package com.twitter.search.earlybird\_root;

import java.util.Arrays;

import java.util.HashMap;

import java.util.Map;

import javax.annotation.Nullable;

import javax.inject.Named;

import javax.inject.Singleton;

import com.google.inject.Provides;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

import com.twitter.finagle.Service;

import com.twitter.finagle.stats.StatsReceiver;

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

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

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

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

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

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

import com.twitter.search.common.schema.earlybird.EarlybirdCluster;

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

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

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

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

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

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

public class RealtimeScatterGatherModule extends ScatterGatherModule {

private static final Logger LOG =

LoggerFactory.getLogger(RealtimeScatterGatherModule.class);

/\*\*

\* Provides a scatter gather service for the realtime cluster that redirects to experimental

\* clusters when the experiment cluster parameter is set on the EarlybirdRequest.

\*

\* Note: if an alternate client is specified via altPartitionConfig or

\* altRootClientServiceBuilder, it will be built and used for the "control" cluster, but the

\* experiment cluster takes precedence (if the experiment cluster is set on the request, the

\* alternate client will never be used.

\*/

@Provides

@Singleton

@Named(NAMED\_SCATTER\_GATHER\_SERVICE)

@Override

public Service<EarlybirdRequestContext, EarlybirdResponse> provideScatterGatherService(

EarlybirdServiceScatterGatherSupport scatterGatherSupport,

RequestSuccessStats requestSuccessStats,

PartitionLoggingSupport<EarlybirdRequestContext> partitionLoggingSupport,

RequestContextToEarlybirdRequestFilter requestContextToEarlybirdRequestFilter,

PartitionAccessController partitionAccessController,

PartitionConfig partitionConfig,

RootClientServiceBuilder<EarlybirdService.ServiceIface> rootClientServiceBuilder,

@Named(EarlybirdCommonModule.NAMED\_EXP\_CLUSTER\_CLIENT)

RootClientServiceBuilder<EarlybirdService.ServiceIface>

expClusterRootClientServiceBuilder,

@Named(EarlybirdCommonModule.NAMED\_ALT\_CLIENT) @Nullable PartitionConfig altPartitionConfig,

@Named(EarlybirdCommonModule.NAMED\_ALT\_CLIENT) @Nullable

RootClientServiceBuilder<EarlybirdService.ServiceIface> altRootClientServiceBuilder,

StatsReceiver statsReceiver,

EarlybirdCluster cluster,

SearchDecider decider) {

Service<EarlybirdRequestContext, EarlybirdResponse> controlService =

buildScatterOrSplitterService(

scatterGatherSupport,

requestSuccessStats,

partitionLoggingSupport,

requestContextToEarlybirdRequestFilter,

partitionAccessController,

partitionConfig,

rootClientServiceBuilder,

altPartitionConfig,

altRootClientServiceBuilder,

statsReceiver,

cluster,

decider

);

Map<ExperimentCluster, ScatterGatherService<EarlybirdRequestContext, EarlybirdResponse>>

experimentScatterGatherServices = new HashMap<>();

LOG.info("Using ScatterGatherWithExperimentRedirectsService");

LOG.info("Control Partition Path: {}", partitionConfig.getPartitionPath());

Arrays.stream(ExperimentCluster.values())

.filter(v -> v.name().toLowerCase().startsWith("exp"))

.forEach(experimentCluster -> {

String expPartitionPath = partitionConfig.getPartitionPath()

+ "-" + experimentCluster.name().toLowerCase();

LOG.info("Experiment Partition Path: {}", expPartitionPath);

experimentScatterGatherServices.put(experimentCluster,

createScatterGatherService(

"",

scatterGatherSupport,

requestSuccessStats,

partitionLoggingSupport,

requestContextToEarlybirdRequestFilter,

partitionAccessController,

partitionConfig.getNumPartitions(),

expPartitionPath,

expClusterRootClientServiceBuilder,

statsReceiver,

cluster,

decider,

experimentCluster.name().toLowerCase()));

});

return new ScatterGatherWithExperimentRedirectsService(

controlService,

experimentScatterGatherServices);

}

}