package com.twitter.search.earlybird\_root;

import org.apache.thrift.TException;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

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

import com.twitter.search.common.root.ValidationBehavior;

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

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

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

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

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

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

public class EarlybirdServiceValidationBehavior

extends ValidationBehavior.DefaultValidationBehavior<EarlybirdRequest, EarlybirdResponse> {

private static final Logger LOG =

LoggerFactory.getLogger(EarlybirdServiceValidationBehavior.class);

private static final EarlybirdDebugInfo EARLYBIRD\_DEBUG\_INFO =

new EarlybirdDebugInfo().setHost("earlybird\_root");

private static final SearchCounter INVALID\_SUCCESS\_RESPONSE\_THRESHOLD\_TOO\_LOW =

SearchCounter.export("invalid\_success\_response\_threshold\_too\_low");

private static final SearchCounter INVALID\_SUCCESS\_RESPONSE\_THRESHOLD\_TOO\_HIGH =

SearchCounter.export("invalid\_success\_response\_threshold\_too\_high");

protected EarlybirdResponse createErrorResponse(String errorMsg) {

EarlybirdResponse response = new EarlybirdResponse(EarlybirdResponseCode.CLIENT\_ERROR, 0);

// We're changing some ERROR logs to WARN on our side, so we want to ensure

// that the response contains the debug information the client needs to

// resolve the problem.

response.setDebugInfo(EARLYBIRD\_DEBUG\_INFO);

response.setDebugString(errorMsg);

return response;

}

@Override

public EarlybirdResponse getResponseIfInvalidRequest(EarlybirdRequest request) {

// First, fix up the query.

EarlybirdRequestUtil.checkAndSetCollectorParams(request);

EarlybirdRequestUtil.logAndFixExcessiveValues(request);

try {

request.validate();

} catch (TException e) {

String errorMsg = "Invalid EarlybirdRequest. " + request;

LOG.warn(errorMsg);

return createErrorResponse(errorMsg);

}

if (request.isSetSearchSegmentId() && request.getSearchSegmentId() <= 0) {

String errorMsg = "Bad time slice ID: " + request.getSearchSegmentId();

LOG.warn(errorMsg);

return createErrorResponse(errorMsg);

}

if (request.isSetTermStatisticsRequest()

&& request.getTermStatisticsRequest().isSetHistogramSettings()

&& request.getTermStatisticsRequest().getHistogramSettings().getNumBins() == 0) {

String errorMsg = "numBins for term statistics histograms request cannot be zero: " + request;

LOG.warn(errorMsg);

return createErrorResponse(errorMsg);

}

if (!request.isSetSearchQuery()

|| request.getSearchQuery() == null) {

String errorMsg = "Invalid EarlybirdRequest, no ThriftSearchQuery specified. " + request;

LOG.warn(errorMsg);

return createErrorResponse(errorMsg);

}

ThriftSearchQuery searchQuery = request.getSearchQuery();

if (!searchQuery.getCollectorParams().isSetNumResultsToReturn()) {

String errorMsg = "ThriftSearchQuery.numResultsToReturn not set. " + request;

LOG.warn(errorMsg);

return createErrorResponse(errorMsg);

}

if (searchQuery.getCollectorParams().getNumResultsToReturn() < 0) {

String errorMsg = "Invalid ThriftSearchQuery.collectorParams.numResultsToReturn: "

+ searchQuery.getCollectorParams().getNumResultsToReturn() + ". " + request;

LOG.warn(errorMsg);

return createErrorResponse(errorMsg);

}

if (request.isSetSuccessfulResponseThreshold()) {

double successfulResponseThreshold = request.getSuccessfulResponseThreshold();

if (successfulResponseThreshold <= 0) {

String errorMsg = "Success response threshold is below or equal to 0: "

+ successfulResponseThreshold + " request: " + request;

LOG.warn(errorMsg);

INVALID\_SUCCESS\_RESPONSE\_THRESHOLD\_TOO\_LOW.increment();

return createErrorResponse(errorMsg);

} else if (successfulResponseThreshold > 1) {

String errorMsg = "Success response threshold is above 1: " + successfulResponseThreshold

+ " request: " + request;

LOG.warn(errorMsg);

INVALID\_SUCCESS\_RESPONSE\_THRESHOLD\_TOO\_HIGH.increment();

return createErrorResponse(errorMsg);

}

}

return null;

}

}