package com.twitter.search.earlybird\_root;

import javax.inject.Inject;

import javax.inject.Named;

import javax.inject.Singleton;

import com.twitter.finagle.Service;

import com.twitter.finagle.mtls.authorization.server.MtlsServerSessionTrackerFilter;

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

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

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

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

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

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

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

import com.twitter.search.earlybird\_root.caching.FacetsCacheFilter;

import com.twitter.search.earlybird\_root.caching.RecencyCacheFilter;

import com.twitter.search.earlybird\_root.caching.RelevanceCacheFilter;

import com.twitter.search.earlybird\_root.caching.RelevanceZeroResultsCacheFilter;

import com.twitter.search.earlybird\_root.caching.StrictRecencyCacheFilter;

import com.twitter.search.earlybird\_root.caching.TermStatsCacheFilter;

import com.twitter.search.earlybird\_root.caching.TopTweetsCacheFilter;

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

import com.twitter.util.Future;

@Singleton

public class RealtimeCgRootService implements EarlybirdService.ServiceIface {

private final Service<EarlybirdRequest, EarlybirdResponse> allFiltersAndService;

@Inject

public RealtimeCgRootService(

TopLevelExceptionHandlingFilter topLevelExceptionHandlingFilter,

ResponseCodeStatFilter responseCodeStatFilter,

LoggingFilter<EarlybirdRequest, EarlybirdResponse> loggingFilter,

RequestValidationFilter<EarlybirdRequest, EarlybirdResponse> validationFilter,

MtlsServerSessionTrackerFilter<EarlybirdRequest, EarlybirdResponse> mtlsFilter,

FinagleClientStatsFilter<EarlybirdRequest, EarlybirdResponse> finagleStatsFilter,

InitializeFilter initializeFilter,

InitializeRequestContextFilter initializeRequestContextFilter,

QueryLangStatFilter queryLangStatFilter,

DropAllProtectedOperatorFilter dropAllProtectedOperatorFilter,

QueryOperatorStatFilter queryOperatorStatFilter,

RequestResultStatsFilter requestResultStatsFilter,

PreCacheRequestTypeCountFilter preCacheCountFilter,

RecencyCacheFilter recencyCacheFilter,

RelevanceCacheFilter relevanceCacheFilter,

RelevanceZeroResultsCacheFilter relevanceZeroResultsCacheFilter,

StrictRecencyCacheFilter strictRecencyCacheFilter,

FacetsCacheFilter facetsCacheFilter,

TermStatsCacheFilter termStatsCacheFilter,

TopTweetsCacheFilter topTweetsCacheFilter,

PostCacheRequestTypeCountFilter postCacheCountFilter,

ClientIdTrackingFilter clientIdTrackingFilter,

ClientIdQuotaFilter quotaFilter,

MetadataTrackingFilter metadataTrackingFilter,

NullcastTrackingFilter nullcastTrackingFilter,

ClientRequestTimeFilter clientRequestTimeFilter,

DeadlineTimeoutStatsFilter deadlineTimeoutStatsFilter,

EarlybirdFeatureSchemaAnnotateFilter featureSchemaAnnotateFilter,

SearchPayloadSizeLocalContextFilter searchPayloadSizeLocalContextFilter,

@Named(ProtectedScatterGatherModule.NAMED\_SCATTER\_GATHER\_SERVICE)

Service<EarlybirdRequestContext, EarlybirdResponse> scatterGatherService,

StratoAttributionClientIdFilter stratoAttributionClientIdFilter) {

this.allFiltersAndService =

loggingFilter

.andThen(topLevelExceptionHandlingFilter)

.andThen(stratoAttributionClientIdFilter)

.andThen(clientRequestTimeFilter)

.andThen(searchPayloadSizeLocalContextFilter)

.andThen(responseCodeStatFilter)

.andThen(requestResultStatsFilter)

.andThen(validationFilter)

.andThen(mtlsFilter)

.andThen(finagleStatsFilter)

.andThen(clientIdTrackingFilter)

.andThen(quotaFilter)

.andThen(metadataTrackingFilter)

.andThen(initializeFilter)

.andThen(initializeRequestContextFilter)

.andThen(deadlineTimeoutStatsFilter)

.andThen(queryLangStatFilter)

.andThen(nullcastTrackingFilter)

.andThen(dropAllProtectedOperatorFilter)

.andThen(queryOperatorStatFilter)

.andThen(preCacheCountFilter)

.andThen(recencyCacheFilter)

.andThen(relevanceCacheFilter)

.andThen(relevanceZeroResultsCacheFilter)

.andThen(strictRecencyCacheFilter)

.andThen(facetsCacheFilter)

.andThen(termStatsCacheFilter)

.andThen(topTweetsCacheFilter)

.andThen(postCacheCountFilter)

.andThen(featureSchemaAnnotateFilter)

.andThen(scatterGatherService);

}

@Override

public Future<String> getName() {

return Future.value("realtime\_cg root");

}

@Override

public Future<EarlybirdStatusResponse> getStatus() {

throw new UnsupportedOperationException("not supported");

}

@Override

public Future<EarlybirdResponse> search(EarlybirdRequest request) {

return allFiltersAndService.apply(request);

}

}