Color Picker

Alberto Pes
Francesco Littarru

Un dispositivo dotato di una fotocamera, un pulsante e di un LED RGB (o un monitor):

- Quando viene premuto il pulsante, la fotocamera registra l'immagine di un oggetto che ha davanti e la invia al backend Cloud;
- Una funzione in Cloud identifica il colore dominante dell'immagine e lo converte in valori R/G/B, che vengono inviati al dispositivo;
- Il dispositivo imposta il LED RGB (o mostra sul monitor) il colore individuato
- Un'interfaccia web mostra le immagini acquisite e i valori R G B riconosciuti

Specifiche del progetto

Il progetto mira alla realizzazione di un dispositivo IoT in grado di catturare un'immagine a colori tramite una fotocamera azionata mediante un apposito pulsante. Successivamente il dispositivo invia l'immagine al server cloud che provvede al processamento della stessa estrapolando il valore RGB corrispondente al colore predominante presente nell'immagine e provvede infine all'invio di tale informazione al dispositivo IoT. Il dispositivo provvederà alla visualizzazione di tale valore mediante un display LCD.
Le immagini inviate al server cloud e i valori RGB ricavati saranno inoltre disponibili tramite interfaccia web per la successiva visualizzazione e gestione da parte dell'utente.

Possiamo dunque individuare in questa descrizione di dettaglio le seguenti macro specifiche:

- Catturare un'immagine a colori tramite una fotocamera.
- Azionare la fotocamera mediante un apposito pulsante.
- Invio dell'immagine acquisita al server cloud.
- Processamento lato server dell'immagine ai fini dell'individuazione del colore predominante e del valore RGB ad esso corrispondente.
- Invio da parte del server cloud del valore RGB al dispositivo.
- Ricezione del valore RGB da parte del dispositivo.
- Visualizzazione del valore ricevuto mediante display LCD.
- Realizzazione dell'interfaccia web che consente la visualizzazione delle immagini acquisite dal dispositivo IoT e i relativi valori RGB ricavati.

Segue un approfondimento per descrivere come il sistema adempie alle specifiche elencate e quali sono state le scelte progettuali che hanno caratterizzato lo sviluppo.

Catturare un'immagine a colori tramite una fotocamera
Per realizzare questa prima specifica l'hardware di base del sistema, costituito dalla board NodeMCU (aggiungere breve definizione) è stato dotato di una fotocamera ad esso compatibile tramite interfaccia uart.
La fotocamera scelta consente il collegamento alla board tramite i seguenti quattro pin:

1. TX: collegato al pin D3 della NodeMCU
2. RX: collegato al pin D4 della NodeMCU
3. GND: collegato al pin GND della NodeMCU
4. 5V: collegato al pin 3,3V della NodeMCU

La libreria fornita dal produttore Adafruit include tutte le funzioni necessarie per il suo utilizzo:

- Inizializzazione ed ottenimento della versione della camera utilizzata
- Settaggio della risoluzione di acquisizione
- Esecuzione dello scatto
- Ottenimento della dimensione dell'immagine acquisita
- Lettura del flusso di bit relativi all'immagine per la successiva memorizzazione

La libreria, denominata Adafruit-VC0706-Serial-Camera-Library, è stata dunque inclusa nel progetto per consentire la gestione della fotocamera.

Azionare la fotocamera mediante un apposito pulsante
Questa seconda specifica ha richiesto l'aggiunta di un pulsante al sistema. La funzione di scatto viene così attivata a seguito della pressione del bottone posizionato nella breadboard. Il collegamento del bottone è stato effettuato come segue.

Un pin del bottone viene alimentato tramite il pin da 3,3V della NodeMCU.
L'altro pin del bottone viene collegato ad una resistenza da 10KOhm a GND e poi tramite un jumper al pin D0 della NodeMCU.

Lato software si è provveduto ad impostare il pin D0 in modalità input e poi a leggere il suo stato all'interno di un ciclo infinito. La pressione del bottone viene così rilevata e determina l'esecuzione di tutte le operazioni relative alla fotocamera, alla memorizzazione dell'immagine nella memoria della NodeMCU e al successivo invio al server cloud.

