DISTRIBUTED SYSTEMS AND BIG DATA

Relazione Homework 2- Salvatore Pulvirenti

Il secondo homework eredita dal primo la struttura del sistema e le funzionalità (gRPC , circuit braker). In questa parte abbiamo inserito il pattern CQRS per l'interfacciamento con il DB e il protocollo KAFKA per la comunicazione fra le parti richieste di alert system e alarm notifier.

Descrizione

Lo schema del precedente homework è riportato nella figura 1

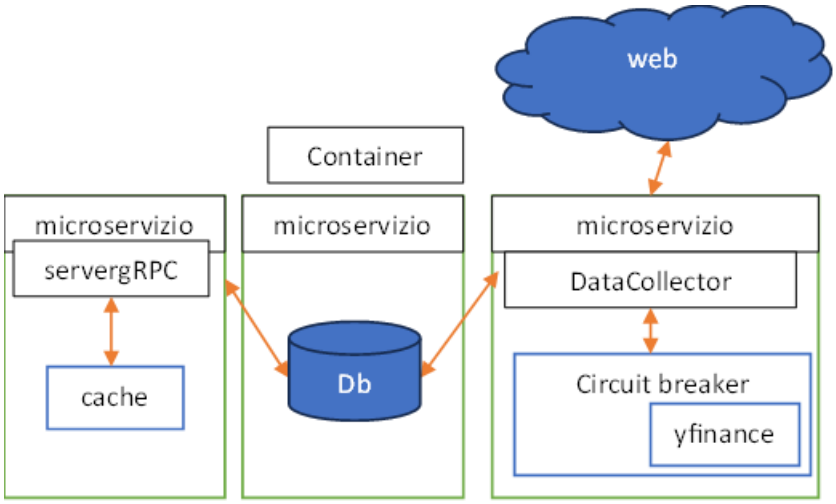

Figura 1

Nell' homework oggetto di questa relazione è cambiata come in figura 2

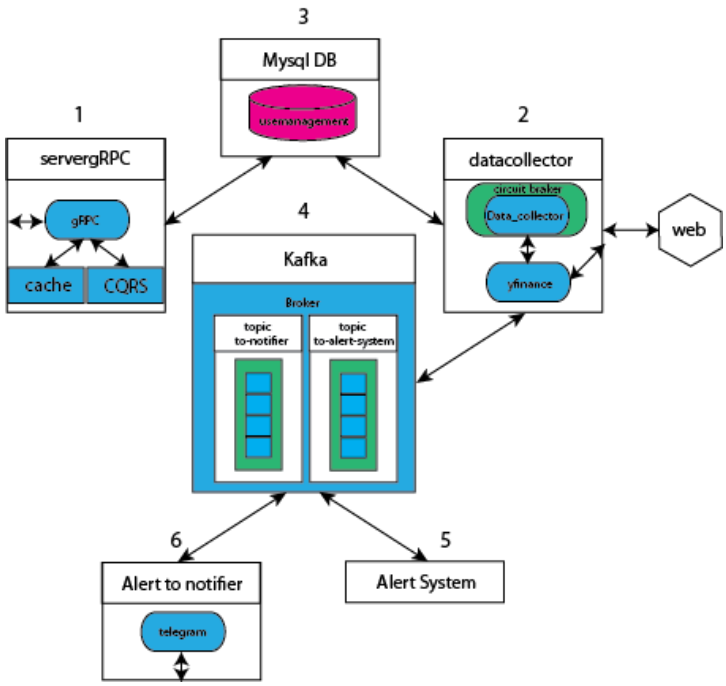

Figura 2

La directory di lavoro è così strutturata figura3:

