package com.twitter.tweetypie

package repository

import com.twitter.search.blender.services.strato.UserSearchSafetySettings

import com.twitter.spam.rtf.thriftscala.SafetyLabel

import com.twitter.spam.rtf.thriftscala.SafetyLabelMap

import com.twitter.spam.rtf.thriftscala.SafetyLabelType

import com.twitter.stitch.Stitch

import com.twitter.strato.client.Fetcher

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

import com.twitter.strato.thrift.ScroogeConvImplicits.\_

import com.twitter.visibility.common.UserSearchSafetySource

object StratoSafetyLabelsRepository {

type Type = (TweetId, SafetyLabelType) => Stitch[Option[SafetyLabel]]

def apply(client: StratoClient): Type = {

val safetyLabelMapRepo = StratoSafetyLabelMapRepository(client)

(tweetId, safetyLabelType) =>

safetyLabelMapRepo(tweetId).map(

\_.flatMap(\_.labels).flatMap(\_.get(safetyLabelType))

)

}

}

object StratoSafetyLabelMapRepository {

type Type = TweetId => Stitch[Option[SafetyLabelMap]]

val column = "visibility/baseTweetSafetyLabelMap"

def apply(client: StratoClient): Type = {

val fetcher: Fetcher[TweetId, Unit, SafetyLabelMap] =

client.fetcher[TweetId, SafetyLabelMap](column)

tweetId => fetcher.fetch(tweetId).map(\_.v)

}

}

object StratoUserSearchSafetySourceRepository {

type Type = UserId => Stitch[UserSearchSafetySettings]

def apply(client: StratoClient): Type = {

val fetcher: Fetcher[UserId, Unit, UserSearchSafetySettings] =

client.fetcher[UserId, UserSearchSafetySettings](UserSearchSafetySource.Column)

userId => fetcher.fetch(userId).map(\_.v.getOrElse(UserSearchSafetySource.DefaultSetting))

}

}