

Corso di laurea Magistrale in Ingegneria Informatica

DISTRIBUTED SYSTEMS AND BIG DATA

WeatherNotif

Aurora Tallarita -Giulio Samperi

Sommario

[**Introduzione** 3](#_Toc156826849)

[**Descrizione dell’applicazione** 3](#_Toc156826850)

[**Comando /sub** 4](#_Toc156826851)

# **Introduzione**

Si è scelto di implementare un applicazione distribuita composta da diversi microservizi, eseguiti su docker containers, seguendo la traccia del tema n.1: weather event notifier.

L’applicazione permette agli utenti, tramite botTelegram [WeatherNotif](https://github.com/T-Aurora/WeatherNotif), di ricevere notifiche sul meteo nelle città di interesse. L’utente può sottoscriversi richiedendo le notifiche per una città(ogni sottoscrizione riguarda una sola città), fornendo per ognuno una serie di condizioni da verificare ad esempio: Catania Rain Tempmax:30 Tempmin:20, qualora si verificasse anche solo una delle condizioni indicate nella sottoscrizione, il sistema restituirà una notifica all’utente. Il sistema recupererà le informazioni meteorologiche tramite l’API fornita OpenWeatherMap, filtrando ed elaborando i dati (eseguendo quindi un data scraping) in base alla richiesta dell’utente.

# **Descrizione dell’applicazione**

per lo sviluppo software come linguaggio è stato usato Python.

Nel microservizio TelegramNotif, si è deciso di utilizzare una classe RedisLink per sfruttare un db redis che crea una correlazione tra username, che verrà salvato assieme ad altri dati dell’utente in un db mysql, e il suo chat id che permette l’invio delle notifiche meteo tramite bot telegram in modo da disaccoppiarlo totalmente da altri microservizi.

Per reallizzare ciò si utilizza la libreria python-telegram-bot, usando il modulo telegram.ext per la gestione dei comandi tramite il bot, nel momento iniziale in cui l’utente avvia il bot ed esegue il comando /start verrà chiamato il metodo link\_user dell’istanza di RedisLink che andrà a creare una corrispondenza username/chat-id dell’utente. Inoltre, sono implementati tutti i comandi che l’utente avrà a disposizione per poter interagire con il sistema sfruttando il botTelegram.

---INSERIRE IMAG

## **Comando /sub**

Con il comando /sub si è implementata la funzione di gestione sottoscrizione alle informazioni meteorologiche per la città desiderata dall’utente, esempio /sub Catania Rain Tempmax:30 Tempmin:20

Prima di tutto si è deciso di impostare un controllo sulla città inserita facendo una banale query a Openweather in modo da verificare se l’API abbia informazioni disponibili per quella città, dopodiché si passa elaborare le relative informazioni alla sottoscrizione richiesta e vengono aggiunti i dati dell'utente e le informazioni di sottoscrizione a specifici endpoint attraverso richieste POST.

Immagine che contiene testo, schermata, Carattere

Descrizione generata automaticamente

Nel microservizio Wnotif, è stata implementata un’applicazione Flask che offre un servizio di backend per la gestione degli endpoint usati per manipolare i dati riguardo utenti e sottoscrizioni, nello specifico sono stati implementati i seguenti endpoint:

Add\_user: riceve i dati dell’utente tramite una richiesta POST, verifica se l’utente esiste già nel database altrimenti aggiunge un nuovo utente

Show\_users: genera una risposta contenente le informazioni su tutti gli utenti, non usato nel concreto, usato solo in debug/testing

Find\_user: riceve nome ed username dell’utente tramite richiesta POST, esegue la query e ne restituisce le informazioni sull’utente se esiste nel database.

Usato per verificare la corrispondenza username/chat-id

Add\_subb: riceve i dati della sottoscrizione tramite una richiesta POST e aggiunge una nuova sottoscrizione al database.

Show\_subs: riceve l’username dell’utente tramite una richiesta POST, restituisce le informazioni sulle sottoscrizioni dell’utente in formato json.

All\_sub: restituisce tutte le sottoscrizioni nel database in formato json, utilizzata dal microservizio ApiHandler per richiedere i dati da Openweather in base a quali città sono presenti nelle sottoscrizioni.

Immagine che contiene testo, schermata, Carattere

Descrizione generata automaticamenteNel microservizio ApiHandler, è stata implementata la funzione per processare le informazioni ottenute dalle sottoscrizioni ottenute dall’endpoint /all\_sub, prima di fare una query ad Openweather viene usato un database Redis come cache per evitare di fare query equivalenti per le stesse città presenti in più sottoscrizioni. Per ciascuna sottoscrizione, vengono valutate le diverse condizioni meteorologiche in base alle informazioni ottenute dai dati meteorologici di Openweather, se le condizioni sono soddisfatte, viene costruito un messaggio contenente le informazioni richieste solo delle condizioni che sono state soddisfatte (es. se la temperatura va sotto la temp minima specificata dall’utente ma quella massima viene ancora rispettata, l’utente verrà notificato solo della prima).

# **Diagramma Architettura**

