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

import com.google.inject.Provides

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

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

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

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

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

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

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.SkitHighPrecisionTopicTweetSimilarityEngine

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

import com.twitter.cr\_mixer.similarity\_engine.SkitTopicTweetSimilarityEngine.Query

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

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

import com.twitter.finagle.stats.StatsReceiver

import com.twitter.inject.TwitterModule

import com.twitter.storehaus.ReadableStore

import com.twitter.topic\_recos.thriftscala.TopicTweet

import com.twitter.topic\_recos.thriftscala.TopicTweetPartitionFlatKey

import javax.inject.Named

import javax.inject.Singleton

object SkitTopicTweetSimilarityEngineModule extends TwitterModule {

@Provides

@Singleton

@Named(ModuleNames.SkitHighPrecisionTopicTweetSimilarityEngine)

def providesSkitHighPrecisionTopicTweetSimilarityEngine(

@Named(ModuleNames.SkitStratoStoreName) skitStratoStore: ReadableStore[

TopicTweetPartitionFlatKey,

Seq[TopicTweet]

],

timeoutConfig: TimeoutConfig,

decider: CrMixerDecider,

statsReceiver: StatsReceiver

): StandardSimilarityEngine[

EngineQuery[Query],

TopicTweetWithScore

] = {

new StandardSimilarityEngine[EngineQuery[Query], TopicTweetWithScore](

implementingStore =

SkitHighPrecisionTopicTweetSimilarityEngine(skitStratoStore, statsReceiver),

identifier = SimilarityEngineType.SkitHighPrecisionTopicTweet,

globalStats = statsReceiver.scope(SimilarityEngineType.SkitHighPrecisionTopicTweet.name),

engineConfig = SimilarityEngineConfig(

timeout = timeoutConfig.topicTweetEndpointTimeout,

gatingConfig = GatingConfig(

deciderConfig =

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

enableFeatureSwitch = None

)

)

)

}

@Provides

@Singleton

@Named(ModuleNames.SkitTopicTweetSimilarityEngine)

def providesSkitTfgTopicTweetSimilarityEngine(

@Named(ModuleNames.SkitStratoStoreName) skitStratoStore: ReadableStore[

TopicTweetPartitionFlatKey,

Seq[TopicTweet]

],

timeoutConfig: TimeoutConfig,

decider: CrMixerDecider,

statsReceiver: StatsReceiver

): StandardSimilarityEngine[

EngineQuery[Query],

TopicTweetWithScore

] = {

new StandardSimilarityEngine[EngineQuery[Query], TopicTweetWithScore](

implementingStore = SkitTopicTweetSimilarityEngine(skitStratoStore, statsReceiver),

identifier = SimilarityEngineType.SkitTfgTopicTweet,

globalStats = statsReceiver.scope(SimilarityEngineType.SkitTfgTopicTweet.name),

engineConfig = SimilarityEngineConfig(

timeout = timeoutConfig.topicTweetEndpointTimeout,

gatingConfig = GatingConfig(

deciderConfig =

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

enableFeatureSwitch = None

)

)

)

}

}