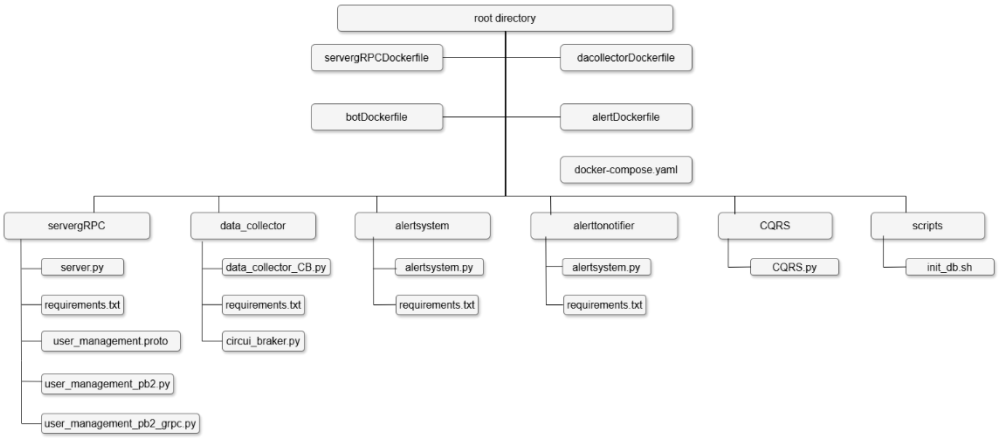

Figura 3

I file dockers ono stati inseriti nella directory principale perché si voleva condividere fra le varie parti del progetto il pattern CQRS per l'accesso al database che per motivi organizzativi sta in una sua directory. I riferimenti nell'esecuzione dei file docker è sempre il punto di avvio e poi si va in cascata a scendere nelle diretory e non è possibile accedere a directory che stanno nell'ordine più alto. Nel homework si lasciava la scelta fra mail o altro. Purtroppo dal primo di gennaio al server di posta di Google non è possibile accede con username e password, per questo motivo si è optato per sviluppare un bot a cui mandare le notifiche.

Database

Il progetto dell'homework ha richiesto la modifica delle tabelle del database, iniziamo dalla tabella degli **utenti**.

Field	Type	Null	Key	Default	Extra
email	varchar(255)	NO	PRI	NULL	
ticker	varchar(10)	NO		NULL	
High_value	DOUBLE	NO		NULL	
Low_value	DOUBLE	NO		NULL	
telegramid	Bigint	YES		NULL	

Questa tabella è quella che ha subito modifiche sostanziali, avendo dovuto inserire per motivi di progetto le colonne High_value, Low_value e telegramid . I campi relativi al range per le soglie che servono per segnalare eventuali valori fuori da questi range, il campo telegramid serve per mandare le informazioni correttamente agli utenti più avanti motiveremo questa scelta.

Field	Type	Null	Key	Default	Extra
Id	Int	NO	PRI	NULL	Auto_increment
email	varchar(255)	YES	MUL	NULL	
ticker	varchar(10)	YES		NULL	
Value	Double	NO		NULL	
timestamp	Datetime	NO		NULL	

La tabella **stock_data** è rimasta inalterata.

servergRPC

Questa parte del progetto è stata adeguata alle richieste dell'homework, che chiedevano l'uso del pattern CQRS(Command Query Responsibility Segregation) per intefacciarsi con il DB. Questo pattern è stato racchiuso in un modulo che viene utilizzato dal microservizio del server per intefacciarsi con quello del DB Mysql.

datacollector

Il data_collector_CB.py si occupa di leggere il db per conoscere gli utenti e il ticker scelto e telegramid. Qui inizia la comunicarione Kafka con un produttore sincrono sequenziale.

alertsystem

In questa parte di codice avviene l'elaborazione dei dati, così come da richiesta e se il valore del ticker è fuori sognlia viene generato un messaggio verso un altro consumer. La lettura dei dati dalla piattaforma di streaming kafka avviene in modalità sincrona mettendo il parametro enable.auto.commit a false.

alertonotifier

Come già accennato dal primo gennaio l'accesso al server di posta da parte di app di terze parti sembra che debba avvenire con autenticazione a due fasi. Per questo motivo si è deciso di optare per testare il sistema, di mandare la comunicazione di fuori range attraverso un bot telegram. Per fare questo dopo avere creato da "**botfather"** il numero che identifica unicamente un bot che semplicemente dopo il comando ***/start*** chiede ogni quanti secondicon il comando ***/set <secondi>*** inviare la comunicazione. Affinchè sia indirizzata correttamente la comunicazione viene letto id dell'utente e se è identico a quello inserito in fase di registrazione il messaggio viene stampato.

CQRS

Affinchè le modifiche fatte al file in cui il pattern cqrs è scitto avesse un punto di accesso unico, è stato inserito in una directory. Questo file è importato sia da servergPRC/server.py che sa datacollercto(data_collector_CB.py.

KAFKA

Due parole sulla piattaforma di streaming kafka. Si è provato ad avviare la comunicazione attraverso un broker con due topic, come da richiesta per passare a due. Nel file di configurazione dei container docker-compose.yaml