

**7-App:** semplice esempio di social media



- Landi Gennaro Francesco (Matricola 0212801127)
- Ferrandino Salvatore (Matricola 0212801123)
- Gravina Lorenzo (Matricola 0212801098)
- Cavallaro Matteo

Anno Accademico 2024/2025

<b>Executive Summary.....</b>	<b>3</b>
Tema e problema d'interesse.....	3
Obiettivo.....	3
Descrizione sintetica del sito.....	3
<b>Data warehousing.....</b>	<b>9</b>
<b>Usage Map e Ottimizzazione.....</b>	<b>11</b>
<b>Integrazione di MongoDB.....</b>	<b>15</b>
Motivazioni della scelta di MongoDB per contenuti “social” .....	15
Perché proprio i post?.....	15
Analisi dello schema relazionale esistente.....	15
Processo di migrazione (ETL).....	16
Modello orientato ai documenti: la collezione posts.....	17
Perché embedding?.....	17
5. Creazione e popolamento della collezione.....	17
Query di esempio in MongoDB.....	18
<b>Tabella di partecipazione al progetto.....</b>	<b>20</b>

# Executive Summary

## Tema e problema d'interesse

Oggigiorno, la soglia di attenzione è pari a circa sette secondi. Il mondo vuole soluzioni sempre più semplici a problemi sempre più complessi.

Anche i social media si muovono verso questa direzione, e noi, nel nostro piccolo, non vogliamo essere da meno. 7-App si propone come un social media molto semplice, nell'utilizzo e nelle funzionalità, che vertono essenzialmente su post (e commenti), feedback e messaggistica istantanea 1-to-1 e di gruppo.

## Obiettivo

Il nostro obiettivo è quello di supportare le operazioni quotidiane della piattaforma, ovvero sia gestire i dati relativi a profili utente, post, commenti, reazioni, connessioni, gruppi, messaggi privati, e fornire dati per algoritmi di raccomandazione e analisi dell'engagement.

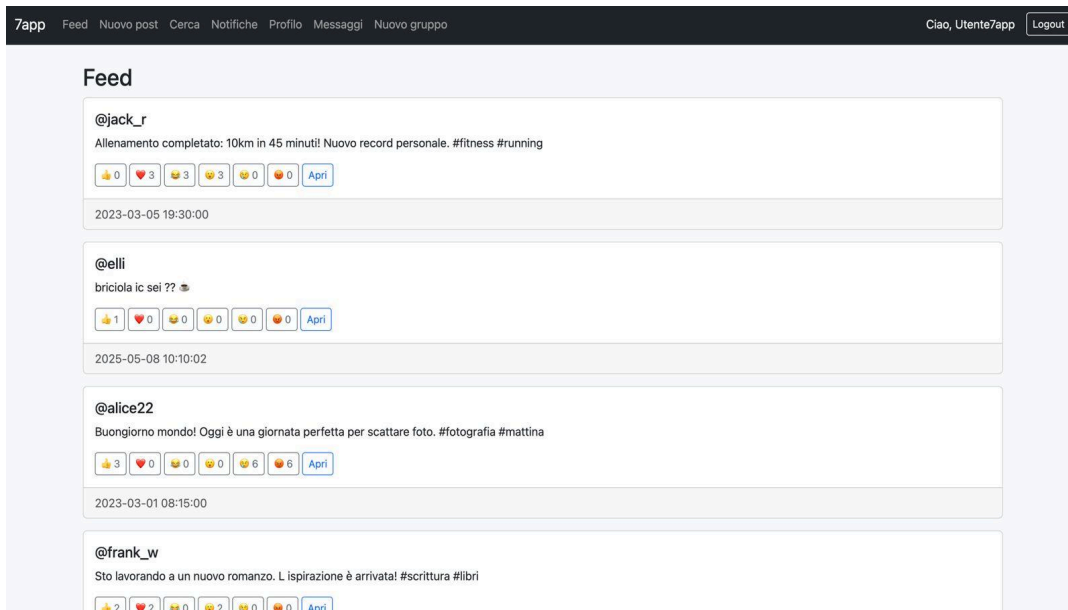
## Descrizione sintetica del sito

Il sito è un piccolo esempio di utilizzo previsto di un social media da parte di un utente.

Innanzitutto, vi è data la possibilità di effettuare login o registrazione per accedere al social o creare un nuovo account. Una volta effettuato l'accesso, è possibile poi visualizzare il feed relativo ai post pubblicati dagli account seguiti; qualora non ve ne siano di nuovi, l'algoritmo raccomanderà quei post che più hanno ricevuto feedback e commenti da parte dell'utenza. Inoltre, vi è la possibilità di inviare messaggi privati ad altri utenti - messaggistica 1-to-1 - o di creare e inviare messaggi in gruppi di conversazione con altri utenti.

Infine, peculiarità del sito è la presenza del sistema di notifiche, il quale, una volta che un utente effettua il login, avviserà l'utente stesso di eventuali commenti ai suoi post o nuovi messaggi ricevuti.