Invio dell'immagine acquisita al server cloud
Per completare questa fase si è reso necessario memorizzare preventivamente l'immagine acquisita nella scheda SD.

Si è dunque provveduto per prima cosa ad ottenere la dimensione in byte dell'immagine acquisita, per poi impostare all'interno di un ciclo una lettura a chunk di 64 byte dal buffer della fotocamera. Passo dopo passo, ogni chunk è stato memorizzato in una variabile e poi scritto su un file nella scheda SD.

Per inviare l'immagine si è provveduto ad aprire il file come buffer in lettura, quindi inviato con le funzioni FTP importate.

Processamento lato server dell'immagine ai fini dell'individuazione del colore predominante e del valore RGB ad esso corrispondente.

Il server di processamento è stato configurato utilizzando la piattaforma di servizi web Amazon AWS:

- Amazon S3 per la memorizzazione delle immagini inviate dai dispositivi.
- Amazon EC2 per la attivazione dell'FTP
- Amazon Lambda per l'integrazione del codice Python che si occupa dell'elaborazione dell'immagine per determinare il colore predominante ed ottenere il relativo valore RGB.

Nell'istanza EC2 Ubuntu è stato montato il bucket di S3 attraverso l'uso del filesystem virtuale s3fs, permettendo il trasporto trasparente delle immagini dall'istanza EC2 al bucket S3.
L'inserimento delle immagini nel bucket montato comporta l'attivazione di un evento, in questo caso specifico, l'inserimento di immagini con suffisso ".jpg" comporta l'esecuzione della funzione lambda che calcola il colore predominante.
La funzione legge l'evento in cui sono presenti i metadati dell'immagine e del bucket, quindi preleva l'immagine dal bucket utilizzando la libreria boto3 e infine inserisce i risultati del calcolo in un file .json che registra le immagini di ogni utente e per ogni immagine i suoi valori di colore.

Nel dettaglio la funzione Lambda è stata implementata come segue:
La funzione è stata implementata in python con le librerie opencv e paho-mqtt.
Il modulo boto3 permette il trasferimento da e verso S3.
L'immagine viene prima caricata nell'ambiente locale da S3, si procede quindi alla sua elaborazione mediante opencv effettuando la somma dei 3 colori principali.
Infine con la libreria mqtt si invia la stringa rappresentante i colori dell'immagine al dispositivo nodemcu.

Invio da parte del server cloud del valore RGB al dispositivo.

Questa specifica è stata osservata realizzando una comunicazione tra server e dispositivo mediante MQTT.
Il nome dell'immagine è formato dal mac address del dispositivo inviante, concatenato con il numero di serie dell'immagine (001, 002 ecc).
Per gestire in modo semplice più utenti, la funzione utilizza il mac address letto dal nome dell'immagine come topic su cui pubblicare il messaggio.
Il messaggio viene inviato come una stringa di otto caratteri nel formato "RR/GG/BB"

Ricezione del valore RGB da parte del dispositivo

Per adempiere a questa specifica il dispositivo deve sottoscrivere il topic denominato con il suo mac address per leggere il valore relativo all'immagine trasmessa.

Tramite una funzione di callback il messaggio contenente il valore RGB pubblicato dal server nel topic potrà essere letto dal dispositivo e di conseguenza visualizzato come descritto nel seguito.

Visualizzazione del valore ricevuto mediante display LCD ed attivazione del LED RGB

Questa specifica ha reso necessaria l'aggiunta al sistema del display LCD 16x2 con interfaccia I2C disponibile in dotazione nel kit NodeMCU.
Il collegamento del display è stato realizzato nel modo seguente:

- GND: collegato al pin GND della NodeMCU
- VCC: collegato al pin VIN della NodeMCU
- SDA: collegato al pin D2 della NodeMCU
- SCL: collegato al pin D1 della NodeMCU

L'utilizzo del pin VIN è stato necessario in quanto lo schermo richiede 5V di alimentazione. Si è utilizzata una libreria specifica per l'utilizzo del display denominata *LiquidCrystal_I2C* che consente la gestione della stampa dei caratteri sul display, ciò ha consentito quindi di riportare il valore RGB riconosciuto nonché mostrare delle indicazioni per l'utente mediante la funzione *printlcd()*.

