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

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

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

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 checks for presence of a successfully hydrated [[TweetAuthorIdFeature]]

\*/

case class HasAuthorIdFeatureFilter[Candidate <: TweetCandidate]()

extends Filter[PipelineQuery, Candidate] {

override val identifier = FilterIdentifier("HasAuthorIdFeature")

override def apply(

query: PipelineQuery,

candidates: Seq[CandidateWithFeatures[Candidate]]

): Stitch[FilterResult[Candidate]] = {

val (kept, removed) = candidates.partition(\_.features.getTry(TweetAuthorIdFeature).isReturn)

Stitch.value(FilterResult(kept.map(\_.candidate), removed.map(\_.candidate)))

}

}