Per quanto riguarda le specifiche tecniche, esso è stato realizzato mediante il web-framework Flask. Scritto in Python, si tratta di un micro-framework semplice e minimalista, che non richiede particolari librerie o tool, ma che supporta a sua volta molteplici estensioni per varie finalità.



## Feed



## Pubblicazione post

**7app**FeedNuovo postCercaNotifiche●ProfiloMessaggiNuovo gruppo

## Notifiche

Message – @landigf ha scritto nel gruppo 51.2025-05-22 08:55:48

Message – @salferra02 ha scritto nel gruppo 51.2025-05-22 08:54:59

Message – @landigf ha scritto nel gruppo 51.2025-05-22 08:57:19

## Sistema di notifiche

**Cerca**

Utenti e Gruppi▼Cerca

Risultati per "Sa"

Utenti

@diana\_s  
Diana Santoro

Apri

@nina\_s  
Nina Sanchez

Apri

@quinn\_s  
Quinn Santini

Apri

@salferra02  
Salvatore Ferrandino

Apri

@sam\_t  
Samuele Trentini

Apri

Gruppi

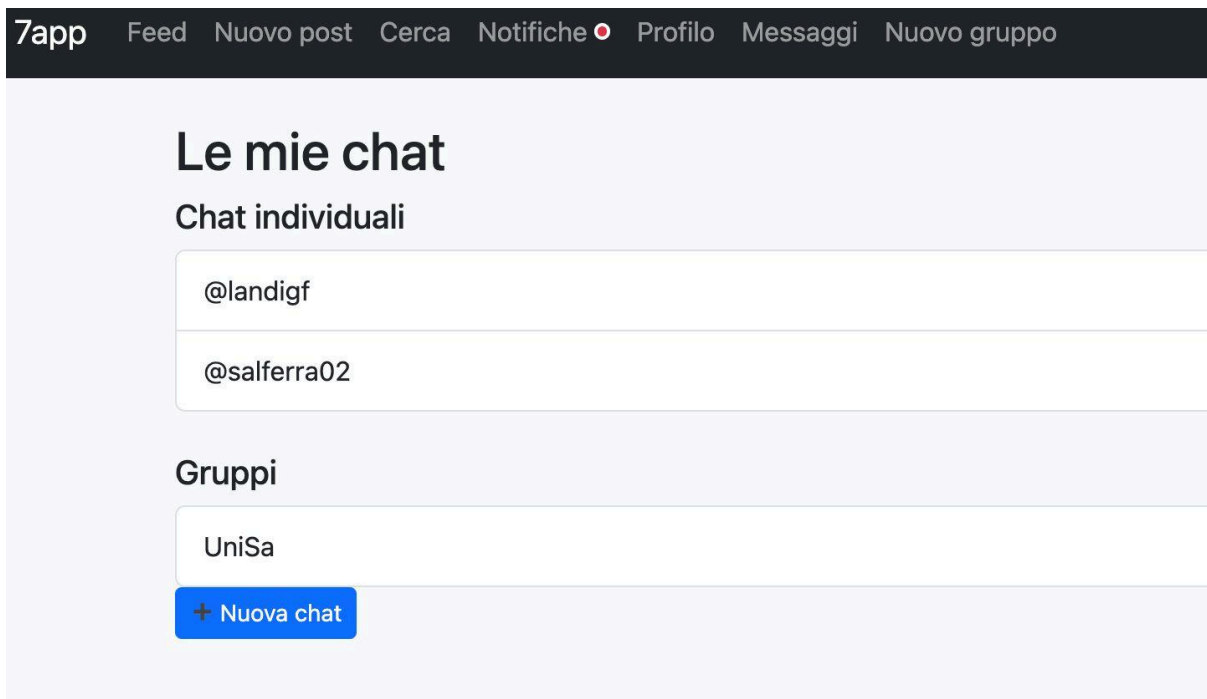
Fitness a Casa  
Allenamenti e consigli per restare in forma

