package com.twitter.search.earlybird\_root.caching;

import java.util.Map;

import java.util.concurrent.ConcurrentHashMap;

import com.twitter.search.common.caching.filter.PerClientCacheStats;

import com.twitter.search.common.metrics.SearchRateCounter;

import com.twitter.search.earlybird.common.EarlybirdRequestUtil;

import com.twitter.search.earlybird\_root.common.EarlybirdRequestContext;

public class EarlybirdRequestPerClientCacheStats

extends PerClientCacheStats<EarlybirdRequestContext> {

private String cacheOffByClientStatFormat;

private final Map<String, SearchRateCounter> cacheTurnedOffByClient;

private String cacheHitsByClientStatFormat;

private final Map<String, SearchRateCounter> cacheHitsByClient;

public EarlybirdRequestPerClientCacheStats(String cacheRequestType) {

this.cacheOffByClientStatFormat =

cacheRequestType + "\_client\_id\_%s\_cache\_turned\_off\_in\_request";

this.cacheTurnedOffByClient = new ConcurrentHashMap<>();

this.cacheHitsByClientStatFormat = cacheRequestType + "\_client\_id\_%s\_cache\_hit\_total";

this.cacheHitsByClient = new ConcurrentHashMap<>();

}

@Override

public void recordRequest(EarlybirdRequestContext requestContext) {

if (!EarlybirdRequestUtil.isCachingAllowed(requestContext.getRequest())) {

String client = requestContext.getRequest().getClientId();

SearchRateCounter counter = cacheTurnedOffByClient.computeIfAbsent(client,

cl -> SearchRateCounter.export(String.format(cacheOffByClientStatFormat, cl)));

counter.increment();

}

}

@Override

public void recordCacheHit(EarlybirdRequestContext requestContext) {

String client = requestContext.getRequest().getClientId();

SearchRateCounter counter = cacheHitsByClient.computeIfAbsent(client,

cl -> SearchRateCounter.export(String.format(cacheHitsByClientStatFormat, cl)));

counter.increment();

}

}