



Specifica Architettuale

Informazioni sul Documento

Versione	1.0.0
Approvatori	
Redattori	Francesco Protopapa Greta Cavedon
Verificatori	Michele Gatto
Uso	Interno
Distribuzione	Prof. Vardanega Tullio Prof. Cardin Riccardo Gruppo <i>DreamTeam</i>

e-mail: dreamteam.unipd@gmail.com



Registro delle Modifiche

Versione	Data	Nominativo	Ruolo	Descrizione
v0.0.3	2022-05-05	Greta Cavedon	Progettista	Stesura §; (Verificatore:)
v0.0.2	2022-05-04	Francesco Protopapa	Progettista	Stesura §2; (Verificatore: <i>Michele Gatto</i>)
v0.0.1	2022-04-30	Francesco Protopapa	Amministratore	Creazione scheletro documento e stesura §1; (Verificatore: <i>Michele Gatto</i>)



Indice

1	Introduzione	4
1.1	Scopo del Documento	4
1.2	Scopo del Prodotto	4
1.3	Glossario	4
1.4	Riferimenti	4
1.4.1	Normativi	4
1.4.2	Informativi	4
2	Architettura del Prodotto	5
2.1	Architettura generale	5
2.1.1	Schema	5
2.1.2	Descrizione	5
2.2	Architettura del Crawling Service	6
2.2.1	Descrizione	6
2.2.2	Diagrammi delle classi	6
2.2.3	Diagrammi di sequenza	6
2.3	Struttura messaggio SQS	8
2.3.1	Design pattern notevoli utilizzati	9
2.3.2	Schema del database	9
2.4	Architettura del Ranking Service	10
2.4.1	Descrizione	10
2.4.2	Diagrammi delle classi	10
2.4.3	Diagrammi di sequenza	10
2.4.4	Note sul processo di analisi	12
2.4.5	Design pattern notevoli utilizzati	13
2.4.6	Schema del database	13
2.5	Architettura del FrontEnd	14
3	Requisiti Soddisfatti	15

Elenco delle figure

1	Architettura generale	5
2	Crawling Service - Diagramma delle classi	6
3	Crawling Service - Diagramma di sequenza - 1	7
4	Crawling Service - Diagramma di sequenza - 2	7
5	Crawling Service - Diagramma di sequenza - 3	8
6	Crawling Service - Esempio di un messaggio SQS	8
7	Crawling Service - Schema ER del database	9
8	Ranking Service - Diagramma delle classi	10
9	Ranking Service - Diagramma di sequenza - 1	11
10	Ranking Service - Diagramma di sequenza - 2	12
11	Ranking Service - Schema ER del database	13



1 Introduzione

1.1 Scopo del Documento

Lo scopo di questo documento è di definire le norme, le convenzioni e le procedure adottate da tutti i membri di *DreamTeam*, in modo da poter definire un metodo di lavoro comune. Per raggiungere questo scopo ogni membro è tenuto a visionare periodicamente il documento e a rispettare tutte le norme in esso presenti. Per la redazione viene adottata una filosofia incrementale, quindi il documento allo stato attuale è incompleto e le norme saranno definite passo passo partendo dalle più urgenti, con l'aspettativa di avere un processo normato prima del suo avvio, considerando che, in generale, ogni norma può essere soggetta a cambiamenti.

1.2 Scopo del Prodotto

L'obiettivo di Sweeat e dell'azienda Zero12 è la creazione di un sistema software costituito da una Webapp. Lo scopo del prodotto è di fornire all'utente una guida dei locali gastronomici sfruttando i numerosi contenuti digitali creati dagli utenti sulle principali piattaforme social (Instagram e TikTok). In questo modo, è possibile realizzare una classifica basata sulle impressioni e reazioni di chiunque usufruisca dei servizi dei locali, non solo da professionisti ed esperti del settore.

1.3 Glossario

Per evitare ambiguità relative alle terminologie utilizzate è stato creato un documento denominato “*Glossario*”. Questo documento comprende tutti i termini tecnici scelti dai membri del gruppo e utilizzati nei vari documenti con le relative definizioni. Tutti i termini inclusi in questo glossario, vengono segnalati all'interno del documento con l'apice ^G accanto alla parola.

1.4 Riferimenti

1.4.1 Normativi

1.4.2 Informativi

2 Architettura del Prodotto

2.1 Architettura generale

2.1.1 Schema

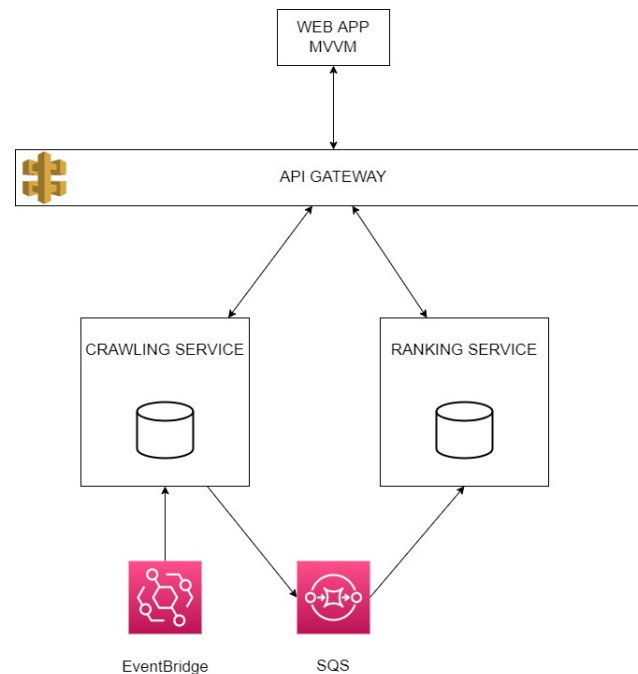


Figura 1: Architettura generale

2.1.2 Descrizione

Come richiesto dal capitolato si è deciso di utilizzare un'architettura a microservizi, i quali comunicano con il frontend tramite API gateway. In particolare sono stati individuati i seguenti microservizi:

