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.timelinemixer.clients.persistence.TimelinePersistenceManhattanClientBuilder

import com.twitter.timelinemixer.clients.persistence.TimelinePersistenceManhattanClientConfig

import com.twitter.timelinemixer.clients.persistence.TimelineResponseBatchesClient

import com.twitter.timelinemixer.clients.persistence.TimelineResponseV3

import com.twitter.util.Duration

import javax.inject.Singleton

object TimelinesPersistenceStoreClientModule extends TwitterModule {

private val StagingDataset = "timeline\_response\_batches\_v5\_nonprod"

private val ProdDataset = "timeline\_response\_batches\_v5"

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

flag[Duration](Timeout, 300.millis, "Timeout per request")

@Provides

@Singleton

def providesTimelinesPersistenceStoreClient(

@Flag(Timeout) timeout: Duration,

injectedServiceIdentifier: ServiceIdentifier,

statsReceiver: StatsReceiver

): TimelineResponseBatchesClient[TimelineResponseV3] = {

val timelineResponseBatchesDataset =

injectedServiceIdentifier.environment.toLowerCase match {

case "prod" => ProdDataset

case \_ => StagingDataset

}

val timelineResponseBatchesConfig = new TimelinePersistenceManhattanClientConfig {

val dataset = timelineResponseBatchesDataset

val isReadOnly = false

val serviceIdentifier = injectedServiceIdentifier

override val defaultMaxTimeout = timeout

override val maxRetryCount = 2

}

TimelinePersistenceManhattanClientBuilder.buildTimelineResponseV3BatchesClient(

timelineResponseBatchesConfig,

statsReceiver

)

}

}