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

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

import com.twitter.notificationservice.thriftscala.CreateGenericNotificationRequest

import com.twitter.notificationservice.thriftscala.DisplayText

import com.twitter.notificationservice.thriftscala.DisplayTextEntity

import com.twitter.notificationservice.thriftscala.GenericType

import com.twitter.notificationservice.thriftscala.InlineCard

import com.twitter.notificationservice.thriftscala.StoryContext

import com.twitter.notificationservice.thriftscala.TapThroughAction

import com.twitter.util.Future

import com.twitter.util.Time

trait NTabRequestHydrator extends NTabRequest with CandidateNTabCopy {

self: PushCandidate =>

// Represents the sender of the recommendation

def senderIdFut: Future[Long]

// Consists of a sequence representing the social context user ids.

def facepileUsersFut: Future[Seq[Long]]

// Story Context is required for Tweet Recommendations

// Contains the Tweet ID of the recommended Tweet

def storyContext: Option[StoryContext]

// Inline card used to render a generic notification.

def inlineCard: Option[InlineCard]

// Represents where the recommendation should land when clicked

def tapThroughFut: Future[String]

// Hydration for fields that are used within the NTab copy

def displayTextEntitiesFut: Future[Seq[DisplayTextEntity]]

// Represents the social proof text that is needed for specific NTab copies

def socialProofDisplayText: Option[DisplayText]

// MagicRecs NTab entries always use RefreshableType as the Generic Type

final val genericType: GenericType = GenericType.RefreshableNotification

def refreshableType: Option[String] = ntabCopy.refreshableType

lazy val ntabRequest: Future[Option[CreateGenericNotificationRequest]] = {

Future.join(senderIdFut, displayTextEntitiesFut, facepileUsersFut, tapThroughFut).map {

case (senderId, displayTextEntities, facepileUsers, tapThrough) =>

Some(

CreateGenericNotificationRequest(

userId = target.targetId,

senderId = senderId,

genericType = GenericType.RefreshableNotification,

displayText = DisplayText(values = displayTextEntities),

facepileUsers = facepileUsers,

timestampMillis = Time.now.inMillis,

tapThroughAction = Some(TapThroughAction(Some(tapThrough))),

impressionId = Some(impressionId),

socialProofText = socialProofDisplayText,

context = storyContext,

inlineCard = inlineCard,

refreshableType = refreshableType

))

}

}

}