### Gruppo di lavoro:

Sofia Manno 1000067618, Giuliano Sicali 1000014800

### Abstract

Il progetto riguarda la creazione di un sistema distribuito per la gestione di informazioni finanziarie. La soluzione si basa su un’architettura a microservizi (sviluppati come container docker e gestiti ed eseguiti con docker compose), con un server gRPC che rappresenta l’interfaccia tra l’utente e l’intero sistema. Riceve le richieste dall’utente e interagisce con il database per fornire le funzionalità di gestione degli utenti (registrazione, aggiornamento e cancellazione) e di recupero di dati finanziari da yfinance (recupero dell’ultimo valore disponibile e calcolo della media degli ultimi x valori). Le operazioni di gestione degli utenti nel server gRPC devono essere implementate con una politica “at-most-once”, per evitare la duplicazione delle richieste.

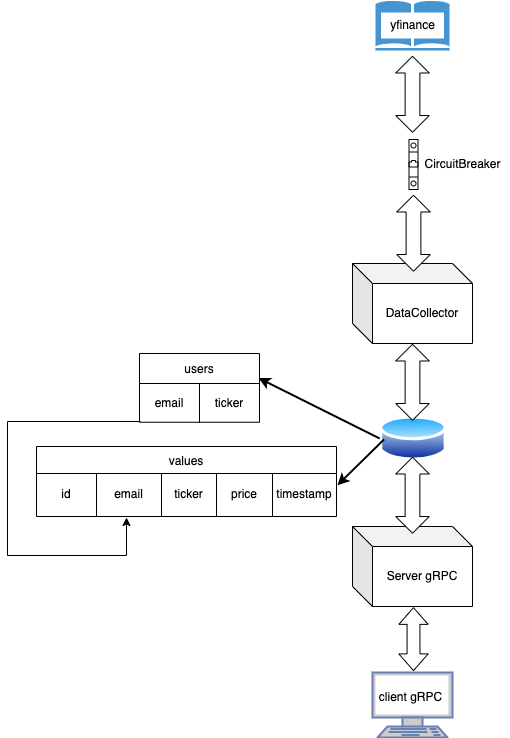
Il DataCollector è microservizio che esegue periodicamente la lettura della lista degli utenti dal database MySQL (tabella: *users*) e recupera i ticker azionari associati. Utilizzando la libreria yfinance, il servizio ottiene l'ultimo valore disponibile per ciascun titolo azionario monitorato dagli utenti e salva i risultati tabella nel database dedicata (*stock\_values*). Le chiamate verso yfinance sono protette da un Circuit Breaker, per gestire errori o ritardi nelle risposte. Nel database, ogni utente è identificato tramite il proprio indirizzo email a cui è associato un ticker (codice dell’azione).

È stato realizzato un client dotato di un’interfaccia a menù testuale, accessibile tramite terminale, che consente agli utenti di effettuare le operazioni di cui sopra.

### Diagramma Architetturale

Abbiamo scelto di suddividere il sistema in due microservizi distinti: Server e DataCollector. Questi sono supportati e comunicano tramite database MySQL containerizzato tramite Docker e sono gestiti all'interno di Docker Compose.

Il DataCollector recupera email e ticker dalla tabella users nel Database e crea un dizionario di ticker che viene usato per recuperarne i valori da yfinance che verranno inseriti nella tabella stock\_values.

