Nicoletta Carrubba, 578287

Università di Pisa-Dipartimento di Informatica
Corso di Laurea in Informatica
Primo progetto intermedio di Programmazione II
a.a. 2020-21

Introduzione

Il progetto di programmazione II ha come obbiettivo la realizzazione di un componente software di supporto alla gestione e l'analisi di una rete sociale, per far ciò è stato implementato quanto segue.

La classe Post

La classe post viene realizzata come implementazione dell'interfaccia "Post_interface" per rappresentare un generico post pubblicato su una rete sociale, in particolare per la rappresentazione sono state utilizzate le seguenti variabili d'istanza:

- Id, è un intero che identifica univocamente un'istanza della classe Post. L'id di ogni post viene inizializzato tramite un metodo all'interno della classe "SocialNetwork" che garantisce univocità. Viene contrassegnata come final in quanto non modificabile.
- Author, è una stringa che identifica il nome utente dell'autore del Post. Viene contrassegnata come final in quanto non modificabile.
- Text, è una stringa di lunghezza compresa tra 1 e 140 che identifica il testo del Post. Viene contrassegnata come final in quanto non modificabile.
- Timestamp, è una variabile di classe Timestamp, identifica la data e l'ora della creazione del Post. Viene contrassegnata come final in quanto non modificabile.
- Likes, è una LinkedList di stringhe che contiene i nomi degli utenti che hanno messo like al post.

All'interno della classe post sono stati implementate le funzionalità richieste dalla consegna tramite la creazioni di diversi metodi tra cui i metodi di tipo get: getId, getAuthor, getText, getTimeStamp e getLikes che restituiscono gli elementi specificati dal nome del metodo. Inoltre è stato implementato il metodo addLikes che permette l'inserimento di un like all'istanza Post aggiungendo l'utente che ha inserito il like all'interno della lista Likes. All'interno di quest'ultimo metodo sono stati aggiunti dei controlli per verificare che l'utente che vuole inserire il like non sia l'autore del post e non abbia già messo mi piace, però non vengono lanciate eccezioni in quanto l'azione non viene ritenuta un errore e quindi viene ignorata.

La classe SocialNetwork

La classe SocialNetwork viene realizzata come implementazione dell'interfaccia "SocialNetwork_interface" per rappresentare una rete sociale.

Le variabili di istanza utilizzate sono:

- UsersPosts, è una struttura dati di tipo Map<String,Set<Post>> che identifica l'insieme degli utenti iscritti all'interno della rete sociale con l'insieme dei loro post. Viene inizializzata con

un nuovo utente al momento della sua iscrizione alla rete sociale, implicando che gli utenti possono non avere un post ed essere comunque parte della rete.

- UsersFollowing, è una struttura dati di tipo Map<String,Set<String>> che identifica l'insieme degli utenti iscritti all'interno della rete sociale con l'insieme degli utenti da loro seguiti. Viene inizializzata con un nuovo utente al momento della sua iscrizione alla rete sociale e viene aggiornata quando l'utente mette un like ad un altro utente inserendo quest'ultimo all'interno della struttura.
- Posts, è una struttura dati di tipo Map<Integer, Post> che identifica una tabella che associa l'id ad un post.
- IDCount, è un intero utilizzato per dare Id univoci ai post che vengono creati.
- Soglia, è un intero utilizzato all'interno del metodo influencers. Inizialmente è uguale ad 1 e all'aumentare del numero di utenti registrati nella rete la soglia aumenta chiamando il metodo Scalasoglia(numeroutenti) .

La funzione di astrazione della classe è definita come $\alpha(c) = \{c.posts.get(i) \mid 0 <= i < c.posts.size()\}$, questo poiché l'insieme dei post contiene già tutte le informazioni necessarie ai metodi presenti nella classe e le altre strutture dati che sono state implementate sono di supporto per rendere la gestione della classe ottimizzata.

