**Progetto dsbd 2020/21**

Nicola Pollino 1000008865

Paolo Zagarella 1000008880

Elaborato numero 8: Logging System

Variante A: Sviluppo con Java Spring MVC, JPA e MySql

|  |  |
| --- | --- |
| **Traccia:**  **Scrivere un micro-servizio sottoscrittore kafka del topic logging Alla ricezione di ognuno dei possibili messaggi, salva sul db il loro contenuto.**  **Trovare un modello dati comune per i messaggi.**  **Nota: se non e' disponibile un timestamp aggiungerlo con quello attuale** | |
| get i log con key = {key} che abbiano tempo compreso fra due forniti nei parametri della GET (obbligatori) | GET /keys/{key}from=  unixTimestampStart&end=unixTimestampEnd |
| get tutti i log message di tipo http\_errors associati al service {service} | GET /http\_errors/services/{service}from=  unixTimestampStart&end=unixTimestampEnd |
| get info sull'availability del service {service}. cioè tutti i messaggi service\_down associati al service {service},li considera validi per 30 secondi, li conta, raggruppati per status e ritorna un json con le availability | GET /uptime/services/{service}from=  unixTimeStampStart&end=unixTimestampEnd |

****

Docker

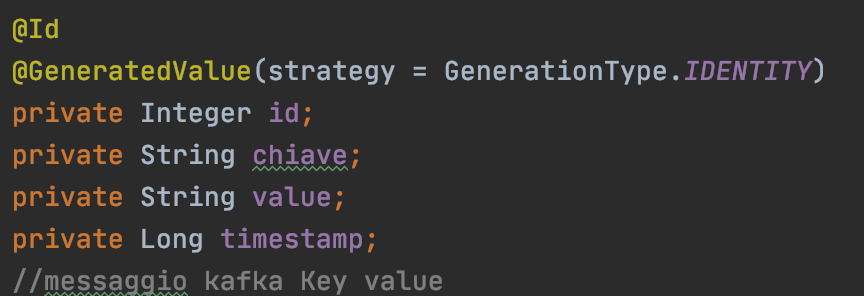
Sono stati creati i container come visibile da l’immagine qui di fianco, loggingdocker, mysqldb, phpmyadmin, fallback ed infine zoo, kafka e il container fakeproducer per inviare tramite script in python dei fake log al topic logging.

Entita’

L’entità è la stessa per ogni messaggio.

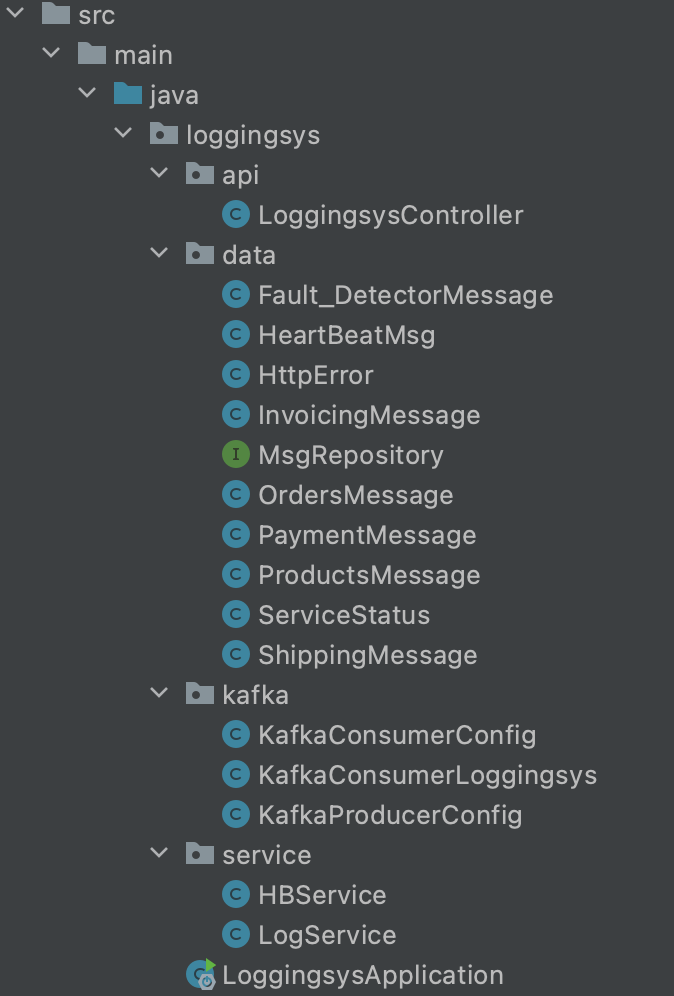
E’ stato aggiunto, oltre ad un id automatico tramite IDENTITY, un campo timestamp che è lo stesso contenuto nel value.