Apri

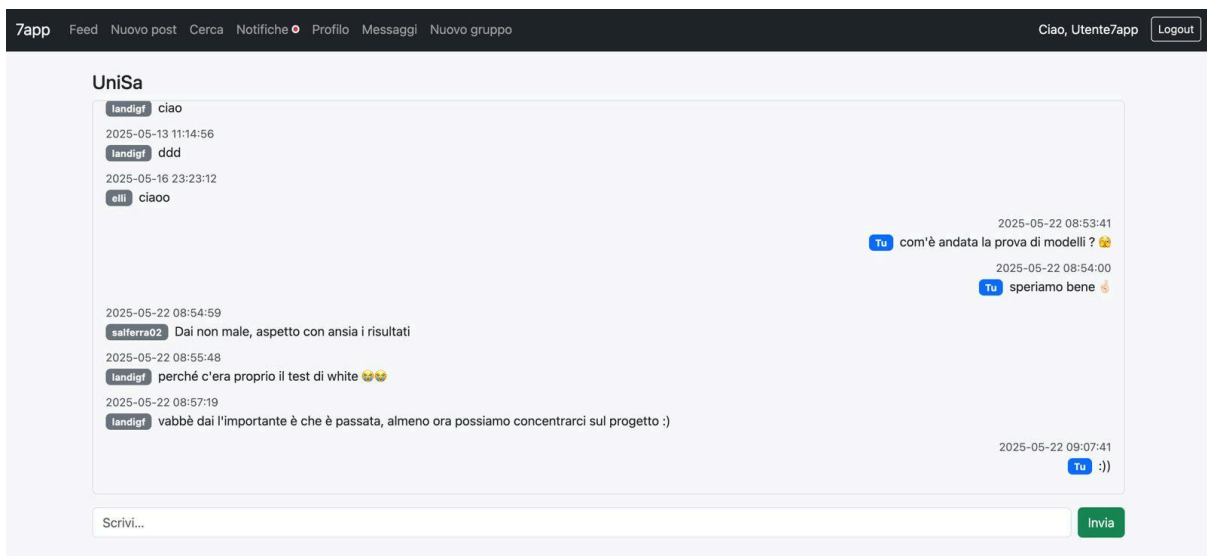
UniSa  
Gruppo Unisa

Apri

## Ricerca



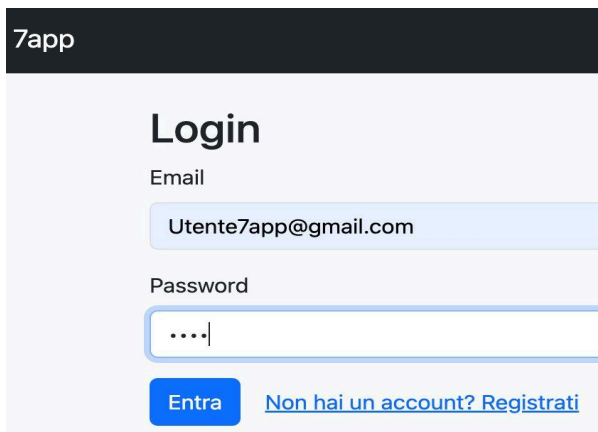
Schermata iniziale messaggistica



Visualizzazione messaggi di gruppo



Visualizzazione informazioni gruppo

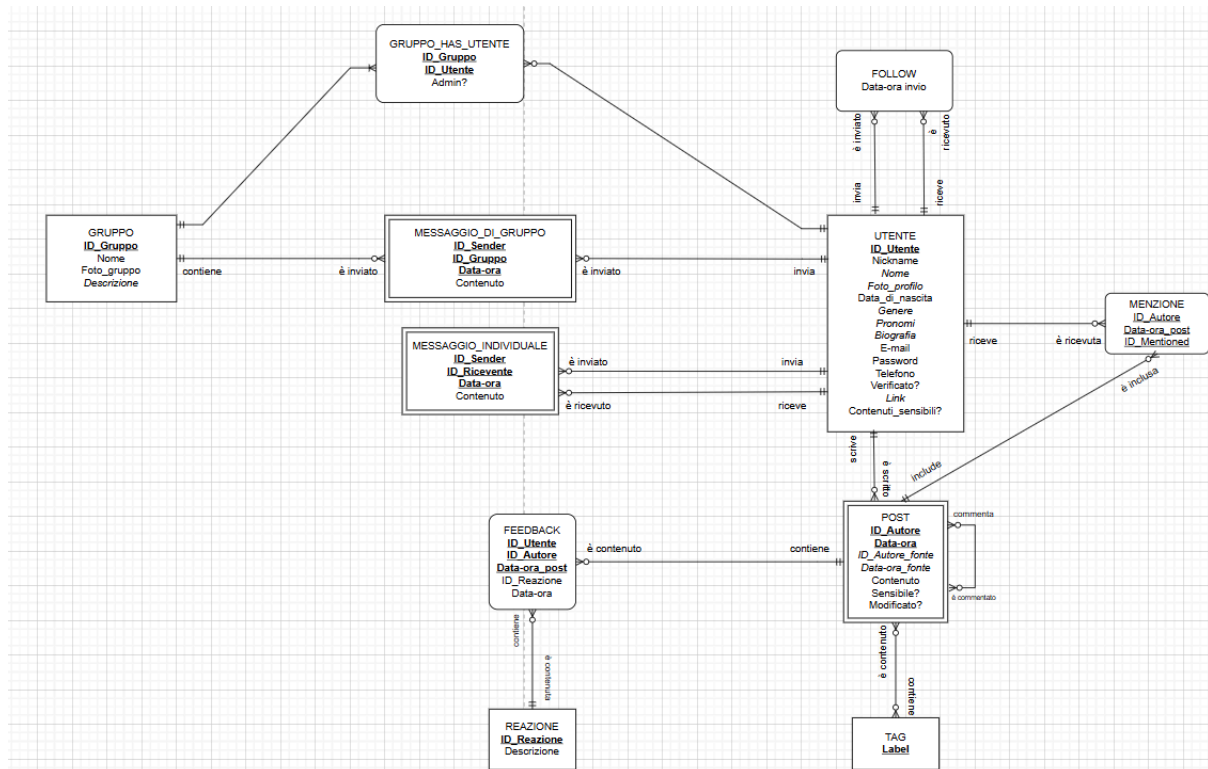
A screenshot of a web application's login page. At the top, there is a dark header with the text '7app' in white. Below the header, the word 'Login' is displayed in a large, bold, black font. Underneath 'Login', the word 'Email' is shown in a smaller font, followed by a text input field containing the email address 'Utente7app@gmail.com'. Below the email field, the word 'Password' is shown, followed by a password input field with four dots indicating masked characters. At the bottom of the login form, there is a blue button labeled 'Entra' and a blue hyperlink that reads 'Non hai un account? Registrati'.

