package com.twitter.frigate.pushservice.refresh\_handler

import com.twitter.finagle.stats.Stat

import com.twitter.finagle.stats.StatsReceiver

import com.twitter.frigate.common.base.CandidateDetails

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

import com.twitter.frigate.common.candidate.TargetABDecider

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

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

import com.twitter.frigate.pushservice.target.TargetScoringDetails

class RFPHRestrictStep()(implicit stats: StatsReceiver) {

private val statsReceiver: StatsReceiver = stats.scope("RefreshForPushHandler")

private val restrictStepStats: StatsReceiver = statsReceiver.scope("restrict")

private val restrictStepNumCandidatesDroppedStat: Stat =

restrictStepStats.stat("candidates\_dropped")

/\*\*

\* Limit the number of candidates that enter the Take step

\*/

def restrict(

target: TargetUser with TargetABDecider with TargetScoringDetails,

candidates: Seq[CandidateDetails[PushCandidate]]

): (Seq[CandidateDetails[PushCandidate]], Seq[CandidateDetails[PushCandidate]]) = {

if (target.params(PushFeatureSwitchParams.EnableRestrictStep)) {

val restrictSizeParam = PushFeatureSwitchParams.RestrictStepSize

val (newCandidates, filteredCandidates) = candidates.splitAt(target.params(restrictSizeParam))

val numDropped = candidates.length - newCandidates.length

restrictStepNumCandidatesDroppedStat.add(numDropped)

(newCandidates, filteredCandidates)

} else (candidates, Seq.empty)

}

}