package com.twitter.cr\_mixer.module

import com.google.inject.Provides

import com.twitter.bijection.Injection

import com.twitter.bijection.scrooge.CompactScalaCodec

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.CandidateTweetsList

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 OfflineCandidateStoreModule extends TwitterModule {

type UserId = Long

implicit val tweetCandidatesInjection: Injection[CandidateTweetsList, Array[Byte]] =

CompactScalaCodec(CandidateTweetsList)

@Provides

@Singleton

@Named(ModuleNames.OfflineTweet2020CandidateStore)

def offlineTweet2020CandidateMhStore(

serviceIdentifier: ServiceIdentifier

): ReadableStore[UserId, CandidateTweetsList] = {

buildOfflineCandidateStore(

serviceIdentifier,

datasetName = "offline\_tweet\_recommendations\_from\_interestedin\_2020"

)

}

@Provides

@Singleton

@Named(ModuleNames.OfflineTweet2020Hl0El15CandidateStore)

def offlineTweet2020Hl0El15CandidateMhStore(

serviceIdentifier: ServiceIdentifier

): ReadableStore[UserId, CandidateTweetsList] = {

buildOfflineCandidateStore(

serviceIdentifier,

datasetName = "offline\_tweet\_recommendations\_from\_interestedin\_2020\_hl\_0\_el\_15"

)

}

@Provides

@Singleton

@Named(ModuleNames.OfflineTweet2020Hl2El15CandidateStore)

def offlineTweet2020Hl2El15CandidateMhStore(

serviceIdentifier: ServiceIdentifier

): ReadableStore[UserId, CandidateTweetsList] = {

buildOfflineCandidateStore(

serviceIdentifier,

datasetName = "offline\_tweet\_recommendations\_from\_interestedin\_2020\_hl\_2\_el\_15"

)

}

@Provides

@Singleton

@Named(ModuleNames.OfflineTweet2020Hl2El50CandidateStore)

def offlineTweet2020Hl2El50CandidateMhStore(

serviceIdentifier: ServiceIdentifier

): ReadableStore[UserId, CandidateTweetsList] = {

buildOfflineCandidateStore(

serviceIdentifier,

datasetName = "offline\_tweet\_recommendations\_from\_interestedin\_2020\_hl\_2\_el\_50"

)

}

@Provides

@Singleton

@Named(ModuleNames.OfflineTweet2020Hl8El50CandidateStore)

def offlineTweet2020Hl8El50CandidateMhStore(

serviceIdentifier: ServiceIdentifier

): ReadableStore[UserId, CandidateTweetsList] = {

buildOfflineCandidateStore(

serviceIdentifier,

datasetName = "offline\_tweet\_recommendations\_from\_interestedin\_2020\_hl\_8\_el\_50"

)

}

@Provides

@Singleton

@Named(ModuleNames.OfflineTweetMTSCandidateStore)

def offlineTweetMTSCandidateMhStore(

serviceIdentifier: ServiceIdentifier

): ReadableStore[UserId, CandidateTweetsList] = {

buildOfflineCandidateStore(

serviceIdentifier,

datasetName = "offline\_tweet\_recommendations\_from\_mts\_consumer\_embeddings"

)

}

@Provides

@Singleton

@Named(ModuleNames.OfflineFavDecayedSumCandidateStore)

def offlineFavDecayedSumCandidateStore(

serviceIdentifier: ServiceIdentifier

): ReadableStore[UserId, CandidateTweetsList] = {

buildOfflineCandidateStore(

serviceIdentifier,

datasetName = "offline\_tweet\_recommendations\_from\_decayed\_sum"

)

}

@Provides

@Singleton

@Named(ModuleNames.OfflineFtrAt5Pop1000RankDecay11CandidateStore)

def offlineFtrAt5Pop1000RankDecay11CandidateStore(

serviceIdentifier: ServiceIdentifier

): ReadableStore[UserId, CandidateTweetsList] = {

buildOfflineCandidateStore(

serviceIdentifier,

datasetName = "offline\_tweet\_recommendations\_from\_ftrat5\_pop1000\_rank\_decay\_1\_1"

)

}

@Provides

@Singleton

@Named(ModuleNames.OfflineFtrAt5Pop10000RankDecay11CandidateStore)

def offlineFtrAt5Pop10000RankDecay11CandidateStore(

serviceIdentifier: ServiceIdentifier

): ReadableStore[UserId, CandidateTweetsList] = {

buildOfflineCandidateStore(

serviceIdentifier,

datasetName = "offline\_tweet\_recommendations\_from\_ftrat5\_pop10000\_rank\_decay\_1\_1"

)

}

private def buildOfflineCandidateStore(

serviceIdentifier: ServiceIdentifier,

datasetName: String

): ReadableStore[UserId, CandidateTweetsList] = {

ManhattanRO

.getReadableStoreWithMtls[Long, CandidateTweetsList](

ManhattanROConfig(

HDFSPath(""), // not needed

ApplicationID("multi\_type\_simclusters"),

DatasetName(datasetName),

Apollo

),

ManhattanKVClientMtlsParams(serviceIdentifier)

)

}

}