package com.twitter.follow\_recommendations.common.candidate\_sources.sims

import com.twitter.finagle.stats.NullStatsReceiver

import com.twitter.finagle.stats.StatsReceiver

import com.twitter.follow\_recommendations.common.models.CandidateUser

import com.twitter.follow\_recommendations.common.models.HasSimilarToContext

import com.twitter.hermit.model.Algorithm

import com.twitter.product\_mixer.core.functional\_component.candidate\_source.CandidateSource

import com.twitter.product\_mixer.core.model.common.identifier.CandidateSourceIdentifier

import com.twitter.stitch.Stitch

import com.twitter.timelines.configapi.HasParams

import javax.inject.Inject

import javax.inject.Singleton

@Singleton

class SwitchingSimsSource @Inject() (

cachedDBV2SimsStore: CachedDBV2SimsStore,

cachedDBV2SimsRefreshStore: CachedDBV2SimsRefreshStore,

cachedSimsExperimentalStore: CachedSimsExperimentalStore,

cachedSimsStore: CachedSimsStore,

statsReceiver: StatsReceiver = NullStatsReceiver)

extends CandidateSource[HasParams with HasSimilarToContext, CandidateUser] {

override val identifier: CandidateSourceIdentifier = SwitchingSimsSource.Identifier

private val stats = statsReceiver.scope("SwitchingSimsSource")

private val dbV2SimsStoreCounter = stats.counter("DBV2SimsStore")

private val dbV2SimsRefreshStoreCounter = stats.counter("DBV2SimsRefreshStore")

private val simsExperimentalStoreCounter = stats.counter("SimsExperimentalStore")

private val simsStoreCounter = stats.counter("SimsStore")

override def apply(request: HasParams with HasSimilarToContext): Stitch[Seq[CandidateUser]] = {

val selectedSimsStore =

if (request.params(SimsSourceParams.EnableDBV2SimsStore)) {

dbV2SimsStoreCounter.incr()

cachedDBV2SimsStore

} else if (request.params(SimsSourceParams.EnableDBV2SimsRefreshStore)) {

dbV2SimsRefreshStoreCounter.incr()

cachedDBV2SimsRefreshStore

} else if (request.params(SimsSourceParams.EnableExperimentalSimsStore)) {

simsExperimentalStoreCounter.incr()

cachedSimsExperimentalStore

} else {

simsStoreCounter.incr()

cachedSimsStore

}

stats.counter("total").incr()

selectedSimsStore(request)

}

}

object SwitchingSimsSource {

val Identifier: CandidateSourceIdentifier = CandidateSourceIdentifier(Algorithm.Sims.toString)

}