Se invece value è sprovvisto di timestamp viene dunque memorizzato

tramite System.currentTimeMillis().

# **Struttura del progetto**

All’interno della directory del microservizio loggingsys troviamo:

* Il service LogService che implementa la business logic e contiene i metodi che vengono poi invocati dal controller LoggingsysController.
* Il service HBService il quale viene sfruttato sia per la gestione degli errori HTTP del tipo 40x e 50x, sia per l’healt-check.
* I diversi modelli di messaggi che possono essere consumati da loggingsys, il messaggio dell’heartbeater check e il repository che estende le CRUD Repository li troviamo in data
* Sempre in data troviamo il modello di messaggio per la l’availibility del servizio al momento dell’invocazione della terza get.
* infine nella cartella kafka troviamo le configurazioni di Consumer e Producer Kafka; mentre nel file KafkaConsumerLoggingsys abbiamo la funzione che rimane in ascolto e sceglie il messaggio opportuno dalla cartella data, analizzando la chiave del messaggio ricevuto.

Log service:

HBService:

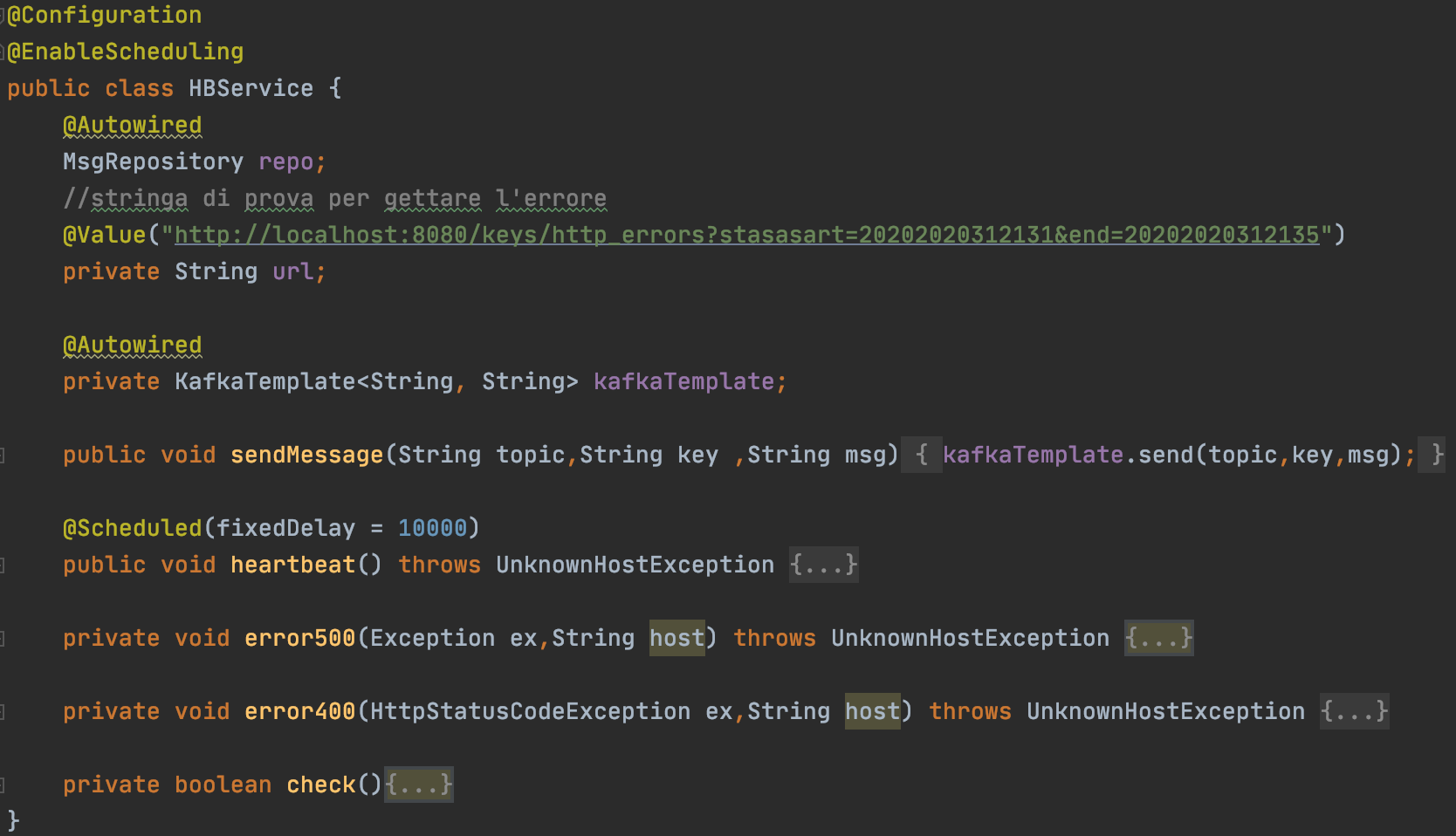
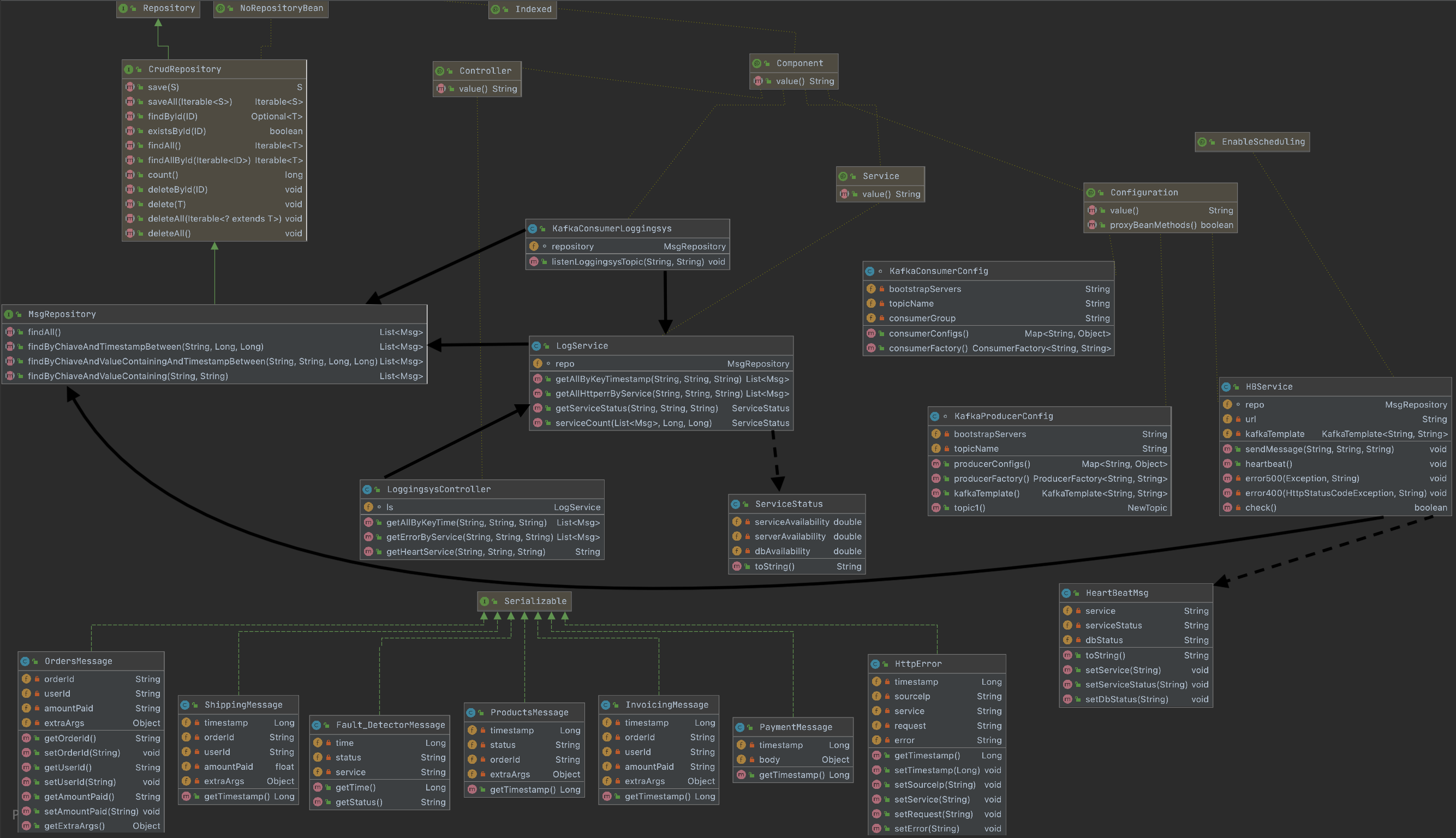


Diagramma delle classi

ogni freccia piena delinea relazioni di tipo 1 a 1