package com.twitter.follow\_recommendations.common.predicates

import com.twitter.follow\_recommendations.common.base.Predicate

import com.twitter.follow\_recommendations.common.base.PredicateResult

import com.twitter.follow\_recommendations.common.models.FilterReason.ExcludedId

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

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

import com.twitter.stitch.Stitch

object ExcludedUserIdPredicate extends Predicate[(HasExcludedUserIds, CandidateUser)] {

val ValidStitch: Stitch[PredicateResult.Valid.type] = Stitch.value(PredicateResult.Valid)

val ExcludedStitch: Stitch[PredicateResult.Invalid] =

Stitch.value(PredicateResult.Invalid(Set(ExcludedId)))

override def apply(pair: (HasExcludedUserIds, CandidateUser)): Stitch[PredicateResult] = {

val (excludedUserIds, candidate) = pair

if (excludedUserIds.excludedUserIds.contains(candidate.id)) {

ExcludedStitch

} else {

ValidStitch

}

}

}