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

sealed trait FilterReason {

def reason: String

}

object FilterReason {

case object NoReason extends FilterReason {

override val reason: String = "no\_reason"

}

case class ParamReason(paramName: String) extends FilterReason {

override val reason: String = s"param\_$paramName"

}

case object ExcludedId extends FilterReason {

override val reason: String = "excluded\_id\_from\_request"

}

case object ProfileSidebarBlacklist extends FilterReason {

override val reason: String = "profile\_sidebar\_blacklisted\_id"

}

case object CuratedAccountsCompetitorList extends FilterReason {

override val reason: String = "curated\_blacklisted\_id"

}

case class InvalidRelationshipTypes(relationshipTypes: String) extends FilterReason {

override val reason: String = s"invalid\_relationship\_types $relationshipTypes"

}

case object ProfileId extends FilterReason {

override val reason: String = "candidate\_has\_same\_id\_as\_profile"

}

case object DismissedId extends FilterReason {

override val reason: String = s"dismissed\_candidate"

}

case object OptedOutId extends FilterReason {

override val reason: String = s"candidate\_opted\_out\_from\_criteria\_in\_request"

}

// gizmoduck predicates

case object NoUser extends FilterReason {

override val reason: String = "no\_user\_result\_from\_gizmoduck"

}

case object AddressBookUndiscoverable extends FilterReason {

override val reason: String = "not\_discoverable\_via\_address\_book"

}

case object PhoneBookUndiscoverable extends FilterReason {

override val reason: String = "not\_discoverable\_via\_phone\_book"

}

case object Deactivated extends FilterReason {

override val reason: String = "deactivated"

}

case object Suspended extends FilterReason {

override val reason: String = "suspended"

}

case object Restricted extends FilterReason {

override val reason: String = "restricted"

}

case object NsfwUser extends FilterReason {

override val reason: String = "nsfwUser"

}

case object NsfwAdmin extends FilterReason {

override val reason: String = "nsfwAdmin"

}

case object HssSignal extends FilterReason {

override val reason: String = "hssSignal"

}

case object IsProtected extends FilterReason {

override val reason: String = "isProtected"

}

case class CountryTakedown(countryCode: String) extends FilterReason {

override val reason: String = s"takedown\_in\_$countryCode"

}

case object Blink extends FilterReason {

override val reason: String = "blink"

}

case object AlreadyFollowed extends FilterReason {

override val reason: String = "already\_followed"

}

case object InvalidRelationship extends FilterReason {

override val reason: String = "invalid\_relationship"

}

case object NotFollowingTargetUser extends FilterReason {

override val reason: String = "not\_following\_target\_user"

}

case object CandidateSideHoldback extends FilterReason {

override val reason: String = "candidate\_side\_holdback"

}

case object Inactive extends FilterReason {

override val reason: String = "inactive"

}

case object MissingRecommendabilityData extends FilterReason {

override val reason: String = "missing\_recommendability\_data"

}

case object HighTweetVelocity extends FilterReason {

override val reason: String = "high\_tweet\_velocity"

}

case object AlreadyRecommended extends FilterReason {

override val reason: String = "already\_recommended"

}

case object MinStateNotMet extends FilterReason {

override val reason: String = "min\_state\_user\_not\_met"

}

case object FailOpen extends FilterReason {

override val reason: String = "fail\_open"

}

}