package com.twitter.cr\_mixer.module

import com.google.inject.Provides

import com.twitter.bijection.Injection

import com.twitter.bijection.scrooge.BinaryScalaCodec

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

import com.twitter.finagle.mtls.authentication.ServiceIdentifier

import com.twitter.inject.TwitterModule

import com.twitter.simclusters\_v2.thriftscala.TweetsWithScore

import com.twitter.storage.client.manhattan.kv.ManhattanKVClientMtlsParams

import com.twitter.storehaus.ReadableStore

import com.twitter.storehaus\_internal.manhattan.Apollo

import com.twitter.storehaus\_internal.manhattan.ManhattanRO

import com.twitter.storehaus\_internal.manhattan.ManhattanROConfig

import com.twitter.storehaus\_internal.util.ApplicationID

import com.twitter.storehaus\_internal.util.DatasetName

import com.twitter.storehaus\_internal.util.HDFSPath

import javax.inject.Named

import javax.inject.Singleton

object DiffusionStoreModule extends TwitterModule {

type UserId = Long

implicit val longCodec = implicitly[Injection[Long, Array[Byte]]]

implicit val tweetRecsInjection: Injection[TweetsWithScore, Array[Byte]] =

BinaryScalaCodec(TweetsWithScore)

@Provides

@Singleton

@Named(ModuleNames.RetweetBasedDiffusionRecsMhStore)

def retweetBasedDiffusionRecsMhStore(

serviceIdentifier: ServiceIdentifier

): ReadableStore[Long, TweetsWithScore] = {

val manhattanROConfig = ManhattanROConfig(

HDFSPath(""), // not needed

ApplicationID("cr\_mixer\_apollo"),

DatasetName("diffusion\_retweet\_tweet\_recs"),

Apollo

)

buildTweetRecsStore(serviceIdentifier, manhattanROConfig)

}

private def buildTweetRecsStore(

serviceIdentifier: ServiceIdentifier,

manhattanROConfig: ManhattanROConfig

): ReadableStore[Long, TweetsWithScore] = {

ManhattanRO

.getReadableStoreWithMtls[Long, TweetsWithScore](

manhattanROConfig,

ManhattanKVClientMtlsParams(serviceIdentifier)

)(longCodec, tweetRecsInjection)

}

}