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

import java.util.EnumMap;

import java.util.List;

import java.util.Map;

import com.google.common.annotations.VisibleForTesting;

import com.twitter.finagle.Service;

import com.twitter.finagle.SimpleFilter;

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

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

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

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

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

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

import com.twitter.util.Future;

import com.twitter.util.FutureEventListener;

/\*\*

\* Filter tracks the isUserProtected metadata stats returned from Earlybirds.

\*/

public class IsUserProtectedMetadataTrackingFilter

extends SimpleFilter<EarlybirdRequestContext, EarlybirdResponse> {

private static final String COUNTER\_PREFIX = "is\_user\_protected\_metadata\_count\_filter\_";

@VisibleForTesting

final Map<EarlybirdRequestType, SearchCounter> totalCounterByRequestTypeMap;

@VisibleForTesting

final Map<EarlybirdRequestType, SearchCounter> isProtectedCounterByRequestTypeMap;

public IsUserProtectedMetadataTrackingFilter() {

this.totalCounterByRequestTypeMap = new EnumMap<>(EarlybirdRequestType.class);

this.isProtectedCounterByRequestTypeMap = new EnumMap<>(EarlybirdRequestType.class);

for (EarlybirdRequestType requestType : EarlybirdRequestType.values()) {

this.totalCounterByRequestTypeMap.put(requestType,

SearchCounter.export(COUNTER\_PREFIX + requestType.getNormalizedName() + "\_total"));

this.isProtectedCounterByRequestTypeMap.put(requestType,

SearchCounter.export(COUNTER\_PREFIX + requestType.getNormalizedName() + "\_is\_protected"));

}

}

@Override

public Future<EarlybirdResponse> apply(

EarlybirdRequestContext request,

Service<EarlybirdRequestContext, EarlybirdResponse> service) {

Future<EarlybirdResponse> response = service.apply(request);

EarlybirdRequestType requestType = request.getEarlybirdRequestType();

response.addEventListener(new FutureEventListener<EarlybirdResponse>() {

@Override

public void onSuccess(EarlybirdResponse response) {

if (!response.isSetSearchResults() || response.getSearchResults().getResults().isEmpty()) {

return;

}

List<ThriftSearchResult> searchResults = response.getSearchResults().getResults();

int totalCount = searchResults.size();

int isUserProtectedCount = 0;

for (ThriftSearchResult searchResult : searchResults) {

if (searchResult.isSetMetadata() && searchResult.getMetadata().isSetExtraMetadata()) {

ThriftSearchResultExtraMetadata extraMetadata =

searchResult.getMetadata().getExtraMetadata();

if (extraMetadata.isIsUserProtected()) {

isUserProtectedCount++;

}

}

}

IsUserProtectedMetadataTrackingFilter.this

.totalCounterByRequestTypeMap.get(requestType).add(totalCount);

IsUserProtectedMetadataTrackingFilter.this

.isProtectedCounterByRequestTypeMap.get(requestType).add(isUserProtectedCount);

}

@Override

public void onFailure(Throwable cause) { }

});

return response;

}

}