package com.twitter.timelineranker.recap\_author

import com.twitter.finagle.stats.StatsReceiver

import com.twitter.servo.util.FutureArrow

import com.twitter.timelineranker.core.CandidateEnvelope

import com.twitter.timelineranker.model.RecapQuery.DependencyProvider

import com.twitter.timelineranker.model.TweetIdRange

import com.twitter.timelines.clients.relevance\_search.SearchClient

import com.twitter.timelines.clients.relevance\_search.SearchClient.TweetTypes

import com.twitter.timelines.model.UserId

import com.twitter.util.Future

/\*\*

\* Fetch recap results based on an author id set passed into the query.

\* Calls into the same search method as Recap, but uses the authorIds instead of the SGS-provided followedIds.

\*/

class RecapAuthorSearchResultsTransform(

searchClient: SearchClient,

maxCountProvider: DependencyProvider[Int],

relevanceOptionsMaxHitsToProcessProvider: DependencyProvider[Int],

enableSettingTweetTypesWithTweetKindOptionProvider: DependencyProvider[Boolean],

statsReceiver: StatsReceiver,

logSearchDebugInfo: Boolean = false)

extends FutureArrow[CandidateEnvelope, CandidateEnvelope] {

private[this] val maxCountStat = statsReceiver.stat("maxCount")

private[this] val numInputAuthorsStat = statsReceiver.stat("numInputAuthors")

private[this] val excludedTweetIdsStat = statsReceiver.stat("excludedTweetIds")

override def apply(envelope: CandidateEnvelope): Future[CandidateEnvelope] = {

val maxCount = maxCountProvider(envelope.query)

maxCountStat.add(maxCount)

val authorIds = envelope.query.authorIds.getOrElse(Seq.empty[UserId])

numInputAuthorsStat.add(authorIds.size)

val excludedTweetIdsOpt = envelope.query.excludedTweetIds

excludedTweetIdsOpt.map { excludedTweetIds => excludedTweetIdsStat.add(excludedTweetIds.size) }

val tweetIdRange = envelope.query.range

.map(TweetIdRange.fromTimelineRange)

.getOrElse(TweetIdRange.default)

val beforeTweetIdExclusive = tweetIdRange.toId

val afterTweetIdExclusive = tweetIdRange.fromId

val relevanceOptionsMaxHitsToProcess = relevanceOptionsMaxHitsToProcessProvider(envelope.query)

searchClient

.getUsersTweetsForRecap(

userId = envelope.query.userId,

followedUserIds = authorIds.toSet, // user authorIds as the set of followed users

retweetsMutedUserIds = Set.empty,

maxCount = maxCount,

tweetTypes = TweetTypes.fromTweetKindOption(envelope.query.options),

searchOperator = envelope.query.searchOperator,

beforeTweetIdExclusive = beforeTweetIdExclusive,

afterTweetIdExclusive = afterTweetIdExclusive,

enableSettingTweetTypesWithTweetKindOption =

enableSettingTweetTypesWithTweetKindOptionProvider(envelope.query),

excludedTweetIds = excludedTweetIdsOpt,

earlybirdOptions = envelope.query.earlybirdOptions,

getOnlyProtectedTweets = false,

logSearchDebugInfo = logSearchDebugInfo,

returnAllResults = true,

enableExcludeSourceTweetIdsQuery = false,

relevanceOptionsMaxHitsToProcess = relevanceOptionsMaxHitsToProcess

).map { results => envelope.copy(searchResults = results) }

}

}