package com.twitter.tweetypie.store

import com.twitter.tweetypie.FutureEffect

import com.twitter.tweetypie.thriftscala.AsyncWriteAction

import com.twitter.tweetypie.thriftscala.RetweetArchivalEvent

trait RetweetArchivalEnqueueStore

extends TweetStoreBase[RetweetArchivalEnqueueStore]

with AsyncSetRetweetVisibility.Store {

def wrap(w: TweetStore.Wrap): RetweetArchivalEnqueueStore =

new TweetStoreWrapper(w, this)

with RetweetArchivalEnqueueStore

with AsyncSetRetweetVisibility.StoreWrapper

}

object RetweetArchivalEnqueueStore {

def apply(enqueue: FutureEffect[RetweetArchivalEvent]): RetweetArchivalEnqueueStore =

new RetweetArchivalEnqueueStore {

override val asyncSetRetweetVisibility: FutureEffect[AsyncSetRetweetVisibility.Event] =

FutureEffect[AsyncSetRetweetVisibility.Event] { e =>

enqueue(

RetweetArchivalEvent(

retweetId = e.retweetId,

srcTweetId = e.srcId,

retweetUserId = e.retweetUserId,

srcTweetUserId = e.srcTweetUserId,

timestampMs = e.timestamp.inMillis,

isArchivingAction = Some(!e.visible)

)

)

}

override val retryAsyncSetRetweetVisibility: FutureEffect[

TweetStoreRetryEvent[AsyncSetRetweetVisibility.Event]

] =

TweetStore.retry(AsyncWriteAction.RetweetArchivalEnqueue, asyncSetRetweetVisibility)

}

}