È possibile provare la nostra app utilizzando queste credenziali:

Email: [Utente7app@gmail.com](mailto:Utente7app@gmail.com)

Password: 1234

# Diagramma EER



In fase di pianificazione, siamo giunti alla modellazione di questo diagramma EER. In 7-App, gli utenti possono seguire altri utenti, pubblicare post - o commenti a post, e questo viene tradotto nel diagramma con ID\_Autore\_fonte e Data\_ora\_fonte, che rappresentano l'ID dell'autore e il timestamp del post al quale si risponde - , arricchirli con tag o menzioni e fornire il proprio feedback tramite le reazioni - in questo caso, la tabella "Reazione" può anche essere vista come una lookup-table per i feedback - . Inoltre, 7-App presenta anche un sistema di messaggistica 1-to-1 e di gruppo, e ciò viene perfettamente rappresentato anche all'interno del diagramma.



# Modello relazionale

## UTENTE

(ID\_Utente, Nickname, Nome, Foto\_profilo, Data\_di\_nascita, Genere, Pronomi, Biografia, Telefono, Verificato, Link, Contenuti\_sensibili)

## UTENTE\_CREDENZIALI

(ID\_Utente, E-mail, Password) <- applicato partizionamento verticale rispetto ad UTENTE per motivi di sicurezza

## FOLLOW

(ID\_Sender, ID\_Followed, Data-ora)

## GRUPPO

(ID\_Gruppo, Nome, Foto\_gruppo, Descrizione)

## GRUPPO\_HAS\_UTENTE

(ID\_Gruppo, ID\_Utente, Admin)

## MESSAGGIO\_DI\_GRUPPO

(ID\_Sender, ID\_Gruppo, Data-ora, Contenuto)

## MESSAGGIO\_INDIVIDUALE

(ID\_Sender, ID\_Ricevente, Data-ora, Contenuto)

## POST

(ID\_Autore, Data-ora, ID\_Autore\_fonte, Data-ora\_fonte, Contenuto, Sensibile, Modificato)

## POST\_HAS\_TAG

(Label, ID\_Autore, Data-ora)

## MENZIONE

(ID\_Autore, Data-ora\_post, ID\_Mentioned)

## REAZIONE

(ID\_Reazione, Label)

## FEEDBACK

(ID\_Utente, ID\_Autore, Data-ora\_post, ID\_Reazione, Data-ora) - autore si riferisce all'autore del post

Sottolineatura: chiave primaria; *corsivo*: chiave esterna

Per motivi inerenti alla tutela dei nostri utenti, abbiamo deciso di operare un partizionamento verticale delle credenziali degli stessi rispetto alla relazione degli utenti. In particolare, abbiamo anche cifrato le password mediante hash SHA-256, in modo tale che esse non siano visibili in chiaro accedendo al database tramite il DBMS.

Per quanto concerne la progettazione fisica, essa è descritta nel file “creazione\_schema\_tabelle.sql” presente all’interno della cartella “creazione schema e tabelle originale - con popolamento” all’interno della sezione “TabelleSQL” tra gli artefatti del progetto. Nella cartella “TabelleSQL” è possibile trovare gli script SQL per la creazione dello schema e delle tabelle con popolamento annesso. In particolare, per quest’ultimo scopo è stata utilizzata l’IA generativa “DeepSeek”, specificando la query - e allegando lo script utilizzato per la creazione dello schema e delle tabelle:

“Avrei bisogno che tu mi popolassi questo database MySQL. Si tratta di un DB relativo a un progettino per un esame universitario che vuole provare a replicare il funzionamento di un social media. La parte relativa ai post gestisce anche le eventuali risposte ai post - descritte dalla chiave esterna - . Inoltre, l'attributo Contenuti\_sensibili per la relazione UTENTE è un booleano tale che è FALSE se l'utente in questione non vuole visualizzare contenuti marcati come sensibili sulla piattaforma, mentre è TRUE se invece dà il consenso per la visualizzazione di tali contenuti. Gli stessi post contengono un attributo Sensibile che specifica se essi siano contenuti sensibili oppure no. Vorrei che mi generassi circa 50 record per ogni relazione, senza però utilizzare processi randomici tramite MySQL stesso.”

