package com.twitter.tweetypie

package repository

import com.twitter.stitch.NotFound

import com.twitter.stitch.Stitch

import com.twitter.tweetypie.core.FilteredState.Unavailable.BounceDeleted

import com.twitter.tweetypie.core.FilteredState.Unavailable.SourceTweetNotFound

import com.twitter.tweetypie.core.FilteredState.Unavailable.TweetDeleted

object ParentUserIdRepository {

type Type = Tweet => Stitch[Option[UserId]]

case class ParentTweetNotFound(tweetId: TweetId) extends Exception

def apply(tweetRepo: TweetRepository.Type): Type = {

val options = TweetQuery.Options(TweetQuery.Include(Set(Tweet.CoreDataField.id)))

tweet =>

getShare(tweet) match {

case Some(share) if share.sourceStatusId == share.parentStatusId =>

Stitch.value(Some(share.sourceUserId))

case Some(share) =>

tweetRepo(share.parentStatusId, options)

.map(tweet => Some(getUserId(tweet)))

.rescue {

case NotFound | TweetDeleted | BounceDeleted | SourceTweetNotFound(\_) =>

Stitch.exception(ParentTweetNotFound(share.parentStatusId))

}

case None =>

Stitch.None

}

}

}