package com.twitter.recosinjector.filters

import com.twitter.finagle.stats.StatsReceiver

import com.twitter.gizmoduck.thriftscala.{LabelValue, User}

import com.twitter.recosinjector.clients.Gizmoduck

import com.twitter.util.Future

class UserFilter(

gizmoduck: Gizmoduck

)(

implicit statsReceiver: StatsReceiver) {

private val stats = statsReceiver.scope(this.getClass.getSimpleName)

private val requests = stats.counter("requests")

private val filtered = stats.counter("filtered")

private def isUnsafe(user: User): Boolean =

user.safety.exists { s =>

s.deactivated || s.suspended || s.restricted || s.nsfwUser || s.nsfwAdmin || s.isProtected

}

private def hasNsfwHighPrecisionLabel(user: User): Boolean =

user.labels.exists {

\_.labels.exists(\_.labelValue == LabelValue.NsfwHighPrecision)

}

/\*\*

\* NOTE: This will by-pass Gizmoduck's safety level, and might allow invalid users to pass filter.

\* Consider using filterByUserId instead.

\* Return true if the user is valid, otherwise return false.

\* It will first attempt to use the user object provided by the caller, and will call Gizmoduck

\* to back fill if the caller does not provide it. This helps reduce Gizmoduck traffic.

\*/

def filterByUser(

userId: Long,

userOpt: Option[User] = None

): Future[Boolean] = {

requests.incr()

val userFut = userOpt match {

case Some(user) => Future(Some(user))

case \_ => gizmoduck.getUser(userId)

}

userFut.map(\_.exists { user =>

val isValidUser = !isUnsafe(user) && !hasNsfwHighPrecisionLabel(user)

if (!isValidUser) filtered.incr()

isValidUser

})

}

/\*\*

\* Given a userId, return true if the user is valid. This id done in 2 steps:

\* 1. Applying Gizmoduck's safety level while querying for the user from Gizmoduck

\* 2. If a user passes Gizmoduck's safety level, check its specific user status

\*/

def filterByUserId(userId: Long): Future[Boolean] = {

requests.incr()

gizmoduck

.getUser(userId)

.map { userOpt =>

val isValidUser = userOpt.exists { user =>

!(isUnsafe(user) || hasNsfwHighPrecisionLabel(user))

}

if (!isValidUser) {

filtered.incr()

}

isValidUser

}

}

}