package com.twitter.simclusters\_v2.summingbird.common

import com.twitter.conversions.DurationOps.\_

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

import com.twitter.finagle.thrift.ClientId

import com.twitter.storehaus\_internal.memcache.ConnectionConfig

import com.twitter.storehaus\_internal.memcache.MemcacheConfig

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

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

import com.twitter.strato.client.Strato

import com.twitter.strato.client.{Client => StratoClient}

object ClientConfigs {

com.twitter.server.Init() // necessary in order to use WilyNS path

final lazy val simClustersCoreAltCachePath =

"/srv#/prod/local/cache/simclusters\_core\_alt"

final lazy val simClustersCoreAltLightCachePath =

"/srv#/prod/local/cache/simclusters\_core\_alt\_light"

final lazy val develSimClustersCoreCachePath =

"/srv#/test/local/cache/twemcache\_simclusters\_core"

final lazy val develSimClustersCoreLightCachePath =

"/srv#/test/local/cache/twemcache\_simclusters\_core\_light"

final lazy val logFavBasedTweet20M145K2020StratoPath =

"recommendations/simclusters\_v2/embeddings/logFavBasedTweet20M145K2020Persistent"

final lazy val logFavBasedTweet20M145K2020UncachedStratoPath =

"recommendations/simclusters\_v2/embeddings/logFavBasedTweet20M145K2020-UNCACHED"

final lazy val develLogFavBasedTweet20M145K2020StratoPath =

"recommendations/simclusters\_v2/embeddings/logFavBasedTweet20M145K2020Devel"

final lazy val entityClusterScoreMemcacheConfig: (String, ServiceIdentifier) => MemcacheConfig = {

(path: String, serviceIdentifier: ServiceIdentifier) =>

new MemcacheConfig {

val connectionConfig: ConnectionConfig = ConnectionConfig(path, serviceIdentifier = serviceIdentifier)

override val keyPrefix: KeyPrefix = KeyPrefix(s"ecs\_")

override val ttl: TTL = TTL(8.hours)

}

}

// note: this should in dedicated cache for tweet

final lazy val tweetTopKClustersMemcacheConfig: (String, ServiceIdentifier) => MemcacheConfig = {

(path: String, serviceIdentifier: ServiceIdentifier) =>

new MemcacheConfig {

val connectionConfig: ConnectionConfig =

ConnectionConfig(path, serviceIdentifier = serviceIdentifier)

override val keyPrefix: KeyPrefix = KeyPrefix(s"etk\_")

override val ttl: TTL = TTL(2.days)

}

}

// note: this should in dedicated cache for tweet

final lazy val clusterTopTweetsMemcacheConfig: (String, ServiceIdentifier) => MemcacheConfig = {

(path: String, serviceIdentifier: ServiceIdentifier) =>

new MemcacheConfig {

val connectionConfig: ConnectionConfig =

ConnectionConfig(path, serviceIdentifier = serviceIdentifier)

override val keyPrefix: KeyPrefix = KeyPrefix(s"ctkt\_")

override val ttl: TTL = TTL(8.hours)

}

}

final lazy val stratoClient: ServiceIdentifier => StratoClient = { serviceIdentifier =>

Strato.client

.withRequestTimeout(2.seconds)

.withMutualTls(serviceIdentifier)

.build()

}

// thrift client id

private final lazy val thriftClientId: String => ClientId = { env: String =>

ClientId(s"simclusters\_v2\_summingbird.$env")

}

}