Sistemi e Applicazioni Cloud - Laboratorio

Appello 23 febbraio 2023 [Tempo consegna: 2h 30m]

Si realizzi un'applicazione su Google Cloud Platform per la gestione di un generico social network per lo scambio di messaggi. I messaggi saranno chiamati *chirps* e possono contenere hashtag nella forma *#topic*

Si rendano disponibili i seguenti tipi di interfacce di utilizzo:

1. API per inserire e recuperare messaggi
2. Pagina Web per inserire messaggi e per visualizzare messaggi sulla base di un hashtag
3. Client da riga di comando per essere aggiornati in tempo reale sui nuovi messaggi di un determinato topic.

L'applicazione deve essere tesata per il deployment su piattaforma GCP utilizzando i servizi visti a lezione:

- App Engine
- Firestore
- Pub/Sub

Parte 1 - Backend API REST

L'applicazione deve esporre le seguenti funzionalità tramite opportune Web API RESTful:

1. inviando richieste `POST` allo URI `/api/v1/chirps/` è possibile inserire un nuovo messaggio
2. inviando richieste `GET` allo URI `/api/v1/chirps/{id}` è possibile leggere un messaggio
3. inviando richieste `POST` allo URI `/api/v1/clean` è possibile cancellare i dati memorizzati nel database

L'interfaccia di utilizzo delle API deve soddisfare **rigorosamente** il file di specifica *OpenAPI* disponibile insieme alla presente specifica

Per testare la funzionalità dell'API è possibile far riferimento al seguente URL:

```
https://sacvalidator.appspot.com/
```

Parte 2 - Web Application

Realizzare tre pagine Web:

1. Una pagina con due form: uno per inserire un messaggio e uno per inserire un hashtag da cercare nel database
2. Una pagina da chiamare in seguito alla ricerca su hashtag che mostra tutti i messaggi con un dato hashtag

L'inserimento di messaggi si comporta come l'invocazione dell'API `POST` allo URI `/api/v1/chirps/`. Una volta inserito il messaggio si ritorna alla pagina di partenza semplicemente informando l'utente che il messaggio è stato inserito

La pagina per la ricerca degli hashtag deve contenere la lista di messaggi con l'hashtag indicato. Per la ricerca devono essere usate le strutture dati descritte nel seguito. Una scansione lineare dei messaggi è un errore **grave**.

Parte 3 - Interconnessione di servizi

Realizzare un sistema Pub/Sub che abbia un topic per ogni hashtag. Se i topic sono generati automaticamente al bisogno il punteggio della prova è superiore, ma è accettabile anche avere un set prefissato di hashtag.

Ogni volta che viene inviato un messaggio, l'Id del messaggio viene comunicato con un messaggio a tutti i topic corrispondenti agli hashtag nel messaggio

Realizzare un client da linea di comando che si invoca come segue:

```
python3 listen_hashtag.py '#topicname'
```

Una volta lanciato, il client si mette in attesa di messaggi con l'hashtag specificato nella linea di comando e mostra il testo di tutti i messaggi con tale hashtag mano a mano che arrivano le notifiche.

Note

- Se la validazione dell'API con il validatore non è positiva, l'esame è automaticamente insufficiente.
- In caso di validazione positiva il voto sarà basato su quante parti dell'esame sono state portate e termine e sulla qualità del codice e delle soluzioni proposte.
- Implementazioni non conformi alle specifiche (anche nelle altre parti del compito) comportano una penalità sul voto.

Dettagli implementativi

Per l'implementazione si ricorda che:

- Per la creazione dinamica di topic Pub/Sub si può fare riferimento alla seguente documentazione: https://cloud.google.com/pubsub/docs/create-topic. Si assume che i topic siano meno di 100.
- La funzione per estrarre la lista degli hashtag da un messaggio è fornita di seguito
- A livello di database devono essere create le seguenti collection: *messages* e *hashtags*
- Nella collection messages in cui ogni documento è un messaggio con un id generato automaticamente. Ogni documento deve contenere i seguenti campi:

  - message → testo del messaggio
  - hashtags → lista degli hashtags del messaggio
  - timestamp → momento di invio del messaggio
- Nella collection hashtags ogni documento corrisponde a un hashtag e il documento contiene la lista dei messaggi con tale hashtag nel formato: 'document_id: timestamp'

Funzione per estrarre i gli hashtag dei messaggi

```
import re

def get_hashtags(msg: str) -> list:
    return re.findall('(#\w+)', msg, re.DOTALL)
```