- *Crawling Service*: questo microservizio si occupa di tutto ciò che riguarda il crawling dei dati da instagram. Il processo di crawling viene innescato da un servizio di AWS chiamato EventBridge che si occupa dello scheduling del crawling. Ogni volta che viene trovato dal crawler un post relativo ad un ristorante, questo viene inviato ad una coda di tipo SQS dalla quale andrà a leggere il servizio di ranking. Infine il Crawling Service espone una API al frontend per permettere di suggerire profili instagram da aggiungere alla lista di quelli osservati dal crawler.
- *Ranking Service*: questo microservizio invece si occupa dell'analisi dei contenuti estratti dal crawler e della realizzazione di una classifica di ristoranti. Il processo di analisi di un post viene fatto partire dalla ricezione di un messaggio sulla coda SQS, una volta letto il messaggio esso viene rimosso dalla coda ed analizzato. Infine il Ranking Service espone molteplici API al frontend in grado di fornire tutte le informazioni necessarie per poter visualizzare la classifica, i dettagli di un locale e la gestione dei preferiti.

Il frontend infine adotta un pattern MVVM e comunica col backend esclusivamente tramite API gateway. Grazie al servizio di AWS chiamato Cognito, tutta la parte relativa all'autenticazione è gestita lato frontend.

2.2 Architettura del Crawling Service

2.2.1 Descrizione

Il microservizio denominato *Crawling Service* si occupa principalmente di due funzionalità:

- Crawling dei dati: questo processo avviene tramite l'utilizzo della libreria Instagrapi. Ogni volta che viene chiamato il metodo `start_crawling` della classe `FacadeCrawling`, il sistema sceglie dal proprio database il profilo instagram che non viene guardato da più tempo e ne effettua il crawling dei dati. Per ogni media prelevato dal profilo, viene analizzata la sua location e nel caso in cui essa sia quella di un ristorante, il media viene salvato nel database ed inviato alla coda SQS in modo che possa essere analizzato dal servizio *Ranking Service*;
- Suggerimento dei profili: questa funzionalità viene esposta tramite una API rest al frontend al fine di fornire all'utente la possibilità di suggerire un profilo instagram sul quale successivamente il sistema andrà ad effettuare il crawling dei dati. Prima di essere aggiunto al database, viene controllato che il profilo non sia già presente, esista e sia pubblico. In base a ciascun esito verrà restituito il corrispettivo ritorno.

2.2.2 Diagrammi delle classi

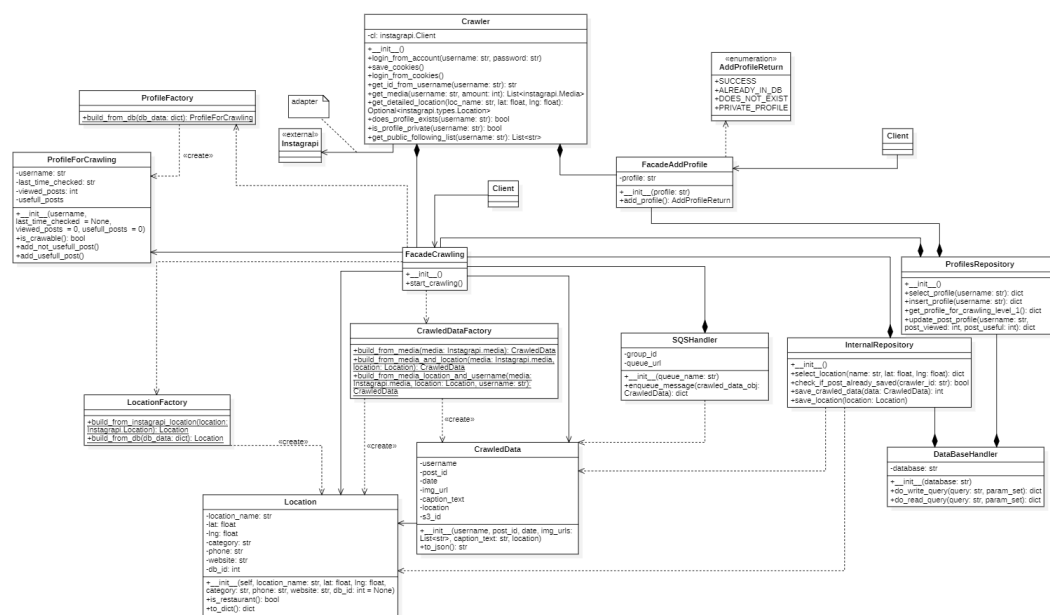


Figura 2: Crawling Service - Diagramma delle classi

2.2.3 Diagrammi di sequenza

In questa sezione vengono presentati i diagrammi di sequenza che modellano le operazioni principali del Crawling Service:

- il suggerimento di un profilo instagram da aggiungere alla lista dei profili su cui viene effettuato il crawling dei dati, nel caso in cui il profilo non sia già presente e sia pubblico;
- il processo di crawling dei dati;
- la formattazione di un singolo media ottenuto tramite crawling

