package com.twitter.cr\_mixer.module

import com.google.inject.Provides

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

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

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

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

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

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

import com.twitter.finagle.stats.StatsReceiver

import com.twitter.inject.TwitterModule

import com.twitter.simclusters\_v2.common.TweetId

import com.twitter.simclusters\_v2.common.UserId

import com.twitter.storehaus.ReadableStore

import javax.inject.Singleton

/\*\*

\* In this example we build a [[StandardSimilarityEngine]] to wrap a dummy store

\*/

object SimpleSimilarityEngineModule extends TwitterModule {

@Provides

@Singleton

def providesSimpleSimilarityEngine(

timeoutConfig: TimeoutConfig,

globalStats: StatsReceiver

): StandardSimilarityEngine[UserId, (TweetId, Double)] = {

// Inject your readableStore implementation here

val dummyStore = ReadableStore.fromMap(

Map(

1L -> Seq((100L, 1.0), (101L, 1.0)),

2L -> Seq((200L, 2.0), (201L, 2.0)),

3L -> Seq((300L, 3.0), (301L, 3.0))

))

new StandardSimilarityEngine[UserId, (TweetId, Double)](

implementingStore = dummyStore,

identifier = SimilarityEngineType.EnumUnknownSimilarityEngineType(9997),

globalStats = globalStats,

engineConfig = SimilarityEngineConfig(

timeout = timeoutConfig.similarityEngineTimeout,

gatingConfig = GatingConfig(

deciderConfig = None,

enableFeatureSwitch = None

)

)

)

}

}

/\*\*

\* In this example we build a [[LookupSimilarityEngine]] to wrap a dummy store with 2 versions

\*/

object LookupSimilarityEngineModule extends TwitterModule {

@Provides

@Singleton

def providesLookupSimilarityEngine(

timeoutConfig: TimeoutConfig,

globalStats: StatsReceiver

): LookupSimilarityEngine[UserId, (TweetId, Double)] = {

// Inject your readableStore implementation here

val dummyStoreV1 = ReadableStore.fromMap(

Map(

1L -> Seq((100L, 1.0), (101L, 1.0)),

2L -> Seq((200L, 2.0), (201L, 2.0)),

))

val dummyStoreV2 = ReadableStore.fromMap(

Map(

1L -> Seq((100L, 1.0), (101L, 1.0)),

2L -> Seq((200L, 2.0), (201L, 2.0)),

))

new LookupSimilarityEngine[UserId, (TweetId, Double)](

versionedStoreMap = Map(

"V1" -> dummyStoreV1,

"V2" -> dummyStoreV2

),

identifier = SimilarityEngineType.EnumUnknownSimilarityEngineType(9998),

globalStats = globalStats,

engineConfig = SimilarityEngineConfig(

timeout = timeoutConfig.similarityEngineTimeout,

gatingConfig = GatingConfig(

deciderConfig = None,

enableFeatureSwitch = None

)

)

)

}

}