La classe contiene vari metodi, in particolare si riportano le implementazioni dei metodi più rilevanti:

GuessFollowers(List<Post> ps), restituisce i follower degli utenti autori dei post passati per input, dove per followers di un utente si intendono gli utenti che hanno messo like ai suoi post.Il metodo è dichiarato static in quanto deriva tutte le informazioni richieste dall'input.

Influencers(Map<String,Set<String>> followers) restituisce gli utenti che possono essere definiti "influencers". Per essere definiti come influencers, si deve avere un numero di followers maggiore del numero di following e un numero di followers maggiore della variabile di istanza "soglia".

createUser(String username) permette la registrazione di un nuovo utente, in particolare, aggiorna le due variabili di istanza "UsersPosts" e "UsersFollowing" aggiungendo il nuovo utente. All'interno del metodo inoltre è stato implementato un controllo per evitare utenti duplicati.

CreateNewPost(String author, String text) permette la creazione di un nuovo post aggiornando le due variabili di istanza "Posts" e "UserPosts" e chiamando il costruttore della classe Post.

Like(int id,String username), viene implementato per aggiungere dei controlli riguardati la rete sociale al metodo addLikes della classe post, in particolare vengono controllate l'esistenza del post con id passato da input e che l'utente che prova a mettere like sia stato registrato nella rete sociale. Inoltre aggiorna la variabile di istanza "UsersFollowing" con le relazioni di following risultanti dall'azione like.

ScalaSoglia(int numeroutenti) permette di scalare la variabile di istanza soglia passando il numero di utenti. All'interno del metodo si ha un controllo per verificare che il numero di utenti sia maggiore di 7, questo perché la soglia deve essere sempre maggiore o uguale ad 1, quindi applicando l'espressione con cui viene calcolata la soglia, cioè ((numeroutenti/2)-3), su un numero maggiore di 7 si rispetta la condizione.

GetUnId(), è un metodo private che restituisce un intero unico, utilizzato per garantire l'unicità degli id dei Post nella rete.

Altri metodi che sono stati implementati sono i metodi getters GetAllPosts, GetFollowing, GetSoglia che restituiscono gli elementi specificati dal nome del metodo e gli altri metodi richiesti dalla consegna del progetto.

La classe CheckSocialNetwork

Questa classe viene proposta come un'estensione gerarchica del tipo di dato SocialNetwork che permette la segnalazione di contenuti offensivi presenti nella rete sociale, infatti si introduce una variabile di istanza ReportedPosts che ha al suo interno un associazione tra i post e gli utenti che hanno segnalato quel post. La segnalazione avviene tramite il metodo "ReportPost(String user, Post ps)" che aggiorna la variabile di istanza con il post e l'user che lo ha segnalato.
Inoltre è stato introdotto un metodo RemoveReportedPost(Post ps) per rimuovere il post dai post segnalati.

MyExceptions

Sono state introdotte tre nuove eccezioni checked che estendono la classe Exception per catturare situazioni anomale all'interno del programma, in particolare:

- DuplicateUserException viene lanciata quando si prova a registrare due volte lo stesso utente.
- NoPostException viene lanciata quando si provano ad usare i metodi della classe SocialNetwork su post non registrati sulla rete sociale.
- NoUserFoundException viene lanciata quando si provano ad usare i metodi della classe SocialNetwork con utenti che non sono stati registrati sulla rete sociale.

TestSet

TestSet è la classe contenente il main, viene utilizzata per poter provare i metodi implementati nelle classi illustrate precedentemente.
Si crea una rete sociale con il metodo creaRete() e su di essa vengono proposti due test, il primo che simula l'utilizzo di tutti i metodi all'interno della classe SocialNetwork e il secondo che simula delle situazioni anomale che hanno bisogno di essere gestite dalle eccezioni.
Infine si crea una nuova rete sociale per simulare, tramite il terzo test, i metodi all'interno della classe CheckSocialNetwork.