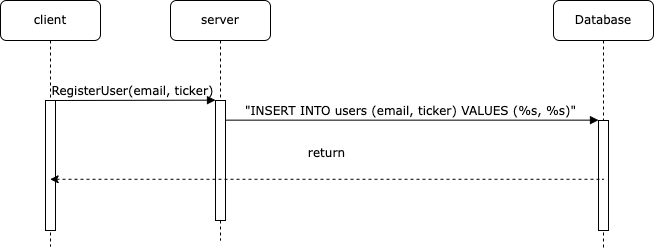


### Diagramma delle interazioni

#### Registra Utente

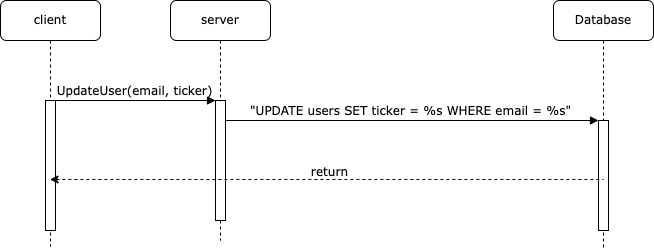
Il sistema implementa una cache per gestire le operazioni utente (registrazione, eliminazione, aggiornamento) secondo la politica "at-most-once", che garantisce che ogni operazione venga eseguita non più di una volta. La cache utilizza tre dizionari: uno per ciascuna operazione. Nei dizionari di registrazione ed eliminazione, la chiave è l'e-mail dell'utente, e il valore rappresenta lo stato: 0 (operazione in corso) o 1 (operazione completata). Se un'operazione non è in cache, si procede accedendo al database.

Tuttavia, per risolvere il problema di un’ipotetica *eliminazione → registrazione → eliminazione* dello stesso utente (dove la cache della prima eliminazione bloccherebbe la seconda), è necessario che, durante una registrazione, vengano rimossi dalla cache eventuali riferimenti all’eliminazione dell'utente, garantendo la corretta esecuzione delle nuove richieste.



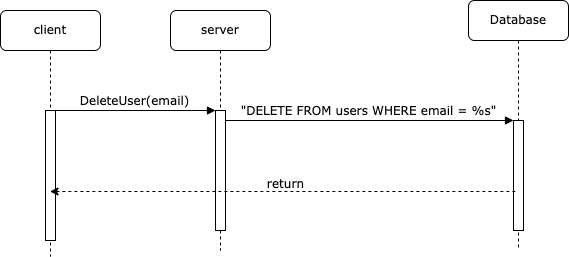
#### Modifica Utente

Per gestire le operazioni di aggiornamento, è importante garantire che ogni richiesta sia unica anche quando riguarda lo stesso utente. Per questo motivo, anziché utilizzare unicamente l'email come identificatore, si adotta una combinazione di due elementi: l'email e il ticker. Questo sistema permette di trattare ciascun aggiornamento come un'operazione indipendente. Anche in questo caso è stata usata la cache per l’implementazione della politica at-most-once.



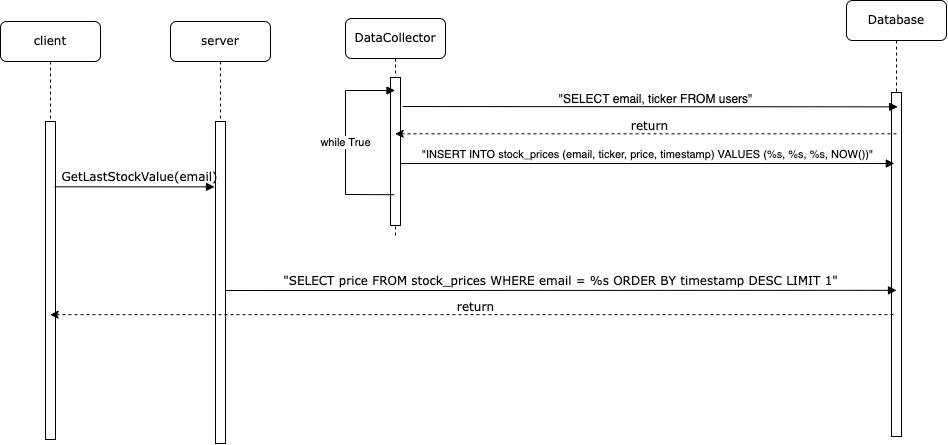
#### Elimina Utente

Il server verifica prima in cache e poi elimina il record dalla tabella users. Anche qui viene gestito il problema di un’ipotetica *registrazione→ eliminazione→ registrazione* (durante un’eliminazione, vengono rimossi dalla cache eventuali riferimenti alla registrazione dell'utente).



#### Recupero dell’Ultimo Valore Disponibile

L'utente richiede il valore più recente di un titolo azionario specifico. Il server esegue una query nella tabella *stock\_values* per recuperare il 'price’ associato all'email specificata dal client avente timestamp più recente.



#### Calcolo della Media degli Ultimi X Valori

Il client in fase di richiesta specifica la variabile ‘x’: numero di valori ‘price’ più recenti di cui fare la media. Il server esegue una query nella tabella stock\_values per raccogliere gli ‘x’ *price* con timestamp più recente associati all’email e viene calcolato e restituito il valore medio.

