e valueFoto (inserendo nel dizionario che la luce di riferimento ha quel processo in esecuzione).

Se si clicca su OFF viene modificato il pulsante in OFF, il codice verifica se il processo è vivo e in caso affermativo lo termina. Il funzionamento di ON è il medesimo solo che accende la luce al 100%

Ad ogni click il dizionario viene aggiornato.

Nella parte html si costruisce la pagina web e grazie al dizionario sempre aggiornato tramite delle condizioni modifichiamo i pulsanti e i colori in modo da visualizzare sullo schermo lo stato di ogni luce in quel momento.

Sicurezza

Per la parte di sicurezza abbiamo riportato qua sotto una piccola ricerca per l'accesso sicuro a Raspberry pi e implementato invece un metodo per l'accesso all'applicazione web:

- Il primo è un'autenticazione SSH (Secure Socket Shell) a due fattori su Raspberry pi Questo procedimento richiede l'immissione di due informazioni per l'accesso, in quanto dovrà essere inserita oltre al normale accesso username e password, un codice generato da Google Authenticator.
  Google Authenticator è un servizio di generazione di token realizzato da Google distribuito come applicazione mobile.
  Nella fase di setup di quest'ultimo vengono forniti un QRcode, una secret key e cinque codici di emergenza in caso di smarrimento del dispositivo associato. Una volta scannerizzato il QRcode, che abilita Google Authenticator sul dispositivo, bisognerà includere questa modalità di autenticazione al file di configurazione pam.d/sshd del Raspberry, inserendo la stringa "auth required pam_google_authenticator.so". Dopo aver riavviato Raspberry pi al momento della connessione in ssh verrà richiesto, oltre all'username e la password, il codice ricevuto sul dispositivo nell'applicazione Google Authenticator.
  In questo modo si intensifica la sicurezza per l'accesso in remoto tramite ssh al Raspberry e quindi proteggere il codice da eventuali attacchi.
- Il secondo, che a differenza del primo è stato implementato, riguarda l'autenticazione per l'accesso alla parte web dove si possono gestire le luci. Questa parte si può realizzare con flask. Dopo aver importato da flask le librerie necessarie (session e redirect), si crea un dizionario user dove viene inserito username e password dell'utente abilitato.
  Da qui dopo aver creato una pagina login.html si richiede l'inserimento delle credenziali per l'accesso.
  I dati inseriti sono gestiti nel codice in python, dove vengono confrontati con il dizionario user.

  Se i dati coincidono si viene indirizzati sulla pagina principale main.html per la gestione delle luci, se al contrario sono errati viene mostrato un messaggio di errore: "username o password errati", in questo modo si blocca l'accesso a utenti non autorizzati. Per completare il login c'è bisogno anche di una opzione logout implementato sulla pagina main.html che esce dalla sessione e ritorna al login.

Wrong username or password