## Query e funzionalità

In questa prima versione dell’applicativo - sia shell, sia web - vengono supportate alcune delle funzionalità essenziali previste dal nostro social. In particolare:

1. Possibilità di creare un account ed effettuare il login;
2. Possibilità di visualizzare il feed, costituito dagli ultimi post degli utenti seguiti e dai post più popolari;
3. Possibilità di creare post e interagire con essi, mediante reazioni e commenti;
4. Messaggistica 1-to-1 e di gruppo, con possibilità di creare gruppi di conversazione o entrare in gruppi già esistenti e di inviare messaggi;
5. Sistema di notifiche per messaggi e reazioni ricevute

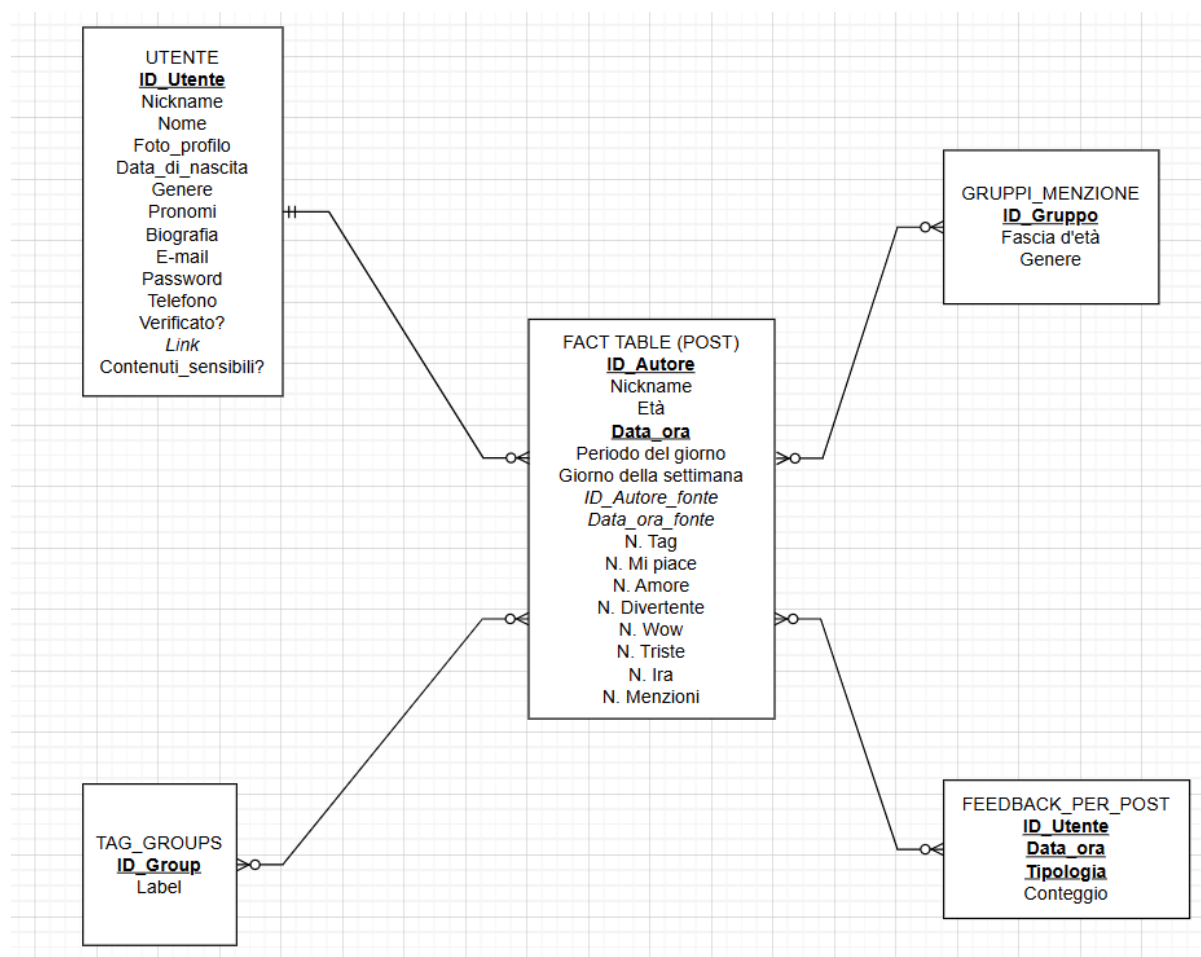
Tutta la parte relativa al login, al feed, alle notifiche e ai post con le relative interazioni è stata curata da *Gennaro Francesco Landi*, che si è anche occupato della realizzazione del prototipo di sito web, mentre tutta la sezione di messaggistica da *Salvatore Ferrandino*.

È possibile consultare le query in dettaglio sugli script .sql nella sezione “Query” degli artefatti.

# Data warehousing

Il data warehousing, nell'ottica di un social media, è molto importante al fine di aiutare il management a prendere decisioni cruciali, in particolare legate all'efficientamento degli algoritmi di raccomandazione e alle politiche relative alle inserzioni pubblicitarie.