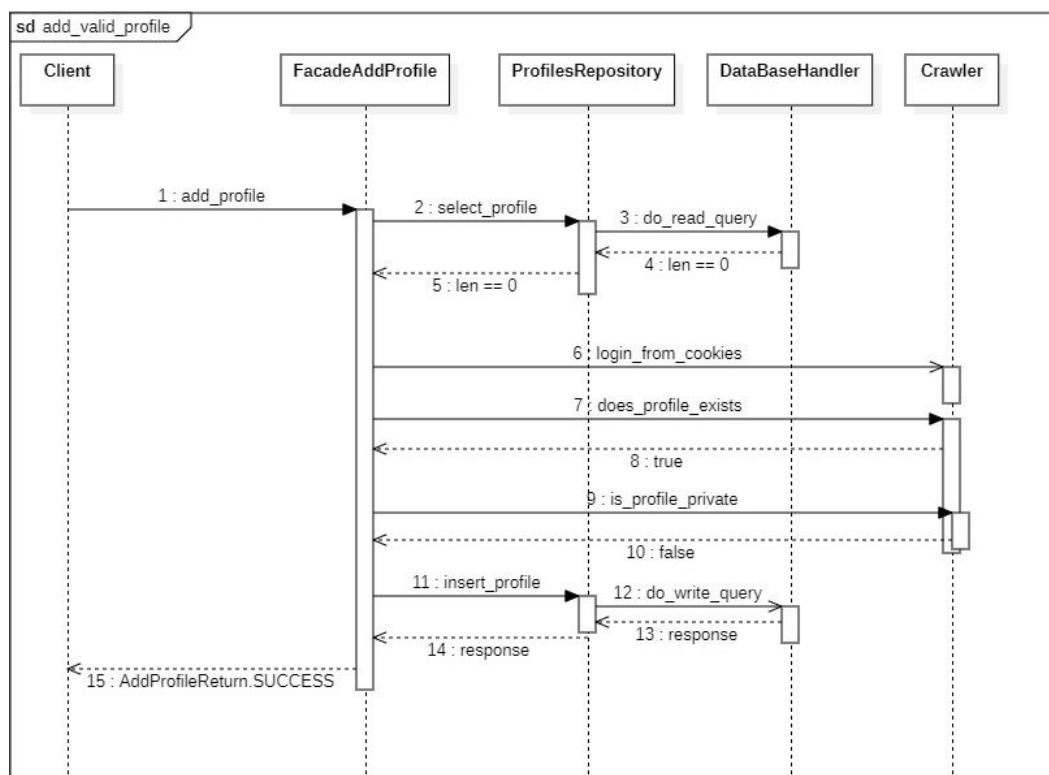


Figura 3: Crawling Service - Diagramma di sequenza - 1

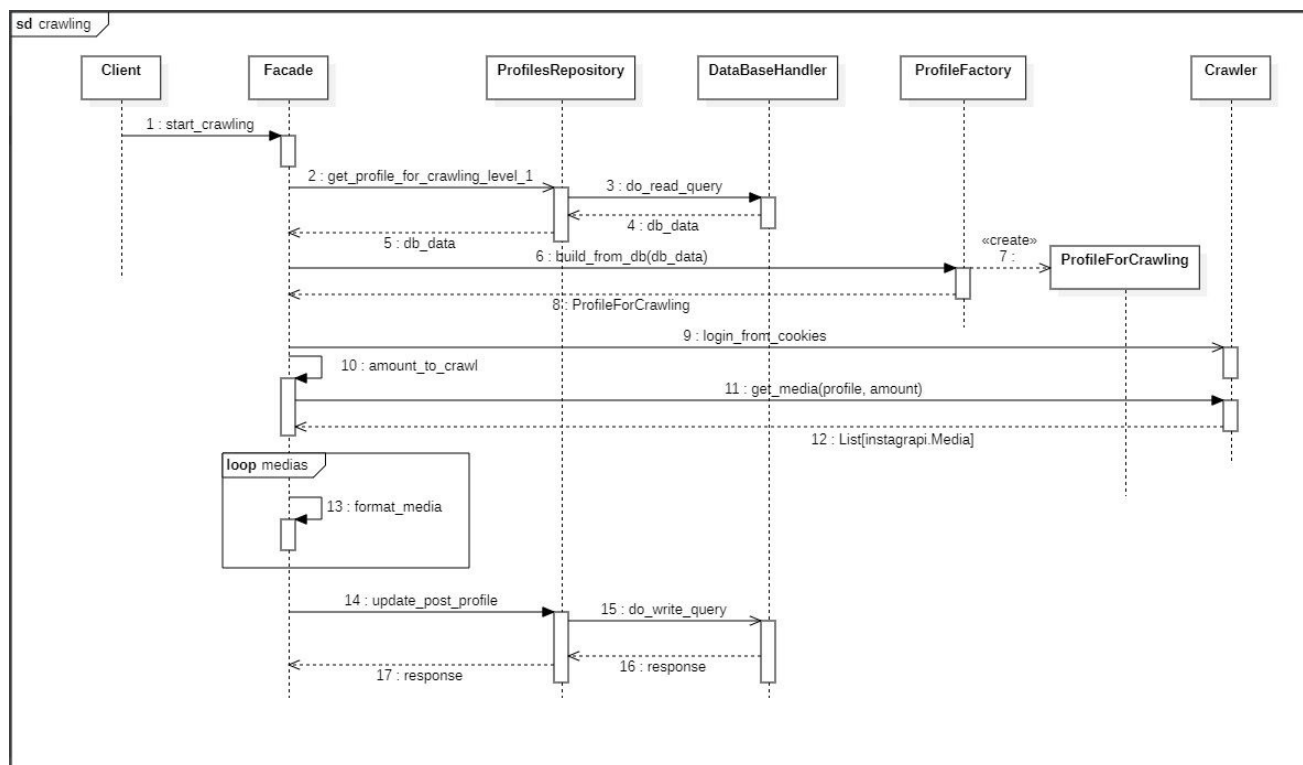


Figura 4: Crawling Service - Diagramma di sequenza - 2

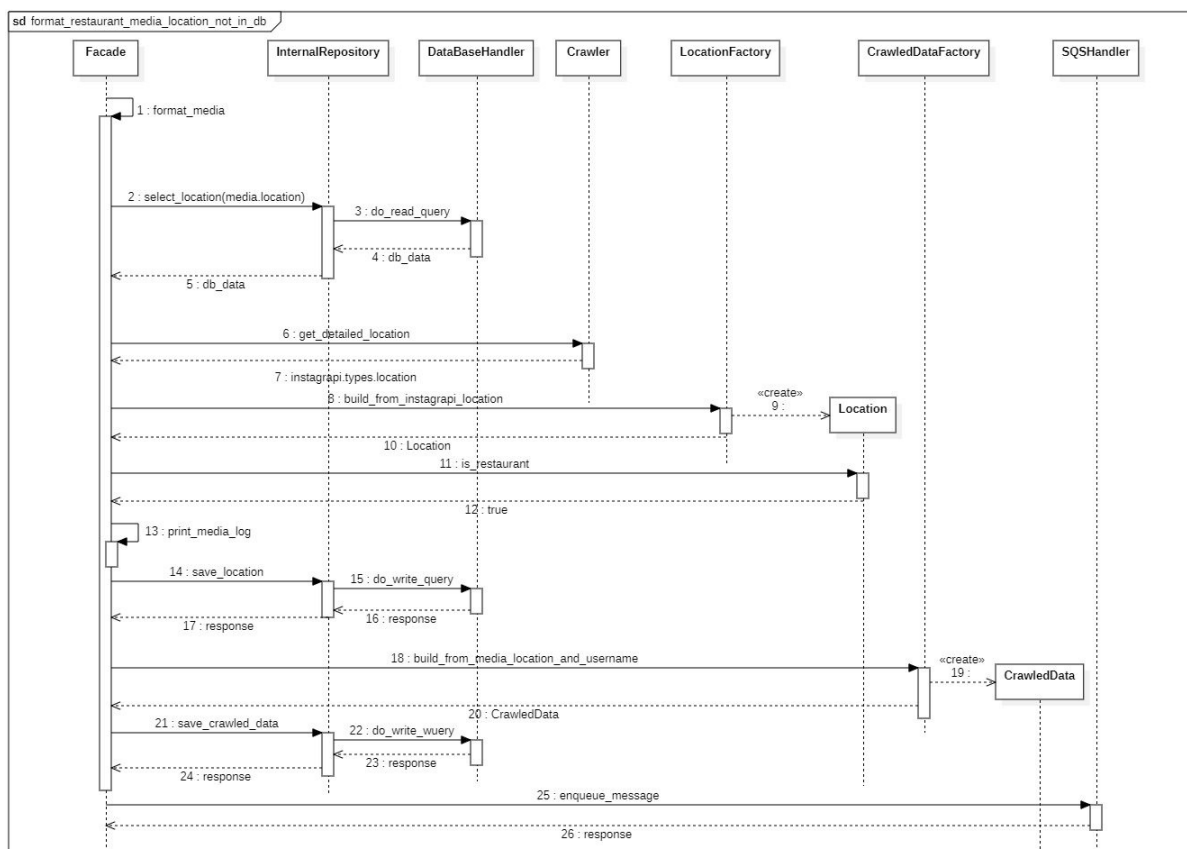


Figura 5: Crawling Service - Diagramma di sequenza - 3

2.3 Struttura messaggio SQS

```

{
  "messageId": "19dd0b57-b21e-4ac1-bd88-01bbb068cb78",
  "receiptHandle": "MessageReceiptHandle",
  "body": {
    "username": "lorenzolinguini",
    "post_id": "2806725869999084478_241913061",
    "date": "2022-04-01 10:10:18",
    "img_url": [
      44,
      123
    ],
    "caption_text": "Ask The Angels 🎸 #PattiSmith 🎸 🎸 \n\n Riassunto langarolo (potenziato) in sfoglia d'agnolotti al sugo d'arrosto 🍷 🍷 🍷 Poi Russa; Tonnato, Cruda & ogni sorta di beatitudine territoriale - confezionata al cesello - tra i capisaldi de @lapiolaaalba 🍷 🍷 🍷 Perché la tradizione, realizzata a queste altezze ascetiche, non riserva mai 'pesci d'Aprile' 🍷 🍷 🍷 Buon Pranzo & Buon WeekEnd Splendori 🍷",
    "location": {
      "location_name": "La Piola Alba",
      "lat": 44.7005,
      "lng": 8.036,
      "category": "Piedmont Restaurant",
      "phone": "0173442800",
      "website": "http://www.lapiola-alba.it/",
      "db_id": 8
    }
  },
  "attributes": {
    "ApproximateReceiveCount": "1",
    "SentTimestamp": "1523232000000",
    "SenderId": "123456789012",
    "ApproximateFirstReceiveTimestamp": "1523232000001"
  },
  "messageAttributes": {
    "md5ofBody": "{{md5_of_body}}",
    "eventSource": "aws:sqs",
    "eventSourceARN": "arn:aws:sqs:us-east-1:123456789012:MyQueue",
    "awsRegion": "us-east-1"
  }
}