Realizzazione dell'interfaccia web che consente la visualizzazione delle immagini acquisite dal dispositivo IoT e i relativi valori RGB ricavati

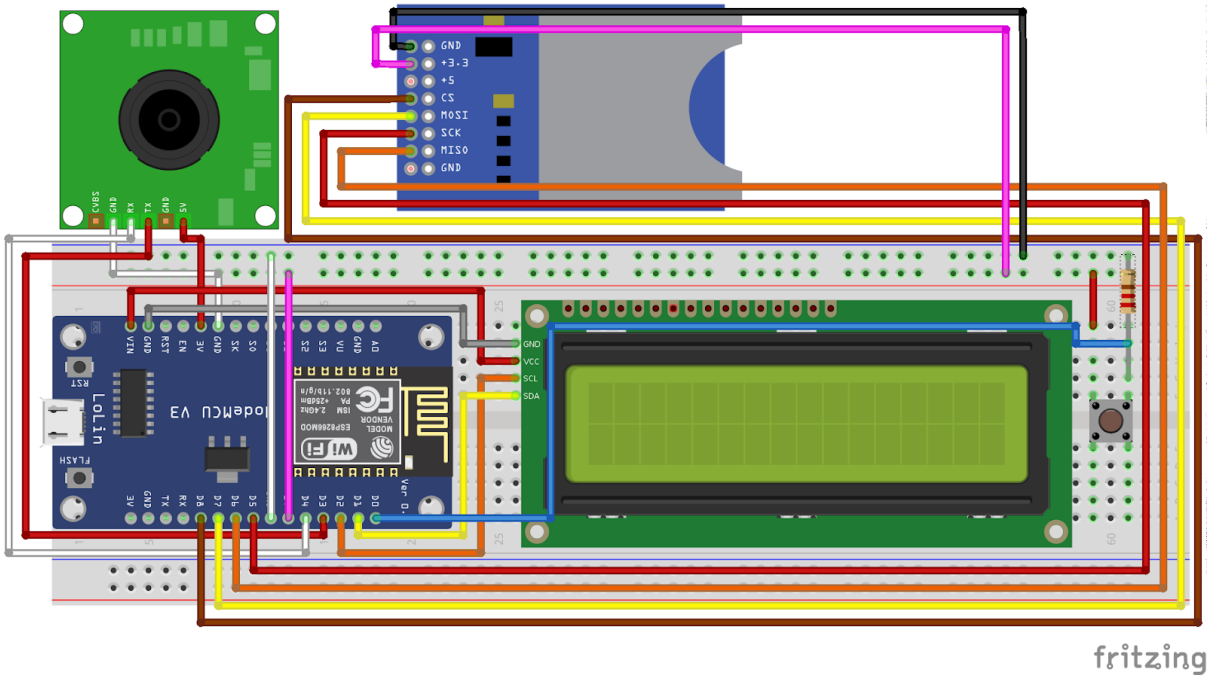
Per questa specifica è stata realizzata un'interfaccia di tipo serverless mediante l'utilizzo del servizio Amazon S3 e Amazon Cognito, nello specifico è stato creato un bucket per il caricamento dei contenuti statici del sito, è stata impostata una policy specifica per il bucket che consente l'accesso pubblico ai file in esso contenuti ed è stato inoltre attivata l'opzione *hosting siti web statici*. L'interfaccia web è stata concepita per consentire la gestione di utenti multipli, questo è stato realizzato mediante la creazione di un pool di utenti Cognito e la relativa aggiunta dell'applicazione al pool di utenti creato.
La funzionalità di visualizzazione delle immagini acquisite dal dispositivo è stata realizzata mediante la gestione di un file .json in cui sono stati memorizzati i percorsi per il caricamento di ogni immagine sull'interfaccia e i relativi valori RGB da mostrare.
L'utente che si collega alla piattaforma potrà dunque effettuare la registrazione inserendo username, numero identificativo univoco associato al dispositivo (mac address) e la password. L'utente riceverà tramite e-mail un codice di verifica per abilitare il proprio account che dovrà inserire nella pagina successiva a quella di registrazione. In caso di successo, l'utente potrà effettuare il login ed accedere al servizio di visualizzazione delle immagini. In questa sezione sarà possibile, passando il mouse su ogni immagine, visualizzare il relativo valore RGB nonché il colore predominante individuato.

