package com.twitter.home\_mixer.product.scored\_tweets.gate

import com.twitter.conversions.DurationOps.\_

import com.twitter.home\_mixer.model.HomeFeatures.LastNonPollingTimeFeature

import com.twitter.product\_mixer.core.functional\_component.gate.Gate

import com.twitter.product\_mixer.core.model.common.identifier.GateIdentifier

import com.twitter.product\_mixer.core.pipeline.PipelineQuery

import com.twitter.stitch.Stitch

/\*\*

\* Gate continues if the amount of time passed since the previous request is greater

\* than the configured amount or if the previous request time in not available

\*/

object MinTimeSinceLastRequestGate extends Gate[PipelineQuery] {

override val identifier: GateIdentifier = GateIdentifier("TimeSinceLastRequest")

private val MinTimeSinceLastRequest = 24.hours

override def shouldContinue(query: PipelineQuery): Stitch[Boolean] = Stitch.value {

query.features.exists { features =>

features

.getOrElse(LastNonPollingTimeFeature, None)

.forall(lnpt => (query.queryTime - lnpt) > MinTimeSinceLastRequest)

}

}

}