```

Figura 6: Crawling Service - Esempio di un messaggio SQS

2.3.1 Design pattern notevoli utilizzati

Per La realizzazione del Crawling Service sono stati utilizzati i seguenti design pattern:

- **Facade:** Utilizzato per la realizzazione delle classi FacadeCrawling e FacadeAddProfile, in modo da fornire ai client un'interfaccia semplice ad un sottosistema molto complesso e disaccoppiando la logica di implementazione del sistema dal client.
- **Adapter:** Utilizzato dalla classe Crawler per disaccoppiare il resto del sistema dai metodi di instagrapi, rendendo disponibili solo quelli necessari tramite un'interfaccia nota al sistema.
- **Static Factory:** Utilizzato per fornire dei metodi statici in grado di creare oggetti di tipo CrawledData, Location, ProfileForCrawling a partire da altri tipi di oggetti.

2.3.2 Schema del database

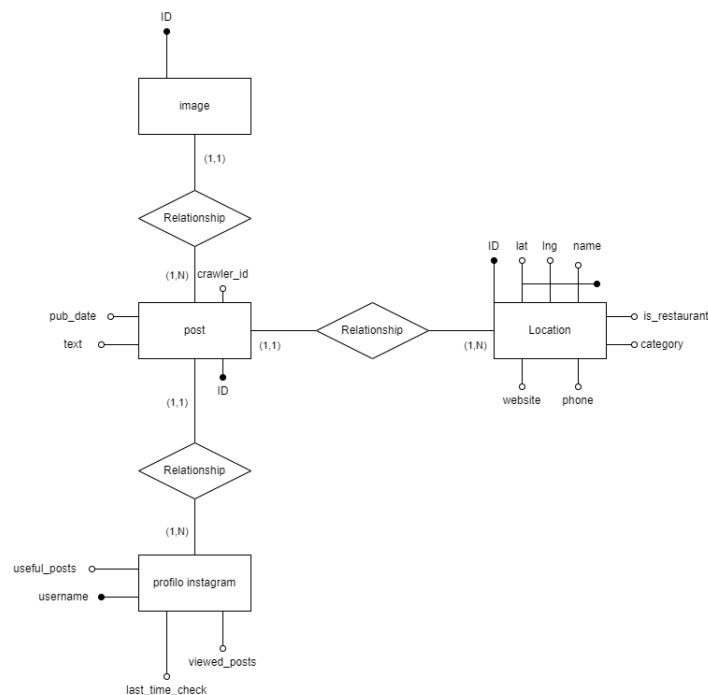


Figura 7: Crawling Service - Schema ER del database

2.4 Architettura del Ranking Service

2.4.1 Descrizione

Il microservizio denominato *Ranking Service* svolge le seguenti funzionalità:

- Gestione della classifica: ogni volta che un nuovo messaggio (struttura?) viene aggiunto alla coda SQS, viene chiamato il metodo **refresh_ranking** che si occupa di recuperare e fare la parsificazione e deserializzazione dei messaggi nella coda (lavorando a batch di massimo 10 messaggi). Dopo di che i messaggi vengono analizzati tramite i servizi Comprehend e Rekognition di AWS e vengono generati i punteggi per le foto, per il testo e per le emoji. Viene quindi aggiornato il database, aggiungendo il ristorante e il media (se non presenti) e aggiornati i punteggi. Vengono inoltre esposti due API endpoint:
- **getRanking**: restituisce una parte della classifica generale;
- **getLabelAndPost**: restituisce i media (e le relative labels) di un particolare ristorante.
- Funzionalità di ricerca: viene esposto un API endpoint **searchByName** che restituisce tutte le informazioni presenti nel database relative ad un ristorante con un nome specifico (media compresi);
- Gestione dei preferiti: viene esposto un API endpoint **favorites** che abilita alla gestione della lista di ristoranti preferiti per ogni utente, fornendo le funzionalità di aggiunta, rimozione e visualizzazione per la lista dei preferiti.

