package com.twitter.timelineranker.util

import com.twitter.finagle.stats.StatsReceiver

import com.twitter.search.earlybird.thriftscala.ThriftSearchResult

import com.twitter.timelines.model.TweetId

import com.twitter.timelines.model.UserId

import com.twitter.timelines.visibility.model.CheckedUserActor

import com.twitter.timelines.visibility.model.HasVisibilityActors

import com.twitter.timelines.visibility.model.VisibilityCheckUser

case class SearchResultWithVisibilityActors(

searchResult: ThriftSearchResult,

statsReceiver: StatsReceiver)

extends HasVisibilityActors {

private[this] val searchResultWithoutMetadata =

statsReceiver.counter("searchResultWithoutMetadata")

val tweetId: TweetId = searchResult.id

val metadata = searchResult.metadata

val (userId, isRetweet, sourceUserId, sourceTweetId) = metadata match {

case Some(md) => {

(

md.fromUserId,

md.isRetweet,

md.isRetweet.getOrElse(false) match {

case true => Some(md.referencedTweetAuthorId)

case false => None

},

// metadata.sharedStatusId is defaulting to 0 for tweets that don't have one

// 0 is not a valid tweet id so converting to None. Also disregarding sharedStatusId

// for non-retweets.

if (md.isRetweet.isDefined && md.isRetweet.get)

md.sharedStatusId match {

case 0 => None

case id => Some(id)

}

else None

)

}

case None => {

searchResultWithoutMetadata.incr()

throw new IllegalArgumentException(

"searchResult is missing metadata: " + searchResult.toString)

}

}

/\*\*

\* Returns the set of users (or 'actors') relevant for Tweet visibility filtering. Usually the

\* Tweet author, but if this is a Retweet, then the source Tweet author is also relevant.

\*/

def getVisibilityActors(viewerIdOpt: Option[UserId]): Seq[CheckedUserActor] = {

val isSelf = isViewerAlsoTweetAuthor(viewerIdOpt, Some(userId))

Seq(

Some(CheckedUserActor(isSelf, VisibilityCheckUser.Tweeter, userId)),

sourceUserId.map {

CheckedUserActor(isSelf, VisibilityCheckUser.SourceUser, \_)

}

).flatten

}

}