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

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

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

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

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

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

import com.twitter.stitch.Stitch

import javax.inject.Singleton

@Singleton

class DismissedCandidatePredicate extends Predicate[(HasDismissedUserIds, CandidateUser)] {

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

val (targetUser, candidate) = pair

targetUser.dismissedUserIds

.map { dismissedUserIds =>

if (!dismissedUserIds.contains(candidate.id)) {

DismissedCandidatePredicate.ValidStitch

} else {

DismissedCandidatePredicate.DismissedStitch

}

}.getOrElse(DismissedCandidatePredicate.ValidStitch)

}

}

object DismissedCandidatePredicate {

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

val DismissedStitch: Stitch[PredicateResult.Invalid] =

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

}