Tecnologie Hardware e Software utilizzate

Ripercorrendo le funzionalità espresse possiamo indicare le seguenti tecnologie Hardware utilizzate per la realizzazione del progetto:

- Board NodeMCU
- Fotocamera Adafruit TTL JPEG Camera (VC0706 chipset)
- Display I2C LCD (16x2)

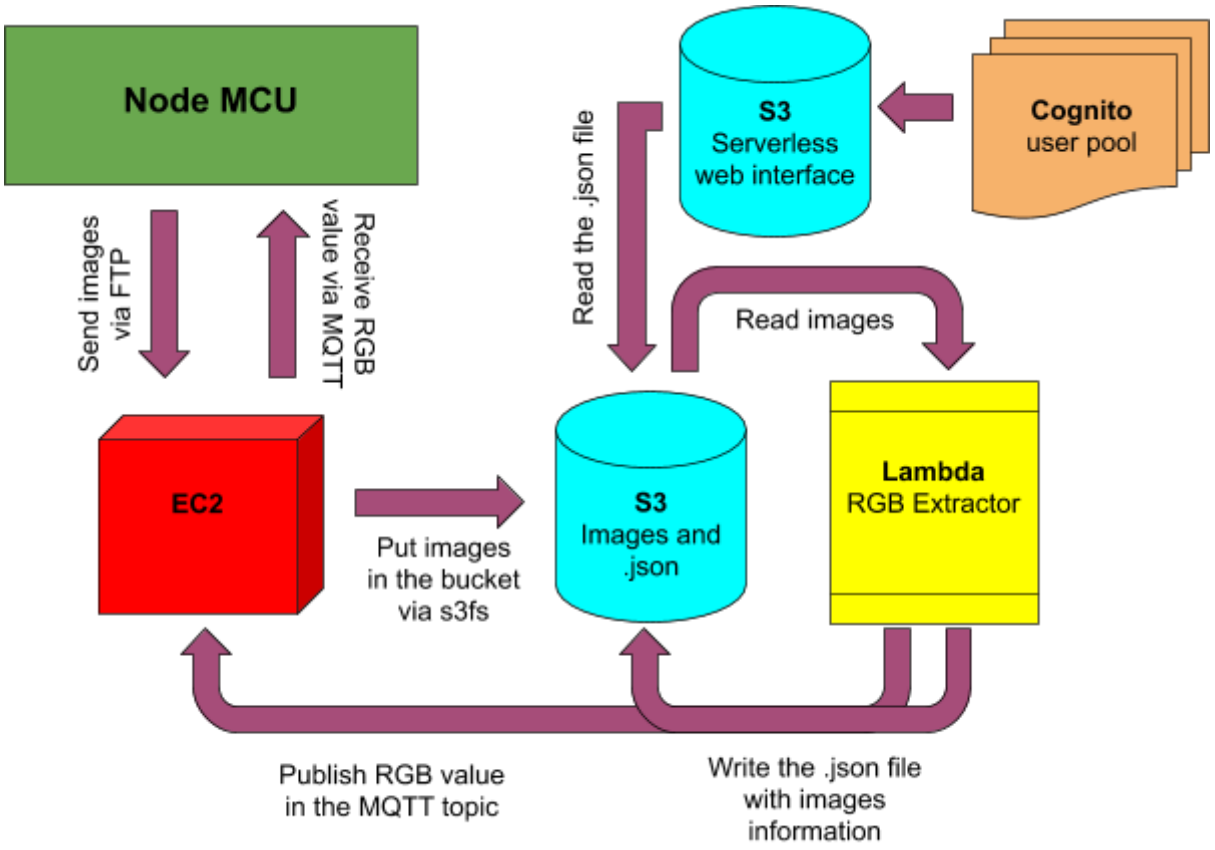
Riportiamo di seguito lo schema del dispositivo:



