package com.twitter.follow\_recommendations.common.candidate\_sources.recent\_engagement

import com.twitter.follow\_recommendations.common.clients.real\_time\_real\_graph.RealTimeRealGraphClient

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

import com.twitter.hermit.model.Algorithm

import com.twitter.product\_mixer.core.functional\_component.candidate\_source.CandidateSource

import com.twitter.product\_mixer.core.model.common.identifier.CandidateSourceIdentifier

import com.twitter.stitch.Stitch

import javax.inject.Inject

import javax.inject.Singleton

@Singleton

class RecentEngagementDirectFollowSource @Inject() (

realTimeRealGraphClient: RealTimeRealGraphClient)

extends CandidateSource[Long, CandidateUser] {

val identifier: CandidateSourceIdentifier =

RecentEngagementDirectFollowSource.Identifier

/\*\*

\* Generate a list of candidates for the target using RealtimeGraphClient

\* and RecentEngagementStore.

\*/

override def apply(targetUserId: Long): Stitch[Seq[CandidateUser]] = {

realTimeRealGraphClient

.getUsersRecentlyEngagedWith(

userId = targetUserId,

engagementScoreMap = RealTimeRealGraphClient.EngagementScoreMap,

includeDirectFollowCandidates = true,

includeNonDirectFollowCandidates = false

)

.map(\_.map(\_.withCandidateSource(identifier)).sortBy(-\_.score.getOrElse(0.0)))

}

}

object RecentEngagementDirectFollowSource {

val Identifier = CandidateSourceIdentifier(Algorithm.RecentEngagementDirectFollow.toString)

}