2.4.2 Diagrammi delle classi

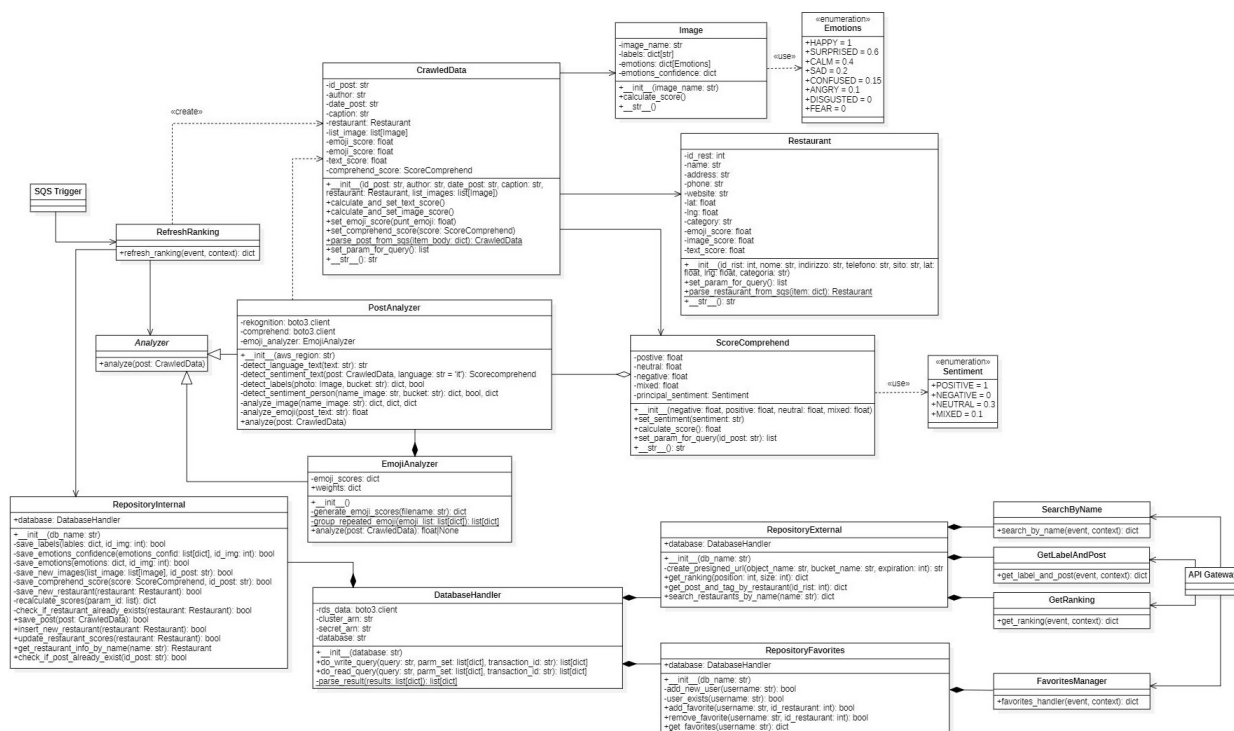


Figura 8: Ranking Service - Diagramma delle classi

2.4.3 Diagrammi di sequenza

In questa sezione vengono presentati i diagrammi di sequenza che modellano le operazioni principali del Ranking Service:

- Il processo di analisi di un media, assumendo che il media e il ristorante non siano già presenti nel database e che siano presenti persone delle immagini;

- Il processo di salvataggio del media analizzato e del ristorante, sempre assumendo che il media e il ristorante non siano già presenti nel database, e l'aggiornamento dei punteggi.