Per quanto concerne le tecnologie Software utilizzate, menzioniamo:

- Servizi Amazon:
  - S3
  - EC2
  - Cognito
  - Lambda
- L'implementazione della Lambda è stata realizzata mediante codice Python
- La comunicazione tra il dispositivo e il server per l'invio delle immagini è stata realizzata mediante FTP.
- La comunicazione tra server e dispositivi per la trasmissione del valore RGB avviene mediante tecnologia MQTT.
- L'implementazione dell'interfaccia Web è stata realizzata mediante codice HTML, CSS, Javascript, JQuery.
- Sono state necessarie le seguenti librerie:
  - NodeMCU: Adafruit_ESP8266
  - Fotocamera: Adafruit VC0706 Serial Camera Library
  - Display: LiquidCrystal_I2C
  - MQTT lato NodeMCU: PubSubClient

Mostriamo di seguito uno schema dell'architettura Software:



Contesto, problematiche da affrontare, motivazione delle scelte progettuali

Nella fase di progettazione e dell'assemblaggio dei componenti è stata considerata la loro disposizione al fine di costruire un dispositivo intuitivo e facilmente maneggiabile. Il display è stato posizionato al centro della BreadBoard, la fotocamera nella parte sinistra e il bottone per il suo azionamento nella parte destra. La scelta del pin D0 per la connessione del bottone è motivata dal fatto che rappresenta tra tutti quello più vicino considerando il posizionamento del bottone. La scelta della fotocamera è stata effettuata a seguito di un'analisi dei dispositivi presenti sul mercato, il modello scelto è risultato il migliore in quanto garantisce una modalità semplice di collegamento tramite soli due pin esclusi quelli di GND e di alimentazione; consente inoltre una buona rese nei colori ad un costo molto contenuto. L'utilizzo del display ha reso necessario inizialmente l'esecuzione di un **codice di analisi** denominato *IC2_Scanner* per individuare l'indirizzo di interfacciamento da esso utilizzato, è stato inoltre necessario **modificare il potenziometro** posizionato posteriormente al display al fine di regolare in modo ottimale il contrasto. Al posto di un semplice led, si è optato per l'integrazione di un display in quanto questo consente di guidare l'utente durante l'utilizzo del dispositivo con semplici indicazioni e di rappresentare in modo più concreto il valore RGB rilevato.

L'integrazione della fotocamera ha rappresentato indubbiamente la fase più **problematica** e colma di **ostacoli**. I primi test di scatto hanno avuto infatti esito negativo, in particolare i file delle immagini risultavano corrotti e non processabili. Si è quindi scelto di testare la fotocamera con la board Arduino coadiuvata da un lettore di schede SD come riportato nella documentazione ufficiale, questo test ha dato esito positivo, ogni immagine catturata veniva

infatti correttamente memorizzata nella scheda. Questa situazione ha determinato la scelta di affiancare un lettore di schede SD alla NodeMCU e ripetere il test di scatto. Quest'ultimo test è risultato positivo, le immagini venivano correttamente memorizzate nella scheda. Possiamo dunque affermare che il funzionamento del filesystem SPIFFS non è sempre ottimale e in alcuni modelli della board può determinare errori di memorizzazione.
Per quanto riguarda la configurazione della tecnologia MQTT si è scelto, pensando ad un numero elevato di messaggi, di impostare un **topic dedicato ad ogni dispositivo**, questo consente inoltre una più rapida lettura del payload lato device senza la necessità di controllare ogni singolo messaggio per valutare ed eventualmente rimuovere il codice identificativo.

