package com.twitter.frigate.pushservice.take.history

import com.twitter.finagle.stats.StatsReceiver

import com.twitter.frigate.common.history.HistoryStoreKeyContext

import com.twitter.frigate.common.history.PushServiceHistoryStore

import com.twitter.frigate.common.rec\_types.RecTypes

import com.twitter.frigate.pushservice.model.PushTypes.PushCandidate

import com.twitter.frigate.pushservice.params.PushFeatureSwitchParams

import com.twitter.frigate.thriftscala.FrigateNotification

import com.twitter.util.Duration

import com.twitter.util.Future

import com.twitter.conversions.DurationOps.\_

class HistoryWriter(historyStore: PushServiceHistoryStore, stats: StatsReceiver) {

private lazy val historyWriterStats = stats.scope(this.getClass.getSimpleName)

private lazy val historyWriteCounter = historyWriterStats.counter("history\_write\_num")

private lazy val loggedOutHistoryWriteCounter =

historyWriterStats.counter("logged\_out\_history\_write\_num")

private def writeTtlForHistory(candidate: PushCandidate): Duration = {

if (candidate.target.isLoggedOutUser) {

60.days

} else if (RecTypes.isTweetType(candidate.commonRecType)) {

candidate.target.params(PushFeatureSwitchParams.FrigateHistoryTweetNotificationWriteTtl)

} else candidate.target.params(PushFeatureSwitchParams.FrigateHistoryOtherNotificationWriteTtl)

}

def writeSendToHistory(

candidate: PushCandidate,

frigateNotificationForPersistence: FrigateNotification

): Future[Unit] = {

val historyStoreKeyContext = HistoryStoreKeyContext(

candidate.target.targetId,

candidate.target.pushContext.flatMap(\_.useMemcacheForHistory).getOrElse(false)

)

if (candidate.target.isLoggedOutUser) {

loggedOutHistoryWriteCounter.incr()

} else {

historyWriteCounter.incr()

}

historyStore

.put(

historyStoreKeyContext,

candidate.createdAt,

frigateNotificationForPersistence,

Some(writeTtlForHistory(candidate))

)

}

}