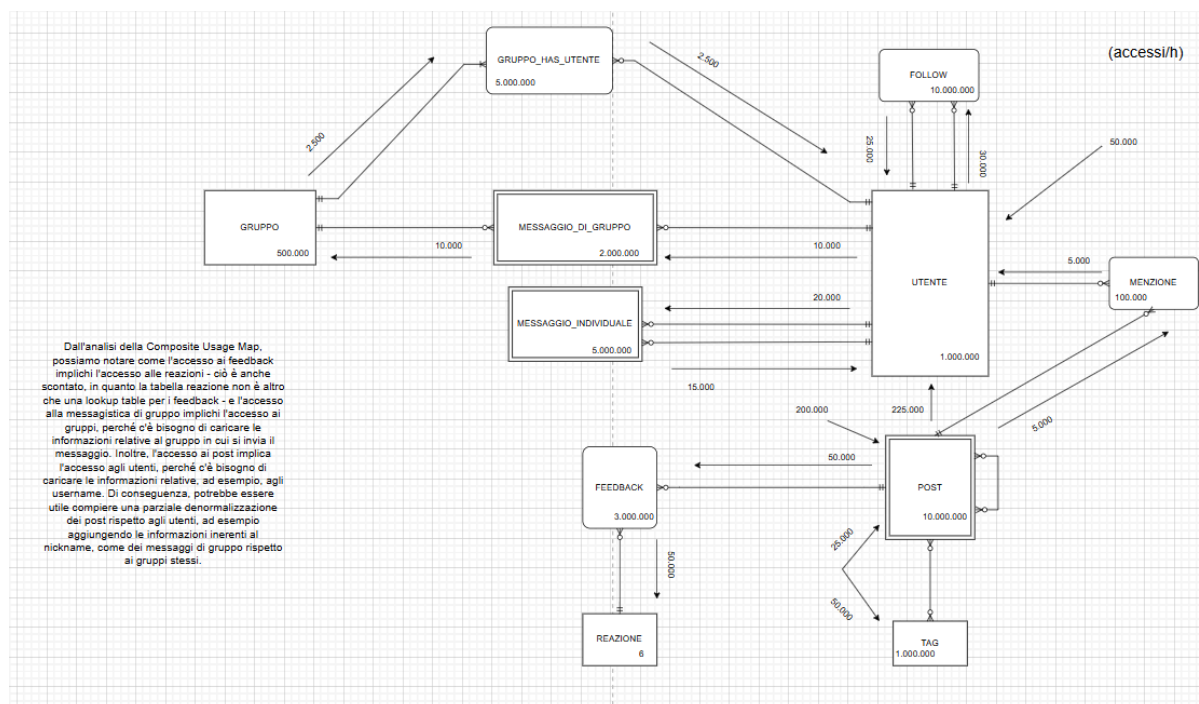
Ecco un possibile esempio di star schema:



In questo caso, la fact table individuata è un'estensione della tabella relativa ai post, contenendo essi la maggior parte delle informazioni essenziali al management per le analisi necessarie; le dimensioni individuate, invece, sono quelle relative ai singoli utenti, a gruppi di tag - raggruppati ad esempio per macrocategorie - , alle menzioni - in cui abbiamo ipotizzato un raggruppamento per fasce d'età e genere - e ai feedback per i singoli post.

Una volta estratta la fact table tramite MySQL, abbiamo poi esportato la stessa come .csv per poi importarlo in R come data.frame . Abbiamo scelto di utilizzare R proprio perché si tratta di un linguaggio orientato all'analisi statistica, che include al suo interno diversi strumenti e strutture dati ottimizzate per una numerosità molto ampia di dati. Inoltre, tramite l'utilizzo di RStudio è anche possibile produrre report automatizzati in formato .html o .pdf. Un esempio di applicazione è presente nella cartella "Data Warehousing" nella sezione degli artefatti.

# Usage Map e Ottimizzazione



Per come è stato realizzato il database e per quello che è il suo utilizzo pratico, che sia tramite shell o tramite webapp, l'accesso ai post implica anche l'accesso agli utenti, dal momento che è necessario accedere all'username dell'utente che ha pubblicato il post, così come l'accesso ai feedback implica l'accesso alla tabella reazione, dal momento che quest'ultima è una semplice look-up table per i feedback. Inoltre, la maggior parte degli utenti che accede ai follow tramite accesso dagli utenti accede nuovamente agli utenti stessi, e l'accesso alle menzioni tramite i post implica, ancora una volta, l'accesso alla tabella degli utenti.

L'accesso alla messaggistica di gruppo implica l'accesso alle informazioni dei gruppi stessi e, in un 25% dei casi, un co-accesso alla relazione "gruppo\_has\_utente" e alla tabella degli utenti. Inoltre, il 75% degli utenti che accede alla messaggistica individuale tramite gli utenti accede nuovamente alla tabella degli utenti.

Di conseguenza, un modo per risolvere il problema relativo ai co-accessi che riguardano, per la maggior parte, la tabella degli utenti, è quella di compiere una parziale denormalizzazione delle tabelle post, messaggio\_individuale, messaggio\_di\_gruppo, menzioni e follow, in modo tale che esse includano quantomeno, oltre agli ID degli utenti, i loro nickname. Tuttavia, a questo va introdotto un sistema di controllo relativo ai possibili cambiamenti dei nickname degli utenti, in modo tale da evitare il più possibile anomalie di modifica. Da non sottovalutare anche il cospicuo aumento nello spazio di archiviazione del database stesso.

Inoltre, dal momento che l'accesso alla tabella dei post è molto massiccio - non a caso, è la prima tabella cui si accede una volta effettuato il login, alla fine di visualizzare il feed - potrebbe essere molto utile procedere a indicizzare i post, magari per l'ID dell'autore. Questo renderebbe la ricerca più rapida, al costo di un aumento nello spazio richiesto nella memoria secondaria.

