package com.twitter.usersignalservice.signals

package common

import com.twitter.simclusters\_v2.thriftscala.InternalId

import com.twitter.socialgraph.thriftscala.EdgesRequest

import com.twitter.socialgraph.thriftscala.EdgesResult

import com.twitter.socialgraph.thriftscala.PageRequest

import com.twitter.socialgraph.thriftscala.RelationshipType

import com.twitter.socialgraph.thriftscala.SocialGraphService

import com.twitter.socialgraph.thriftscala.SrcRelationship

import com.twitter.twistly.common.UserId

import com.twitter.usersignalservice.thriftscala.Signal

import com.twitter.usersignalservice.thriftscala.SignalType

import com.twitter.util.Duration

import com.twitter.util.Future

import com.twitter.util.Time

object SGSUtils {

val MaxNumSocialGraphSignals = 200

val MaxAge: Duration = Duration.fromDays(90)

def getSGSRawSignals(

userId: UserId,

sgsClient: SocialGraphService.MethodPerEndpoint,

relationshipType: RelationshipType,

signalType: SignalType,

): Future[Option[Seq[Signal]]] = {

val edgeRequest = EdgesRequest(

relationship = SrcRelationship(userId, relationshipType),

pageRequest = Some(PageRequest(count = None))

)

val now = Time.now.inMilliseconds

sgsClient

.edges(Seq(edgeRequest))

.map { sgsEdges =>

sgsEdges.flatMap {

case EdgesResult(edges, \_, \_) =>

edges.collect {

case edge if edge.createdAt >= now - MaxAge.inMilliseconds =>

Signal(

signalType,

timestamp = edge.createdAt,

targetInternalId = Some(InternalId.UserId(edge.target)))

}

}

}

.map { signals =>

signals

.take(MaxNumSocialGraphSignals)

.groupBy(\_.targetInternalId)

.mapValues(\_.maxBy(\_.timestamp))

.values

.toSeq

.sortBy(-\_.timestamp)

}

.map(Some(\_))

}

}