package com.twitter.frigate.pushservice.model.ntab

import com.twitter.frigate.common.base.TrendTweetCandidate

import com.twitter.frigate.common.base.TweetAuthorDetails

import com.twitter.frigate.common.base.TweetCandidate

import com.twitter.frigate.pushservice.exception.TweetNTabRequestHydratorException

import com.twitter.frigate.pushservice.model.PushTypes.PushCandidate

import com.twitter.frigate.pushservice.params.PushFeatureSwitchParams

import com.twitter.frigate.pushservice.take.NotificationServiceSender

import com.twitter.frigate.pushservice.util.EmailLandingPageExperimentUtil

import com.twitter.notificationservice.thriftscala.DisplayText

import com.twitter.notificationservice.thriftscala.DisplayTextEntity

import com.twitter.notificationservice.thriftscala.TextValue

import com.twitter.util.Future

trait TrendTweetNtabHydrator extends TweetNTabRequestHydrator {

self: PushCandidate with TrendTweetCandidate with TweetCandidate with TweetAuthorDetails =>

private lazy val trendTweetNtabStats = self.statsReceiver.scope("trend\_tweet\_ntab")

private lazy val ruxLandingOnNtabCounter =

trendTweetNtabStats.counter("use\_rux\_landing\_on\_ntab")

override lazy val displayTextEntitiesFut: Future[Seq[DisplayTextEntity]] =

NotificationServiceSender

.getDisplayTextEntityFromUser(tweetAuthor, fieldName = "author\_name", isBold = true)

.map(

\_.toSeq :+ DisplayTextEntity(

name = "trend\_name",

value = TextValue.Text(trendName),

emphasis = true)

)

override lazy val facepileUsersFut: Future[Seq[Long]] = senderIdFut.map(Seq(\_))

override lazy val socialProofDisplayText: Option[DisplayText] = None

override def refreshableType: Option[String] = ntabCopy.refreshableType

override lazy val tapThroughFut: Future[String] = {

Future.join(tweetAuthor, target.deviceInfo).map {

case (Some(author), Some(deviceInfo)) =>

val enableRuxLandingPage = deviceInfo.isRuxLandingPageEligible && target.params(

PushFeatureSwitchParams.EnableNTabRuxLandingPage)

val authorProfile = author.profile.getOrElse(

throw new TweetNTabRequestHydratorException(

s"Unable to obtain author profile for: ${author.id}"))

if (enableRuxLandingPage) {

ruxLandingOnNtabCounter.incr()

EmailLandingPageExperimentUtil.createNTabRuxLandingURI(authorProfile.screenName, tweetId)

} else {

s"${authorProfile.screenName}/status/${tweetId.toString}"

}

case \_ =>

throw new TweetNTabRequestHydratorException(

s"Unable to obtain author and target details to generate tap through for Tweet: $tweetId")

}

}

}