# Applicativo tramite Shell

Per utilizzare l'applicazione tramite Shell, è fortemente raccomandato seguire le procedure indicate nel readme nella sezione dedicata (Progetto), oltre che servirsi di un ambiente come Visual Studio Code. Una volta eseguiti i vari step ivi indicati, basterà aprire il file [menu.py](#) con il proprio editor preferito, e poi effettuare il login per avere un primo assaggio dell'esperienza su 7-App.

```
---- Benvenuto a 7app! ----

Seleziona operazione:
  0. Effettua il login
  1. Registrati a 7app!
  -1. Esci dall'applicazione :(
Scegli numero operazione: 0

Operazione selezionata: login

🔑 Login
Email: salfe@gmail.com
Password:
✅ Login riuscito! Bentornato/a @salferra02

🔔 Non ci sono nuove notifiche.

Seleziona operazione:
  0. Vai al feed
  1. Cerca utenti o gruppi
  2. Apri i messaggi
  3. Mostra il profilo
  4. Crea un nuovo post
  5. Visualizza notifiche
  -1. Esci dall'applicazione :(
Scegli numero operazione: █
```

```
Operazione selezionata: create_post

📝 Crea un nuovo post
Scrivi il contenuto del post:
> Buonasera a tutti!
Il contenuto è sensibile? (s/n): n
Aggiungi tag (separati da virgola, opzionale): goodvibes
Vuoi menzionare qualcuno? Inserisci gli ID separati da virgola (opzionale):
✅ Post pubblicato con successo!
```

```
--- CHAT INDIVIDUALI ---
1. elli - Ultimo messaggio: 2025-05-13 15:32:34
2. landigf - Ultimo messaggio: 2025-05-07 11:15:23

--- CHAT DI GRUPPO ---

Seleziona operazione:
  0. Visualizza le tue chat recenti
  1. Apri una chat
  2. Inizia una nuova chat
  3. Torna al menù principale
  -1. Esci dall'applicazione :(
Scegli numero operazione: 1

Operazione selezionata: open_chat

Numero della chat da aprire: 2

--- CHAT PRIVATA CON landigf ---
[2025-04-27 12:22:51] Tu: ciao
[2025-04-27 12:27:11] Tu: uee
[2025-04-27 12:28:31] Tu: prova
[2025-04-27 12:46:50] Tu: ciao
[2025-04-27 12:51:11] Tu: ueee
[2025-04-27 12:53:00] Tu: ueeeeeee
[2025-04-28 22:41:16] Tu: ueueeee
[2025-05-07 11:14:41] Tu: salvo prova
[2025-05-07 11:15:23] landigf: ueue

Seleziona operazione:
  0. Carica altri messaggi
  1. Invia un nuovo messaggio
  2. Torna al menù principale
  -1. Esci dall'applicazione :(
Scegli numero operazione: █
```

```
Operazione selezionata: send_message

Scrivi il messaggio (o 'annulla' per tornare indietro): Buonasera anche a te!

--- CHAT PRIVATA CON landigf ---
[2025-04-27 12:22:51] Tu: ciao
[2025-04-27 12:27:11] Tu: uee
[2025-04-27 12:28:31] Tu: prova
[2025-04-27 12:46:50] Tu: ciao
[2025-04-27 12:51:11] Tu: ueee
[2025-04-27 12:53:00] Tu: ueeeeeee
[2025-04-28 22:41:16] Tu: ueueeee
[2025-05-07 11:14:41] Tu: salvo prova
[2025-05-07 11:15:23] landigf: ueue
[2025-05-13 20:07:48] Tu: Buonasera anche a te!
✅ Messaggio inviato!
```

# Integrazione di MongoDB

## Motivazioni della scelta di MongoDB per contenuti “social”

- **Gestione di dati semi-strutturati**  
Commenti, reazioni, tag e metadata (es. immagini, geolocalizzazione) possono variare per ogni post. MongoDB non richiede uno schema rigido a priori: ogni documento può evolvere in modo indipendente.
- **Scalabilità orizzontale**  
Il volume di post e commenti cresce rapidamente. Grazie allo **sharding**, è possibile distribuire la collezione **posts** su più nodi, bilanciando il carico.
- **Performance su letture aggregate**  
Le operazioni tipiche aggregano post, commenti e reazioni: con i documenti embedded, queste query si eseguono in un unico round-trip, evitando join costosi.

### Perché proprio i *post*?

1. **Fulcro dell’engagement**: generano il maggior numero di interazioni utente–contenuto.
2. **Elevata flessibilità**: tag, menzioni, metadata multimediali e commenti possono cambiare forma senza alterare altri documenti.

## Analisi dello schema relazionale esistente

Tabella	Contenuto	Cardinalità tipica
POST	Metadati base (autore, timestamp, testo)	milioni di righe
POST_HAS_TAG	Associazione post ↔ tag	N-a-N
MENZIONE	Associazione post ↔ utente menzionato	N-a-N
FEEDBACK	Reazioni (like, love...) degli utenti ai post	N-a-N
...	(altre: UTENTE, RELAZIONI, MESSAGGI, ecc.)	...

