package com.twitter.home\_mixer.product.list\_tweets

import com.twitter.home\_mixer.candidate\_pipeline.TimelineServiceResponseFeatureTransformer

import com.twitter.home\_mixer.marshaller.timelines.TimelineServiceCursorMarshaller

import com.twitter.home\_mixer.product.list\_tweets.model.ListTweetsQuery

import com.twitter.home\_mixer.product.list\_tweets.param.ListTweetsParam.ServerMaxResultsParam

import com.twitter.home\_mixer.service.HomeMixerAlertConfig

import com.twitter.product\_mixer.component\_library.candidate\_source.timeline\_service.TimelineServiceTweetCandidateSource

import com.twitter.product\_mixer.component\_library.model.candidate.TweetCandidate

import com.twitter.product\_mixer.core.functional\_component.candidate\_source.BaseCandidateSource

import com.twitter.product\_mixer.core.functional\_component.transformer.CandidateFeatureTransformer

import com.twitter.product\_mixer.core.functional\_component.transformer.CandidatePipelineQueryTransformer

import com.twitter.product\_mixer.core.functional\_component.transformer.CandidatePipelineResultsTransformer

import com.twitter.product\_mixer.core.model.common.identifier.CandidatePipelineIdentifier

import com.twitter.product\_mixer.core.pipeline.candidate.CandidatePipelineConfig

import com.twitter.timelineservice.{thriftscala => t}

import javax.inject.Inject

import javax.inject.Singleton

@Singleton

class ListTweetsTimelineServiceCandidatePipelineConfig @Inject() (

timelineServiceTweetCandidateSource: TimelineServiceTweetCandidateSource)

extends CandidatePipelineConfig[ListTweetsQuery, t.TimelineQuery, t.Tweet, TweetCandidate] {

override val identifier: CandidatePipelineIdentifier =

CandidatePipelineIdentifier("ListTweetsTimelineServiceTweets")

override val queryTransformer: CandidatePipelineQueryTransformer[

ListTweetsQuery,

t.TimelineQuery

] = { query =>

val timelineQueryOptions = t.TimelineQueryOptions(

contextualUserId = query.clientContext.userId,

)

t.TimelineQuery(

timelineType = t.TimelineType.List,

timelineId = query.listId,

maxCount = query.maxResults(ServerMaxResultsParam).toShort,

cursor2 = query.pipelineCursor.flatMap(TimelineServiceCursorMarshaller(\_)),

options = Some(timelineQueryOptions),

timelineId2 = Some(t.TimelineId(t.TimelineType.List, query.listId, None))

)

}

override def candidateSource: BaseCandidateSource[t.TimelineQuery, t.Tweet] =

timelineServiceTweetCandidateSource

override val resultTransformer: CandidatePipelineResultsTransformer[t.Tweet, TweetCandidate] = {

sourceResult => TweetCandidate(id = sourceResult.statusId)

}

override val featuresFromCandidateSourceTransformers: Seq[CandidateFeatureTransformer[t.Tweet]] =

Seq(TimelineServiceResponseFeatureTransformer)

override val alerts = Seq(

HomeMixerAlertConfig.BusinessHours.defaultSuccessRateAlert(99.7)

)

}