package com.twitter.tweetypie

package hydrator

import com.twitter.tco\_util.DisplayUrl

import com.twitter.tweetutil.TweetPermalink

import com.twitter.tweetypie.core.\_

import com.twitter.tweetypie.repository.\_

import com.twitter.tweetypie.thriftscala.\_

/\*\*

\* This populates expanded URL and display text in ShortenedUrl struct,

\* which is part of QuotedTweet metadata. We are using User Identity repo

\* to retrieve user's current screen-name to construct expanded url, instead

\* of relying on URL hydration.

\*

\* Expanded urls contain a mutable screen name and an immutable tweetId.

\* when visiting the link, you're always redirected to the link with

\* correct screen name - therefore, it's okay to have permalinks containing

\* old screen names that have since been changed by their user in the cache.

\* Keys will be auto-refreshed based on the 14 days TTL, we can also have

\* a daemon flush the keys with screen-name change.

\*

\*/

object QuotedTweetRefUrlsHydrator {

type Type = ValueHydrator[Option[QuotedTweet], TweetCtx]

/\*\*

\* Return true if longUrl is not set or if a prior hydration set it to shortUrl due to

\* a partial (to re-attempt hydration).

\*/

def needsHydration(s: ShortenedUrl): Boolean =

s.longUrl.isEmpty || s.displayText.isEmpty || s.longUrl == s.shortUrl

def apply(repo: UserIdentityRepository.Type): Type = {

ValueHydrator[QuotedTweet, TweetCtx] { (curr, \_) =>

repo(UserKey(curr.userId)).liftToTry.map { r =>

// we verify curr.permalink.exists pre-hydration

val shortUrl = curr.permalink.get.shortUrl

val expandedUrl = r match {

case Return(user) => TweetPermalink(user.screenName, curr.tweetId).httpsUrl

case Throw(\_) => shortUrl // fall-back to shortUrl as expandedUrl

}

ValueState.delta(

curr,

curr.copy(

permalink = Some(

ShortenedUrl(

shortUrl,

expandedUrl,

DisplayUrl.truncateUrl(expandedUrl, true)

)

)

)

)

}

}

}.onlyIf { (curr, ctx) =>

curr.permalink.exists(needsHydration) &&

ctx.tweetFieldRequested(Tweet.QuotedTweetField) && !ctx.isRetweet

}.liftOption

}