package com.twitter.follow\_recommendations.models

import com.twitter.follow\_recommendations.common.feature\_hydration.common.HasPreFetchedFeature

import com.twitter.follow\_recommendations.common.models.\_

import com.twitter.follow\_recommendations.logging.{thriftscala => offline}

import com.twitter.product\_mixer.core.model.marshalling.request.ClientContext

import com.twitter.product\_mixer.core.model.marshalling.request.HasClientContext

import com.twitter.timelines.configapi.HasParams

import com.twitter.timelines.configapi.Params

case class ScoringUserRequest(

override val clientContext: ClientContext,

override val displayLocation: DisplayLocation,

override val params: Params,

override val debugOptions: Option[DebugOptions] = None,

override val recentFollowedUserIds: Option[Seq[Long]],

override val recentFollowedByUserIds: Option[Seq[Long]],

override val wtfImpressions: Option[Seq[WtfImpression]],

override val similarToUserIds: Seq[Long],

candidates: Seq[CandidateUser],

debugParams: Option[DebugParams] = None,

isSoftUser: Boolean = false)

extends HasClientContext

with HasDisplayLocation

with HasParams

with HasDebugOptions

with HasPreFetchedFeature

with HasSimilarToContext {

def toOfflineThrift: offline.OfflineScoringUserRequest = offline.OfflineScoringUserRequest(

ClientContextConverter.toFRSOfflineClientContextThrift(clientContext),

displayLocation.toOfflineThrift,

candidates.map(\_.toOfflineUserThrift)

)

def toRecommendationRequest: RecommendationRequest = RecommendationRequest(

clientContext = clientContext,

displayLocation = displayLocation,

displayContext = None,

maxResults = None,

cursor = None,

excludedIds = None,

fetchPromotedContent = None,

debugParams = debugParams,

isSoftUser = isSoftUser

)

}