package com.twitter.home\_mixer.module

import com.google.inject.Provides

import com.twitter.conversions.DurationOps.\_

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

import com.twitter.finagle.stats.StatsReceiver

import com.twitter.inject.TwitterModule

import com.twitter.inject.annotations.Flag

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

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

import com.twitter.timelines.clients.manhattan.mhv3.ManhattanClientBuilder

import com.twitter.timelines.impressionstore.store.ManhattanTweetImpressionStoreClientConfig

import com.twitter.timelines.impressionstore.store.ManhattanTweetImpressionStoreClient

import com.twitter.util.Duration

import javax.inject.Singleton

object ManhattanTweetImpressionStoreModule extends TwitterModule {

private val ProdAppId = "timelines\_tweet\_impression\_store\_v2"

private val ProdDataset = "timelines\_tweet\_impressions\_v2"

private val StagingAppId = "timelines\_tweet\_impression\_store\_staging"

private val StagingDataset = "timelines\_tweet\_impressions\_staging"

private val StatsScope = "manhattanTweetImpressionStoreClient"

private val DefaultTTL = 2.days

private final val Timeout = "mh\_impression\_store.timeout"

flag[Duration](Timeout, 150.millis, "Timeout per request")

@Provides

@Singleton

def providesManhattanTweetImpressionStoreClient(

@Flag(Timeout) timeout: Duration,

serviceIdentifier: ServiceIdentifier,

statsReceiver: StatsReceiver

): ManhattanTweetImpressionStoreClient = {

val (appId, dataset) = serviceIdentifier.environment.toLowerCase match {

case "prod" => (ProdAppId, ProdDataset)

case \_ => (StagingAppId, StagingDataset)

}

val config = ManhattanTweetImpressionStoreClientConfig(

cluster = ManhattanClusters.nash,

appId = appId,

dataset = dataset,

statsScope = StatsScope,

defaultGuarantee = Guarantee.SoftDcReadMyWrites,

defaultMaxTimeout = timeout,

maxRetryCount = 2,

isReadOnly = false,

serviceIdentifier = serviceIdentifier,

ttl = DefaultTTL

)

val manhattanEndpoint = ManhattanClientBuilder.buildManhattanEndpoint(config, statsReceiver)

ManhattanTweetImpressionStoreClient(config, manhattanEndpoint, statsReceiver)

}

}