Per la progettazione e successiva implementazione dell'interfaccia web si è scelta un'architettura **serverless** che costituisce una valida strategia per il contenimento dei tempi e dei costi legati alla realizzazione e soprattutto alla manutenzione in un contesto reale di utilizzo. Con questa configurazione infatti è stato possibile realizzare un'interfaccia con relativa gestione utenti senza l'utilizzo di alcun servizio EC2, che avrebbe implicato l'installazione di strumenti web dedicati, ma esclusivamente mediante la configurazione del bucket Amazon S3 e del pool di utenti mediante Amazon Cognito.

Validazione e sperimentazione

Modalità di validazione

Per ciascuna funzionalità descrivere cosa il test funzionale voleva dimostrare e come lo ha dimostrato:
Catturare un'immagine a colori tramite una fotocamera

- Test 1: Scatto di una foto e verifica del corretto salvataggio nella scheda SD
  Il test condotto mira a dimostrare il corretto funzionamento della fotocamera e delle procedure di memorizzazione.
  Sono stati effettuati diversi scatti con configurazioni differenti di risoluzione ed è stato verificato che tutte le immagini scattate fossero disponibili senza alcun errore nel dispositivo di memorizzazione microSD.

Invio dell'immagine acquisita al server cloud

- Test 1: Trasmissione dell'immagine salvata al server Could tramite FTP.
  Il test condotto mira a dimostrare la corretta trasmissione dell'immagine al server.
  Il dispositivo è stato configurato per la connessione tramite FTP e si è verificato che tutte le foto scattate anche a risoluzioni differenti fossero inviate correttamente e fossero disponibili lato server senza errori.

Processamento lato server dell'immagine ai fini dell'individuazione del colore predominante e del valore RGB ad esso corrispondente.

- Test1: Verifica dell'estrapolazione del valore RGB predominante
  Il test condotto mira a verificare la corretta individuazione del colore predominante presente nell'immagine.
  Sono state scattate e trasmesse al server cinque immagini con differenti tonalità di colore, si è quindi verificato che i valori restituiti dalla funzione lambda di

elaborazione fossero congrui e paragonabili a quelli restituiti dal seguente servizio web.

Invio, ricezione e visualizzazione del valore ricevuto mediante display LCD.

- Test 1: Verifica del valore RGB inviato, ricevuto e visualizzato.
  Il test condotto mira a verificare che il valore RGB ricevuto e visualizzato sia quello inviato dal server e relativo all'ultima immagine scattata. Quindi sono state scattate cinque immagini in sequenza che riprendessero differenti tonalità di colore e ci si è assicurati che ogni valore mostrato fosse corretto e relativo all'ultima immagine considerata.

Realizzazione dell'interfaccia web che consente la visualizzazione delle immagini acquisite dal dispositivo IoT e i relativi valori RGB ricavati.

- Test 1: Creazione di un utente e verifica della corretta trasmissione dei dati dall'interfaccia al servizio amazon Cognito.
  Il test condotto ha dimostrato la corretta memorizzazione dei dati relativi all'utente che sono verificabili accedendo alla sezione di gestione del pool utenti di Amazon Cognito.
- Test 2: Visualizzazione delle foto dell'utente nell'interfaccia web.
  Questo test mira a verificare che le immagini scattate dall'utente loggato siano visualizzate nell'interfaccia e nessun'altra immagine di eventuali utenti sia consultabile. La convalida è stata svolta effettuando il login da due account differenti e verificando che le immagini visualizzate appartenessero all'account corretto.

Manuale di riuso

1. Predisposizione dei servizi Amazon:
   Per predisporre il server FTP è necessario prima:
   1) Creare un bucket su s3
   2) Impostare un utente con il servizio IAM che permetta l'accesso al bucket di S3. È possibile anche creare una policy per l'accesso al bucket di S3 e a CloudWatch e quindi associarla all'utente creato.

