package com.twitter.frigate.pushservice.store

import com.twitter.finagle.stats.StatsReceiver

import com.twitter.follow\_recommendations.thriftscala.FollowRecommendationsThriftService

import com.twitter.follow\_recommendations.thriftscala.Recommendation

import com.twitter.follow\_recommendations.thriftscala.RecommendationRequest

import com.twitter.follow\_recommendations.thriftscala.RecommendationResponse

import com.twitter.follow\_recommendations.thriftscala.UserRecommendation

import com.twitter.inject.Logging

import com.twitter.storehaus.ReadableStore

import com.twitter.util.Future

case class FollowRecommendationsStore(

frsClient: FollowRecommendationsThriftService.MethodPerEndpoint,

statsReceiver: StatsReceiver)

extends ReadableStore[RecommendationRequest, RecommendationResponse]

with Logging {

private val scopedStats = statsReceiver.scope(getClass.getSimpleName)

private val requests = scopedStats.counter("requests")

private val valid = scopedStats.counter("valid")

private val invalid = scopedStats.counter("invalid")

private val numTotalResults = scopedStats.stat("total\_results")

private val numValidResults = scopedStats.stat("valid\_results")

override def get(request: RecommendationRequest): Future[Option[RecommendationResponse]] = {

requests.incr()

frsClient.getRecommendations(request).map { response =>

numTotalResults.add(response.recommendations.size)

val validRecs = response.recommendations.filter {

case Recommendation.User(\_: UserRecommendation) =>

valid.incr()

true

case \_ =>

invalid.incr()

false

}

numValidResults.add(validRecs.size)

Some(

RecommendationResponse(

recommendations = validRecs

))

}

}

}