package com.twitter.product\_mixer.core.pipeline.candidate

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

import com.twitter.product\_mixer.core.model.common.presentation.CandidateWithDetails

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

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

import com.twitter.stitch.Arrow

/\*\*

\* A Candidate Pipeline

\*

\* This is an abstract class, as we only construct these via the [[CandidatePipelineBuilder]].

\*

\* A [[CandidatePipeline]] is capable of processing requests (queries) and returning candidates

\* in the form of a [[CandidatePipelineResult]]

\*

\* @tparam Query the domain model for the query or request

\*/

abstract class CandidatePipeline[-Query <: PipelineQuery] private[candidate]

extends Pipeline[CandidatePipeline.Inputs[Query], Seq[CandidateWithDetails]] {

override private[core] val config: BaseCandidatePipelineConfig[Query, \_, \_, \_]

override val arrow: Arrow[CandidatePipeline.Inputs[Query], CandidatePipelineResult]

override val identifier: CandidatePipelineIdentifier

}

object CandidatePipeline {

case class Inputs[+Query <: PipelineQuery](

query: Query,

existingCandidates: Seq[CandidateWithDetails])

}