   Dall'utente si generano le chiavi d'accesso ai servizi amazon nella scheda delle credenziali, basate sulle policy impostate, che verranno utilizzate sull'istanza ec2 e sulla lambda.
   Prima della creazione dell'istanza EC2 è necessario creare un gruppo di sicurezza dalla dashboard con le regole per aprire le porte per FTP e MQTT, solitamente vengono utilizzate 20-21 1024-1048 per FTP e 1883 e 8883 per MQTT.

   A questo punto è possibile lanciare l'istanza EC2 ubuntu server.
   La prima cosa da fare è installare il demone vsftpd, quindi si imposta il suo file di configurazione /etc/vsftpd.conf.
   Nello specifico è necessario abilitare la modalità passiva del demone, in quanto il server si trova dietro un firewall:
   pasv_enable=YES
   pasv_min_port=1024

pasv_max_port=1048
pasv_address=ip.dell.istanza.ec2

È necessario abilitare l'utente locale in scrittura per poter scrivere i dati in entrata, inserendo il suo nome nel file /etc/vsftpd.chroot_list.
write_enable=YES
allow_writeable_chroot=YES
chroot_local_user=YES
chroot_list_enable=YES
chroot_list_file=/etc/vsftpd.chroot_list

Nei passi successivi si utilizza un utente di esempio “ftpuser”.
L'utente di esempio "ftpuser" viene creato con:
sudo adduser ftpuser, a cui segue l'inserimento della password
quindi si può riavviare il server ftp con:
sudo systemctl restart vsftpd.service

A questo punto il server FTP è configurato per memorizzare i dati in arrivo nell'istanza EC2

Montare bucket s3 in ec2.
Per montare il bucket su ec2 si è scelto di utilizzare s3fs, il quale è supportato da amazon.
Si installa normalmente con sudo apt-get update install s3fs
I comandi successivi devono essere eseguiti come utente ftpuser, quindi si esegue su ftpuser
All'interno della home dell'utente si crea il file .passwd-s3fs, contenente le chiavi di accesso ai servizi amazon precedentemente generate nel servizio IAM.
echo KEY:SECRET:KEY > .passwd-s3fs
Ci si assicuri che sia in modalità rw solo per ftpuser, ovvero si esegue
chmod 600 .passwd-s3fs
A questo punto è possibile montare il bucket utilizzando come mount point una cartella locale che deve essere vuota eseguendo il comando:
s3fs nomebucket /mountpoint -o passwd_file=/.passwd-s3fs

questo permette a s3fs di montare il bucket con le credenziali di accesso di aws.
Se il bucket è stato montato correttamente dovrebbe essere possibile vedere un file system s3fs nella lista generata dal comando “df -h”

Affinchè sia possibile comunicare con il dispositivo via mqtt si installano i seguenti pacchetti:
sudo apt-get install mosquitto-clients mosquitto

2. Creazione della funzione lambda.
   La funzione lambda utilizzata è uno script in python 3 che richiede l'uso delle librerie paho-mqtt e opencv-python.

Non è possibile importare tutte le librerie direttamente dall'ambiente della lambda in quanto è un ambiente serverless ristretto, ed è dunque necessario che queste vengano caricate esternamente.
Si procede quindi nel seguente modo:
Si crea una cartella sul proprio pc che si deve obbligatoriamente rinominare "python", quindi al suo interno si installano le librerie necessarie con:
python3 -m pip install paho-mqtt -t .
python3 -m pip install opencv-python -t .
quindi si comprime la cartella "python" in zip.

