package com.twitter.home\_mixer.product.scored\_tweets.feature\_hydrator

import com.twitter.home\_mixer.model.HomeFeatures.\_

import com.twitter.home\_mixer.product.scored\_tweets.param.ScoredTweetsParam.CachedScoredTweets

import com.twitter.home\_mixer.{thriftscala => hmt}

import com.twitter.product\_mixer.core.feature.Feature

import com.twitter.product\_mixer.core.feature.featuremap.FeatureMap

import com.twitter.product\_mixer.core.feature.featuremap.FeatureMapBuilder

import com.twitter.product\_mixer.core.functional\_component.feature\_hydrator.QueryFeatureHydrator

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

import com.twitter.product\_mixer.core.pipeline.PipelineQuery

import com.twitter.servo.cache.TtlCache

import com.twitter.stitch.Stitch

import com.twitter.util.Return

import com.twitter.util.Throw

import com.twitter.util.Time

import javax.inject.Inject

import javax.inject.Singleton

/\*\*

\* Fetch scored Tweets from cache and exclude the expired ones

\*/

@Singleton

case class CachedScoredTweetsQueryFeatureHydrator @Inject() (

scoredTweetsCache: TtlCache[Long, hmt.ScoredTweetsResponse])

extends QueryFeatureHydrator[PipelineQuery] {

override val identifier: FeatureHydratorIdentifier =

FeatureHydratorIdentifier("CachedScoredTweets")

override val features: Set[Feature[\_, \_]] = Set(CachedScoredTweetsFeature)

override def hydrate(query: PipelineQuery): Stitch[FeatureMap] = {

val userId = query.getRequiredUserId

val tweetScoreTtl = query.params(CachedScoredTweets.TTLParam)

Stitch.callFuture(scoredTweetsCache.get(Seq(userId))).map { keyValueResult =>

keyValueResult(userId) match {

case Return(cachedCandidatesOpt) =>

val cachedScoredTweets = cachedCandidatesOpt.map(\_.scoredTweets).getOrElse(Seq.empty)

val nonExpiredTweets = cachedScoredTweets.filter { tweet =>

tweet.lastScoredTimestampMs.exists(Time.fromMilliseconds(\_).untilNow < tweetScoreTtl)

}

FeatureMapBuilder().add(CachedScoredTweetsFeature, nonExpiredTweets).build()

case Throw(exception) => throw exception

}

}

}

}