package com.twitter.frigate.pushservice.predicate

import com.twitter.finagle.stats.StatsReceiver

import com.twitter.frigate.common.base.TargetUser

import com.twitter.frigate.common.candidate.FrigateHistory

import com.twitter.frigate.common.history.History

import com.twitter.frigate.common.predicate.FrigateHistoryFatiguePredicate

import com.twitter.frigate.common.predicate.{FatiguePredicate => TargetFatiguePredicate}

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

import com.twitter.hermit.predicate.Predicate

import com.twitter.timelines.configapi.Param

import com.twitter.util.Duration

object DiscoverTwitterPredicate {

/\*\*

\* Predicate used to determine if a minimum duration has elapsed since the last MR push

\* for a CRT to be valid.

\* @param name Identifier of the caller (used for stats)

\* @param intervalParam The minimum duration interval

\* @param stats StatsReceiver

\* @return Target Predicate

\*/

def minDurationElapsedSinceLastMrPushPredicate(

name: String,

intervalParam: Param[Duration],

stats: StatsReceiver

): Predicate[Target] =

Predicate

.fromAsync { target: Target =>

val interval =

target.params(intervalParam)

FrigateHistoryFatiguePredicate(

minInterval = interval,

getSortedHistory = { h: History =>

val magicRecsOnlyHistory =

TargetFatiguePredicate.magicRecsPushOnlyFilter(h.sortedPushDmHistory)

TargetFatiguePredicate.magicRecsNewUserPlaybookPushFilter(magicRecsOnlyHistory)

}

).flatContraMap { target: TargetUser with FrigateHistory =>

target.history

}.apply(Seq(target)).map {

\_.head

}

}.withStats(stats.scope(s"${name}\_predicate\_mr\_push\_min\_interval"))

.withName(s"${name}\_predicate\_mr\_push\_min\_interval")

}