Per creare la funzione lambda si procede dalla relativa pagina di servizio.
Nell'interfaccia di creazione si seleziona il runtime python >=3.6 e si seleziona il ruolo, che deve avere accesso sia a S3 che a CloudWatch affinchè gli eventi di S3 possano essere rilevati.
Dopodichè è possibile caricare le librerie precedentemente zippate dal menù layers (livelli), scegliendo il runtime python appropriato python >=3.6 e infine associarlo alla lambda corrente dalla schermata principale.
Successivamente è necessario impostare un trigger sul bucket di s3 che permetta di leggere gli eventi quindi su "aggiungi trigger" si sceglie il servizio S3 e il bucket, avendo cura di indicare ".jpg" come suffisso del file.
A questo punto si può concludere inserendo il codice della lambda disponibile nel repository https://github.com/skambuilds/ColorPicker/blob/master/lambda_handler.py

Nel codice è necessario indicare quale sia l'indirizzo pubblico del broker mqtt.

3. Installazione dell'ambiente di sviluppo Arduino.
4. Download della libreria per l'utilizzo del nodeMCU Adafruit_ESP8266.
5. Download della libreria PubSubClient per l'utilizzo del servizio MQTT.
6. Collegamento dei componenti come precedentemente indicato.
7. Download del codice disponibile al seguente collegamento.
    a. Installazione delle librerie presenti nella sotto-cartella *libraries* del progetto.
    b. Esecuzione dello script .ino disponibile nella sotto-cartella *I2C_Scanner* per l'individuazione dell'indirizzo del display.
    c. Aprire lo script disponibile nella sotto-cartella *SNAP_SPIFFS_FTP* e impostare tutti i parametri di connessione, wi-fi e FTP ed inserire infine l'indirizzo ricavato al punto precedente.
8. Collegare la scheda al computer e procedere al caricamento dello script.
9. Effettuare il caricamento dell'interfaccia web disponibile nella sottodirectory *WebInterface*, seguendo le prime due sezioni di questo tutorial.
10. Aggiornare il file ride.js nella sottocartella *WebInterface/js* con le informazioni riguardanti il bucket e il nome del file .json e procedere al suo caricamento sul bucket impostato al punto precedente.
11. Registrarsi tramite apposito form cliccando sul pulsante Sign Up nella homepage dell'interfaccia
12. Effettuare degli scatti con il dispositivo
13. Effettuare il login e visualizzare le foto scattate.

Consigli e suggerimenti sull'utilizzo dei servizi Amazon

Il servizio **IoT Core**, considerato inizialmente per la trasmissione delle immagini al bucket tramite MQTT è stato accantonato a causa di malfunzionamenti in fase di memorizzazione in favore del servizio **EC2** con macchina micro per la trasmissione mediante FTP.

Inoltre le funzionalità di shadow offerte dal servizio IoT Core risultano più adatte nel contesto dei sensori piuttosto che nel caso considerato in cui il dispositivo si limita ad identificare una caratteristica dell'immagine catturata quando l'utente ne ha necessità.

Nell'istanza EC2 Ubuntu è stato montato il bucket di **S3** attraverso l'uso del filesystem virtuale s3fs, permettendo il trasporto trasparente delle immagini dall'istanza EC2 al bucket S3 per la loro memorizzazione.

Il servizio **Lambda** ha consentito la realizzazione della procedura di identificazione del colore predominante integrando codice Python. Questo servizio consente la realizzazione di funzionalità dedicate con l'utilizzo dei più moderni linguaggi di programmazione ed è quindi particolarmente potente.

Per quanto concerne la realizzazione dell'interfaccia web i servizi Amazon hanno consentito rapidamente l'implementazione di un'architettura serverless a basso costo. Il servizio di storage S3 è di facile e immediata configurazione e consente l'attivazione della funzionalità di hosting per siti web statici. Il servizio **Cognito** ha consentito invece di gestire il pool di utenti per l'accesso alla piattaforma con funzionalità avanzate, come l'aggiunta di attributi personalizzati e le funzionalità di verifica dell'account.

La documentazione Amazon è molto ricca e ben strutturata, rappresenta un validissimo strumento di apprendimento.

Consigliamo dunque l'approfondimento dei servizi menzionati in quanto reputiamo possano costituire la spina dorsale di qualsiasi progetto IoT comprensivo di interfaccia web e in quanto rappresentano degli strumenti di cui un informatico dovrebbe essere a conoscenza vista la loro potenzialità e diffusione.