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

import com.google.common.base.Optional;

import com.twitter.search.common.caching.CacheUtil;

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

import com.twitter.search.common.decider.SearchDecider;

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

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

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

public class RelevanceCacheRequestNormalizer extends

CacheRequestNormalizer<EarlybirdRequestContext, EarlybirdRequest> {

private static final SearchCounter RELEVANCE\_FORCE\_CACHED\_LOGGED\_IN\_REQUEST =

SearchCounter.export("relevance\_force\_cached\_logged\_in\_request");

private final SearchDecider decider;

private final String relevanceStripPersonalizationFieldsDeciderKey;

public RelevanceCacheRequestNormalizer(

SearchDecider decider,

String normalizedSearchRootName) {

this.decider = decider;

this.relevanceStripPersonalizationFieldsDeciderKey =

String.format("relevance\_%s\_force\_cache\_logged\_in\_requests", normalizedSearchRootName);

}

@Override

public Optional<EarlybirdRequest> normalizeRequest(EarlybirdRequestContext requestContext) {

boolean cacheLoggedInRequest =

decider.isAvailable(relevanceStripPersonalizationFieldsDeciderKey);

if (cacheLoggedInRequest) {

RELEVANCE\_FORCE\_CACHED\_LOGGED\_IN\_REQUEST.increment();

}

return Optional.fromNullable(CacheUtil.normalizeRequestForCache(

requestContext.getRequest(), cacheLoggedInRequest));

}

}