package com.twitter.product\_mixer.component\_library.filter

import com.twitter.product\_mixer.component\_library.model.candidate.BaseTweetCandidate

import com.twitter.product\_mixer.component\_library.model.candidate.TweetAuthorIdFeature

import com.twitter.product\_mixer.core.functional\_component.filter.Filter

import com.twitter.product\_mixer.core.functional\_component.filter.FilterResult

import com.twitter.product\_mixer.core.model.common.CandidateWithFeatures

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

import com.twitter.product\_mixer.core.pipeline.PipelineQuery

import com.twitter.stitch.Stitch

/\*\*

\* A [[filter]] that filters based on whether query user is the author of the tweet. This will NOT filter empty user ids

\* @note It is recommended to apply [[HasAuthorIdFeatureFilter]] before this, as this will FAIL if feature is unavailable

\*

\* @tparam Candidate The type of the candidates

\*/

case class TweetAuthorIsSelfFilter[Candidate <: BaseTweetCandidate]()

extends Filter[PipelineQuery, Candidate] {

override val identifier: FilterIdentifier = FilterIdentifier("TweetAuthorIsSelf")

override def apply(

query: PipelineQuery,

candidates: Seq[CandidateWithFeatures[Candidate]]

): Stitch[FilterResult[Candidate]] = {

val (kept, removed) = candidates.partition { candidate =>

val authorId = candidate.features.get(TweetAuthorIdFeature)

!query.getOptionalUserId.contains(authorId)

}

val filterResult = FilterResult(

kept = kept.map(\_.candidate),

removed = removed.map(\_.candidate)

)

Stitch.value(filterResult)

}

}