package com.twitter.search.earlybird\_root;

import java.util.concurrent.ConcurrentHashMap;

import com.twitter.common.util.Clock;

import com.twitter.finagle.Service;

import com.twitter.finagle.SimpleFilter;

import com.twitter.search.common.clientstats.RequestCounters;

import com.twitter.search.common.clientstats.RequestCountersEventListener;

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

import com.twitter.search.earlybird.thrift.EarlybirdRequest;

import com.twitter.search.earlybird.thrift.EarlybirdResponse;

import com.twitter.search.earlybird\_root.filters.EarlybirdSuccessfulResponseHandler;

import com.twitter.util.Future;

public class ClientLatencyFilter extends SimpleFilter<EarlybirdRequest, EarlybirdResponse> {

// \_client\_latency\_stats\_for\_ is intended to measure the latency of requests to services that this

// root depends on. This can be used to measure how long a request takes in transit between when

// it leaves a root and when a root receives the response, in case this latency is significantly

// different than Earlybird measured latency. We break it down by client, so that we can tell

// which customers are being hit by this latency.

private static final String STAT\_FORMAT = "%s\_client\_latency\_stats\_for\_%s";

private final ConcurrentHashMap<String, RequestCounters> requestCounterForClient =

new ConcurrentHashMap<>();

private final String prefix;

public ClientLatencyFilter(String prefix) {

this.prefix = prefix;

}

@Override

public Future<EarlybirdResponse> apply(EarlybirdRequest request,

Service<EarlybirdRequest, EarlybirdResponse> service) {

RequestCounters requestCounters = requestCounterForClient.computeIfAbsent(

ClientIdUtil.getClientIdFromRequest(request), client ->

new RequestCounters(String.format(STAT\_FORMAT, prefix, client)));

RequestCountersEventListener<EarlybirdResponse> requestCountersEventListener =

new RequestCountersEventListener<>(requestCounters, Clock.SYSTEM\_CLOCK,

EarlybirdSuccessfulResponseHandler.INSTANCE);

return service.apply(request).addEventListener(requestCountersEventListener);

}

}