package com.twitter.simclustersann

import com.google.inject.Module

import com.twitter.finatra.decider.modules.DeciderModule

import com.twitter.finatra.mtls.thriftmux.Mtls

import com.twitter.finatra.thrift.ThriftServer

import com.twitter.finatra.thrift.filters.\_

import com.twitter.finatra.thrift.routing.ThriftRouter

import com.twitter.inject.thrift.modules.ThriftClientIdModule

import com.twitter.relevance\_platform.common.exceptions.\_

import com.twitter.simclustersann.controllers.SimClustersANNController

import com.twitter.simclustersann.exceptions.InvalidRequestForSimClustersAnnVariantExceptionMapper

import com.twitter.simclustersann.modules.\_

import com.twitter.simclustersann.thriftscala.SimClustersANNService

import com.twitter.finagle.Filter

import com.twitter.finatra.annotations.DarkTrafficFilterType

import com.twitter.inject.annotations.Flags

import com.twitter.relevance\_platform.common.filters.DarkTrafficFilterModule

import com.twitter.relevance\_platform.common.filters.ClientStatsFilter

import com.twitter.simclustersann.common.FlagNames.DisableWarmup

object SimClustersAnnServerMain extends SimClustersAnnServer

class SimClustersAnnServer extends ThriftServer with Mtls {

flag(

name = DisableWarmup,

default = false,

help = "If true, no warmup will be run."

)

override val name = "simclusters-ann-server"

override val modules: Seq[Module] = Seq(

CacheModule,

ServiceNameMapperModule,

ClusterConfigMapperModule,

ClusterConfigModule,

ClusterTweetIndexProviderModule,

DeciderModule,

EmbeddingStoreModule,

FlagsModule,

FuturePoolProvider,

RateLimiterModule,

SimClustersANNCandidateSourceModule,

StratoClientProviderModule,

ThriftClientIdModule,

new CustomMtlsThriftWebFormsModule[SimClustersANNService.MethodPerEndpoint](this),

new DarkTrafficFilterModule[SimClustersANNService.ReqRepServicePerEndpoint]()

)

def configureThrift(router: ThriftRouter): Unit = {

router

.filter[LoggingMDCFilter]

.filter[TraceIdMDCFilter]

.filter[ThriftMDCFilter]

.filter[ClientStatsFilter]

.filter[ExceptionMappingFilter]

.filter[Filter.TypeAgnostic, DarkTrafficFilterType]

.exceptionMapper[InvalidRequestForSimClustersAnnVariantExceptionMapper]

.exceptionMapper[DeadlineExceededExceptionMapper]

.exceptionMapper[UnhandledExceptionMapper]

.add[SimClustersANNController]

}

override protected def warmup(): Unit = {

if (!injector.instance[Boolean](Flags.named(DisableWarmup))) {

handle[SimclustersAnnWarmupHandler]()

}

}

}