package com.twitter.unified\_user\_actions.adapter.favorite\_archival\_events

import com.twitter.finagle.stats.NullStatsReceiver

import com.twitter.finagle.stats.StatsReceiver

import com.twitter.finatra.kafka.serde.UnKeyed

import com.twitter.timelineservice.fanout.thriftscala.FavoriteArchivalEvent

import com.twitter.unified\_user\_actions.adapter.AbstractAdapter

import com.twitter.unified\_user\_actions.adapter.common.AdapterUtils

import com.twitter.unified\_user\_actions.thriftscala.\_

class FavoriteArchivalEventsAdapter

extends AbstractAdapter[FavoriteArchivalEvent, UnKeyed, UnifiedUserAction] {

import FavoriteArchivalEventsAdapter.\_

override def adaptOneToKeyedMany(

input: FavoriteArchivalEvent,

statsReceiver: StatsReceiver = NullStatsReceiver

): Seq[(UnKeyed, UnifiedUserAction)] =

adaptEvent(input).map { e => (UnKeyed, e) }

}

object FavoriteArchivalEventsAdapter {

def adaptEvent(e: FavoriteArchivalEvent): Seq[UnifiedUserAction] =

Option(e).map { e =>

UnifiedUserAction(

userIdentifier = UserIdentifier(userId = Some(e.favoriterId)),

item = getItem(e),

actionType =

if (e.isArchivingAction.getOrElse(true)) ActionType.ServerTweetArchiveFavorite

else ActionType.ServerTweetUnarchiveFavorite,

eventMetadata = getEventMetadata(e)

)

}.toSeq

def getItem(e: FavoriteArchivalEvent): Item =

Item.TweetInfo(

TweetInfo(

// Please note that here we always use TweetId (not sourceTweetId)!!!

actionTweetId = e.tweetId,

actionTweetAuthorInfo = Some(AuthorInfo(authorId = e.tweetUserId)),

retweetedTweetId = e.sourceTweetId

)

)

def getEventMetadata(e: FavoriteArchivalEvent): EventMetadata =

EventMetadata(

sourceTimestampMs = e.timestampMs,

receivedTimestampMs = AdapterUtils.currentTimestampMs,

sourceLineage = SourceLineage.ServerFavoriteArchivalEvents,

)

}