package com.twitter.timelineranker.model

import com.twitter.timelineranker.{thriftscala => thrift}

import com.twitter.util.Future

object CandidateTweetsResult {

val Empty: CandidateTweetsResult = CandidateTweetsResult(Nil, Nil)

val EmptyFuture: Future[CandidateTweetsResult] = Future.value(Empty)

val EmptyCandidateTweet: Seq[CandidateTweet] = Seq.empty[CandidateTweet]

def fromThrift(response: thrift.GetCandidateTweetsResponse): CandidateTweetsResult = {

val candidates = response.candidates

.map(\_.map(CandidateTweet.fromThrift))

.getOrElse(EmptyCandidateTweet)

val sourceTweets = response.sourceTweets

.map(\_.map(CandidateTweet.fromThrift))

.getOrElse(EmptyCandidateTweet)

if (sourceTweets.nonEmpty) {

require(candidates.nonEmpty, "sourceTweets cannot have a value if candidates list is empty.")

}

CandidateTweetsResult(candidates, sourceTweets)

}

}

case class CandidateTweetsResult(

candidates: Seq[CandidateTweet],

sourceTweets: Seq[CandidateTweet]) {

def toThrift: thrift.GetCandidateTweetsResponse = {

val thriftCandidates = candidates.map(\_.toThrift)

val thriftSourceTweets = sourceTweets.map(\_.toThrift)

thrift.GetCandidateTweetsResponse(

candidates = Some(thriftCandidates),

sourceTweets = Some(thriftSourceTweets)

)

}

}