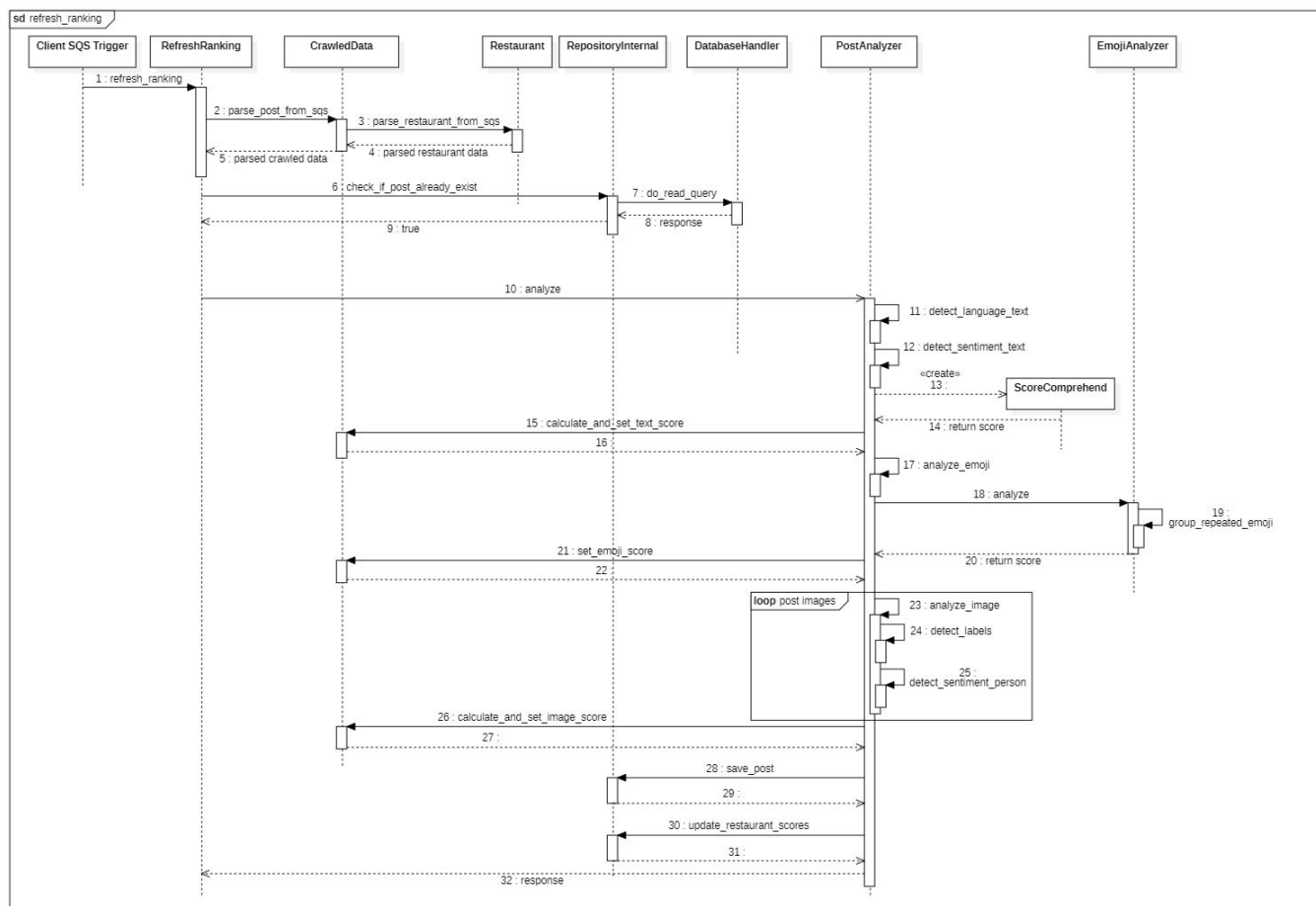


Figura 9: Ranking Service - Diagramma di sequenza - 1

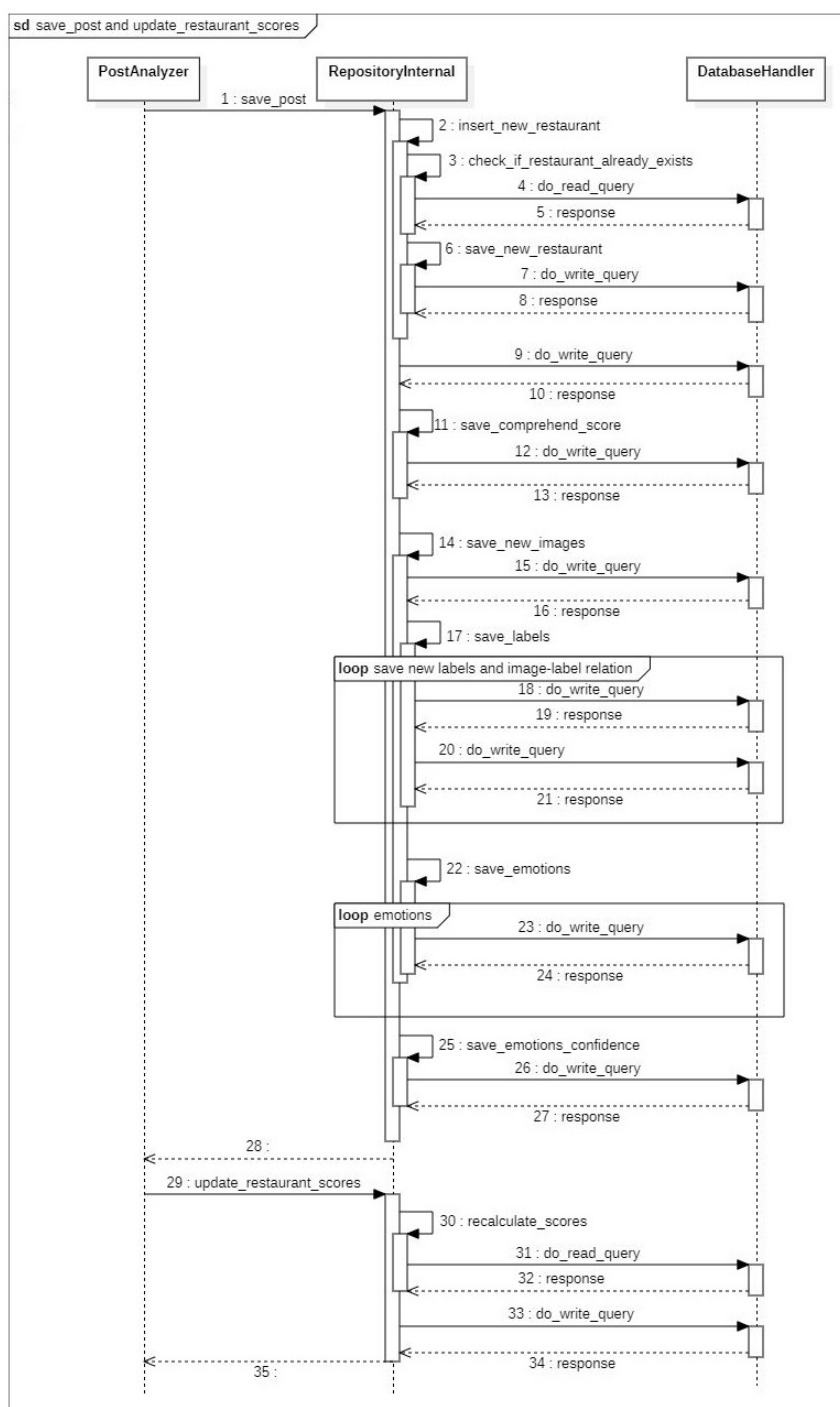


Figura 10: Ranking Service - Diagramma di sequenza - 2

2.4.4 Note sul processo di analisi

Dopo alcune prove ed osservazioni sono state fatte alcune decisioni relative al processo di analisi:

- Dopo l'analisi della immagini, verranno salvate nel database solo le label relative al cibo (quindi con padre "Food") e con confidenza di almeno il 90
- Solo se nelle immagini viene trovata la label "Person" verrà effettuata l'analisi dei sentimenti nell'immagine tramite Rekognition, anche in questo caso verranno salvati solo i sentimenti predominanti con una confidenza di almeno il 90

2.4.5 Design pattern notevoli utilizzati

Per La realizzazione del Ranking Service sono stati utilizzati i seguenti design pattern:

•

2.4.6 Schema del database

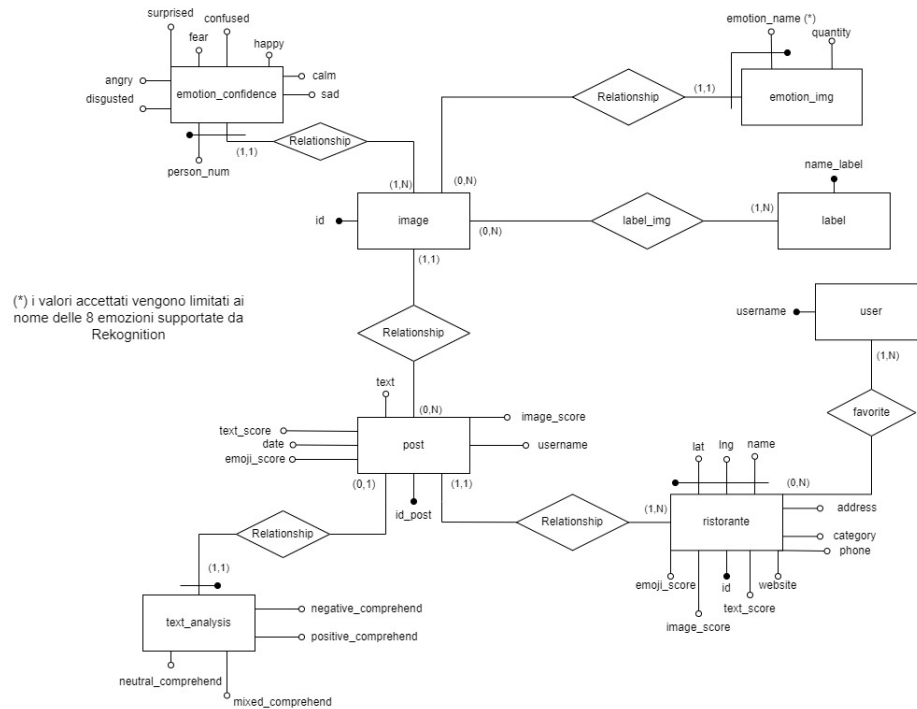


Figura 11: Ranking Service - Schema ER del database



2.5 Architettura del FrontEnd

3 Requisiti Soddisfatti

Fonte	Requisiti	Fonte	Requisiti
R1FW1	<i>Soddisfatto</i>	R2FW8.2	<i>Non Soddisfatto</i>
R1FW2	<i>Soddisfatto</i>	R2FW8.3	<i>Non Soddisfatto</i>
R1FW3	<i>Soddisfatto</i>	R2FW8.4	<i>Non Soddisfatto</i>
R1FE1	<i>Soddisfatto</i>	R1FW8.5	<i>Non Soddisfatto</i>
R1FE2	<i>Soddisfatto</i>	R3FW9	<i>Non Soddisfatto</i>
R1FE3	<i>Soddisfatto</i>	R3FW9.1	<i>Non Soddisfatto</i>
R1FE4	<i>Soddisfatto</i>	R3FW9.2	<i>Non Soddisfatto</i>
R1FE5	<i>Soddisfatto</i>	R1FW10	<i>Soddisfatto</i>
R1FE6	<i>Soddisfatto</i>	R2FE14	?????????
R1FE7	<i>Soddisfatto</i>	R1FW11	<i>Soddisfatto</i>
R1FW4	<i>Soddisfatto</i>	R2FW11.1	<i>Soddisfatto</i>
R1FW4.1	<i>Non Soddisfatto</i>	R2FW11.2	<i>Soddisfatto</i>
R1FE8	<i>Non Soddisfatto</i>	R2FW11.3	<i>Soddisfatto</i>
R1FW4.2	<i>Non Soddisfatto</i>	R2FW11.4	<i>Non Soddisfatto</i>
R1FE9	<i>Non Soddisfatto</i>	R1FW11.1.1	<i>Soddisfatto</i>
R3FW4.3	<i>Soddisfatto</i>	R2FW11.1.2	<i>Soddisfatto</i>
R3FE15	<i>Soddisfatto</i>	R2FW11.1.3	<i>Soddisfatto</i>
R1FW5	<i>Soddisfatto</i>	R3FW11.1.4	<i>Non Soddisfatto</i>
R1FW5	<i>Soddisfatto</i>	R2FW11.1.5	<i>Non Soddisfatto</i>
R2FE10	<i>Non Soddisfatto</i>	R2FW11.1.6	<i>Soddisfatto</i>
R2FE11	<i>Non Soddisfatto</i>	R1FW11.2.1	<i>Soddisfatto</i>
R2FE12	<i>Non Soddisfatto</i>	R1FW11.2.2	<i>Soddisfatto</i>
R1FW7	<i>Soddisfatto</i>	R1FW11.2.3	<i>Soddisfatto</i>
R1FW7.1	<i>Soddisfatto</i>	R1FW11.2.4	<i>Soddisfatto</i>
R1FW7.2	<i>Soddisfatto</i>	R2FW11.3.1	<i>Soddisfatto</i>
R2FW7.3	<i>Soddisfatto</i>	R2FW11.3.2	<i>Soddisfatto</i>
R2FW7.4	<i>Soddisfatto</i>	R2FW11.3.3	<i>Soddisfatto</i>
R1FW8	<i>Non Soddisfatto</i>	R2FW12	<i>Non Soddisfatto</i>
R2FE13	<i>Non Soddisfatto</i>	R2FW13	<i>Non Soddisfatto</i>
R1FW8.1	<i>Non Soddisfatto</i>	R3F1	<i>Non Soddisfatto</i>

Tabella 2: Tabella soddisfacimento requisiti