

Bozza dell'idea progettuale - Homework 1- Distributed Systems and Big Data

Da ALESSIO GIORDANO <alessio.giordano@studium.unict.it>

Data mar 19/11/2024 14:11

A Antonella Di Stefano <antonella.distefano@unict.it>; Giovanni Morana <giovanni.morana@unict.it>;
MASSIMO GOLLO <massimo.gollo@phd.unict.it>

Cc LUCA MONTERA <uni384227@studium.unict.it>

 1 allegato (1 MB)

Why Finance - Homework 1 - Distributed Systems and Big Data.pdf;

Salve, siamo i vostri studenti Alessio Giordano (1000069222) e Luca Montera (1000069226) e vi sottoponiamo la bozza del sistema distribuito che intendiamo sviluppare insieme secondo il documento "Progetto #1 Prova in itinere DSBD aa2024-2025".

Vi invitiamo a fare riferimento al documento allegato per visualizzare i componenti del sistema.

Per prima cosa abbracciamo completamente la filosofia di progettazione a microservizi: le due funzionalità richieste per il server sono state scomposte in due servizi distinti che operano all'interno del loro container Docker definito come parte del compose.yaml; questi sono esposti all'esterno mediante un proxy HTTP, da realizzare attraverso il server web Python Flask (con il quale entrambi noi due abbiamo familiarità) e funge da client gRPC per questi microservizi, che implementeranno le interfacce del server gRPC.

Entrambi i microservizi di "AUTH" o gestione degli account utente e "WATCH" di monitoraggio dei titoli di borsa, saranno implementati in Python usando la stessa libreria vista a lezione e, rispettivamente implementano le funzionalità richieste nel documento di assegnazione del progetto.

La funzionalità di crawling dei titoli di borsa è implementato in un microservizio separato, qui chiamato "CRAWL", anch'esso scritto in Python per usare agevolmente la libreria "yFinance" richiesta, il cui fine è leggere dal microservizio di persistenza i ticker dei quali è richiesto aggiornare i valori, quindi richiedere al servizio esterno per ciascuno di questi il valore, e conservarlo all'interno della persistenza.

Il microservizio di persistenza è un server SQL in un container separato, con una tabella definita con i campi (email, ticker) che viene letta e scritta dal servizio AUTH e letta dal "CRAWL", e una definita con (ticker, value, timestamp) scritta dal "CRAWL" e letta dal "WATCH".

Per quanto riguarda per le funzionalità di robustezza quali politica at-most-once e circuit breaker, intendiamo centralizzarle in due microservizi (l'abbraccio di cui menzionavo sopra è totale 😊), rispettivamente un PROXY che funge da client gRPC e riconosce le richieste duplicate e fornisce loro un valore in cache al fine di evitare modifiche inconsistenti al sistema, mentre tutti gli accessi a servizi esterni vengono richiesti a un microservizio (comunicando attraverso gRPC, il terzo protobuf che scriveremo) "C.BREAKER" che si occuperà di mantenere lo stato di circuito aperto/chiuso/semi-aperto per origin contattata (nel senso tecnico di host-path degli URI) ed eventualmente restituire errore immediatamente in caso che l'origin (Yahoo Finance in questo caso) sia down — fateci sapere se

ritenete queste scelte opportune rispetto a implementare questi meccanismi direttamente nei servizi.

Infine implementeremo Port Mapping e NAT come necessario per esporre i servizi alla rete Internet e raggiungere questa dalla rete virtuale creata dal Docker Compose.

Non garantiamo di consegnarlo, ma potremmo anche fornirvi una elementare client app nativa per macOS (uno di noi due potrebbe avere una certa predilezione per il linguaggio Swift...) per interfacciarvi più agevolmente con il servizio "PROXY" per testare il sistema.

Alessio Giordano - 1000069222
alessio.giordano@studium.unict.it