Per interrogazioni come “mostra post + commenti + conteggio reazioni” erano necessari 3–4 join, penalizzando performance e complessità. MongoDB consente di raggruppare dati correlati in un singolo documento.

## Processo di migrazione (ETL)

### 1. ESTRAZIONE

Query SQL per recuperare, per ogni post, campi base (**POST**), tag (**POST\_HAS\_TAG**), menzioni (**MENZIONE**), reazioni (**FEEDBACK** + **REAZIONE**) e commenti (registrati come record figli in **POST** o tabella dedicata).

SQL

```
SELECT p.ID_Autore, p.Data_ora, p.Contenuto, p.Sensibile,
p.Modificato,
  GROUP_CONCAT(DISTINCT t.Label)      AS tags,
  GROUP_CONCAT(DISTINCT m.ID_Mentioned) AS mentions,
  COUNT(f.ID_Reazione)                AS reactionsCount
FROM POST p
LEFT JOIN POST_HAS_TAG t ON ...
LEFT JOIN MENZIONE m ON ...
LEFT JOIN FEEDBACK f ON ...
GROUP BY p.ID_Autore, p.Data_ora;
```

### 2.

#### TRASFORMAZIONE

- Timestamp convertiti con **ISODate(...)**
- Sub-documento **author: { id, nickname }**
- Array **tags, mentions**; oggetto **reactions** con breakdown per tipo
- Array **comments** di sub-documenti

### 3. CARICAMENTO

Il caricamento avviene tramite lo script Python **ETL\_posts.py**, che utilizza **pymongo** per eliminare la collezione **posts** preesistente e inserire in bulk tutti i documenti trasformati.

## Modello orientato ai documenti: la collezione **posts**

Unset

```
{
  _id: ObjectId(),           // generato da MongoDB
  author: { id: Number, nickname: String },
  content: String,
  createdAt: ISODate,
  source: { authorId, createdAt } | null,
  sensitive: Boolean,
  edited: Boolean,
  tags: [ String, ... ],
  mentions: [ Number, ... ],
  reactionsCount: Number,
  reactions: { like: Number, love: Number, haha: Number, ... },
  comments: [
    {
      _id: ObjectId(),
      author: { id, nickname },
      content: String,
      createdAt: ISODate,
      likesCount: Number
    }, ...
  ]
}
```

### Perché embedding?

- Accesso frequente in lettura: post e commenti quasi sempre richiesti insieme.
- Dimensione documento fino a 16 MB: adeguata a decine di migliaia di commenti.

## Query di esempio in MongoDB

1. **Ultimi N post** (ordinati per data decrescente, con conteggio commenti e reazioni):

Unset

```
db.posts.aggregate([
```



```

    { $project: {
      _id: 0,
      authorId: "$author.id",
      content: 1,
      commentsCount: { $size: { $ifNull: ["$comments", []] } },
    },
    reactionsCount: 1,
    createdAt: 1
  }
},
{ $sort: { createdAt: -1 } },
{ $limit: N }
]);

```

2.

**Post il cui content contiene una stringa (case-insensitive):**

Unset

```

db.posts.find(
  { content: { $regex: "<stringa>", $options: "i" } },
  { "author.id": 1, content: 1, createdAt: 1 }
).sort({ createdAt: -1 });

```

3.

**Post con commenti contenenti una *'data'* stringa:**

Unset

```

db.posts.aggregate([
  { $match: { $text: { $search: "stringa" } } },
  { $project: {
    _id: 0,
    postContent: "$content",
    matchingComments: { $filter: {
      input: "$comments",
      as: "c",
      cond: { $regexMatch: {

```

```

        input: "$$c.content",
        regex: "stringa",
        options: "i"
      }}
    }}
  }
}
]);

```

4. **Post di un autore specifico (AUTHOR\_ID):**


```

Unset
db.posts.find(
  { "author.id": AUTHOR_ID },
  { content: 1, createdAt: 1 }
).sort({ createdAt: -1 });

```

# Tabella di partecipazione al progetto

La parte di progetto relativa a pianificazione e progettazione logica e fisica è stata equamente partecipata da parte dei membri di tutto il team; in particolare, per quanto riguarda gli altri riconoscimenti, :

 Nome	Matricola	Compiti svolti
SALVATORE FERRANDINO	0212801123	Realizzazione funzionalità di messaggistica e relativa integrazione con Python; Realizzazione Composite Usage Map e relative dissertazioni; Progettazione Star Schema, creazione fact table e conduzione di analitiche elementari; Query MongoDB # 3 e #4
GENNARO FRANCESCO LANDI	0212801127	Realizzazione applicativo shell (Python) e webapp, con relativi readme; Realizzazione funzionalità relative a login, registrazione, feed, post, commenti, reazioni e sistema di notifiche; Progettazione collezione post in MongoDB; Query MongoDB #1 e #2
LORENZO GRAVINA MATTEO CAVALLARO		Aggiunta nuove features e query MySQL
Tutto il gruppo		progettazione concettuale, brainstorming, confronto continuo