package com.twitter.cr\_mixer.module.similarity\_engine

import com.google.inject.Provides

import com.twitter.conversions.DurationOps.\_

import com.twitter.cr\_mixer.model.ModuleNames

import com.twitter.cr\_mixer.model.TweetWithScore

import com.twitter.cr\_mixer.config.TimeoutConfig

import com.twitter.cr\_mixer.param.decider.CrMixerDecider

import com.twitter.cr\_mixer.param.decider.DeciderConstants

import com.twitter.cr\_mixer.similarity\_engine.SimilarityEngine.\_

import com.twitter.cr\_mixer.similarity\_engine.SimilarityEngine.keyHasher

import com.twitter.cr\_mixer.similarity\_engine.SimilarityEngine.DeciderConfig

import com.twitter.cr\_mixer.similarity\_engine.SimilarityEngine.GatingConfig

import com.twitter.cr\_mixer.similarity\_engine.SimilarityEngine.SimilarityEngineConfig

import com.twitter.cr\_mixer.similarity\_engine.StandardSimilarityEngine

import com.twitter.cr\_mixer.similarity\_engine.TweetBasedQigSimilarityEngine

import com.twitter.cr\_mixer.thriftscala.SimilarityEngineType

import com.twitter.finagle.memcached.{Client => MemcachedClient}

import com.twitter.finagle.stats.StatsReceiver

import com.twitter.inject.TwitterModule

import com.twitter.qig\_ranker.thriftscala.QigRanker

import javax.inject.Named

import javax.inject.Singleton

object TweetBasedQigSimilarityEngineModule extends TwitterModule {

@Provides

@Singleton

@Named(ModuleNames.TweetBasedQigSimilarityEngine)

def providesTweetBasedQigSimilarTweetsCandidateSource(

qigRanker: QigRanker.MethodPerEndpoint,

@Named(ModuleNames.UnifiedCache) crMixerUnifiedCacheClient: MemcachedClient,

timeoutConfig: TimeoutConfig,

statsReceiver: StatsReceiver,

decider: CrMixerDecider

): StandardSimilarityEngine[

TweetBasedQigSimilarityEngine.Query,

TweetWithScore

] = {

new StandardSimilarityEngine[

TweetBasedQigSimilarityEngine.Query,

TweetWithScore

](

implementingStore = TweetBasedQigSimilarityEngine(qigRanker, statsReceiver),

identifier = SimilarityEngineType.Qig,

globalStats = statsReceiver,

engineConfig = SimilarityEngineConfig(

timeout = timeoutConfig.similarityEngineTimeout,

gatingConfig = GatingConfig(

deciderConfig =

Some(DeciderConfig(decider, DeciderConstants.enableQigSimilarTweetsTrafficDeciderKey)),

enableFeatureSwitch = None

)

),

memCacheConfig = Some(

MemCacheConfig(

cacheClient = crMixerUnifiedCacheClient,

ttl = 10.minutes,

keyToString = { k =>

f"TweetBasedQIGRanker:${keyHasher.hashKey(k.sourceId.toString.getBytes)}%X"

}

)

)

)

}

}