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

import java.util.Map;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

import com.twitter.finagle.Service;

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

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

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

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

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

import com.twitter.util.Future;

public class ScatterGatherWithExperimentRedirectsService

extends Service<EarlybirdRequestContext, EarlybirdResponse> {

private final Service<EarlybirdRequestContext, EarlybirdResponse>

controlScatterGatherService;

private final Map<ExperimentCluster,

ScatterGatherService<EarlybirdRequestContext, EarlybirdResponse>>

experimentScatterGatherServices;

private static final Logger LOG =

LoggerFactory.getLogger(ScatterGatherWithExperimentRedirectsService.class);

public ScatterGatherWithExperimentRedirectsService(

Service<EarlybirdRequestContext, EarlybirdResponse> controlScatterGatherService,

Map<ExperimentCluster,

ScatterGatherService<EarlybirdRequestContext, EarlybirdResponse>>

experimentScatterGatherServices

) {

this.controlScatterGatherService = controlScatterGatherService;

this.experimentScatterGatherServices = experimentScatterGatherServices;

}

@Override

public Future<EarlybirdResponse> apply(EarlybirdRequestContext request) {

if (request.getRequest().isSetExperimentClusterToUse()) {

ExperimentCluster cluster = request.getRequest().getExperimentClusterToUse();

if (!experimentScatterGatherServices.containsKey(cluster)) {

String error = String.format(

"Received invalid experiment cluster: %s", cluster.name());

LOG.error("{} Request: {}", error, request.getRequest());

return Future.value(new EarlybirdResponse()

.setResponseCode(EarlybirdResponseCode.CLIENT\_ERROR)

.setDebugString(error));

}

return experimentScatterGatherServices.get(cluster).apply(request);

}

return controlScatterGatherService.apply(request);

}

}