package com.twitter.product\_mixer.component\_library.pipeline.candidate.who\_to\_follow\_module

import com.twitter.peoplediscovery.api.{thriftscala => t}

import com.twitter.product\_mixer.component\_library.candidate\_source.people\_discovery.PeopleDiscoveryCandidateSource

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

import com.twitter.product\_mixer.core.feature.Feature

import com.twitter.product\_mixer.core.functional\_component.candidate\_source.BaseCandidateSource

import com.twitter.product\_mixer.core.functional\_component.common.alert.Alert

import com.twitter.product\_mixer.core.functional\_component.decorator.CandidateDecorator

import com.twitter.product\_mixer.core.functional\_component.decorator.urt.builder.metadata.BaseFeedbackActionInfoBuilder

import com.twitter.product\_mixer.core.functional\_component.decorator.urt.builder.timeline\_module.BaseModuleDisplayTypeBuilder

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

import com.twitter.product\_mixer.core.functional\_component.gate.Gate

import com.twitter.product\_mixer.core.functional\_component.transformer.CandidateFeatureTransformer

import com.twitter.product\_mixer.core.functional\_component.transformer.CandidatePipelineQueryTransformer

import com.twitter.product\_mixer.core.functional\_component.transformer.CandidatePipelineResultsTransformer

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

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

import com.twitter.product\_mixer.core.pipeline.candidate.CandidatePipelineConfig

import com.twitter.timelines.configapi.FSParam

import com.twitter.timelines.configapi.Param

import com.twitter.timelines.configapi.decider.DeciderParam

object WhoToFollowCandidatePipelineConfig {

val MinCandidatesSize = 3

val MaxCandidatesSize = 20

val identifier: CandidatePipelineIdentifier = CandidatePipelineIdentifier("WhoToFollow")

}

class WhoToFollowCandidatePipelineConfig[Query <: PipelineQuery](

override val identifier: CandidatePipelineIdentifier,

override val enabledDeciderParam: Option[DeciderParam[Boolean]],

override val supportedClientParam: Option[FSParam[Boolean]],

override val alerts: Seq[Alert],

override val gates: Seq[Gate[Query]],

whoToFollowCandidateSource: PeopleDiscoveryCandidateSource,

override val filters: Seq[Filter[Query, UserCandidate]],

moduleDisplayTypeBuilder: BaseModuleDisplayTypeBuilder[Query, UserCandidate],

feedbackActionInfoBuilder: Option[

BaseFeedbackActionInfoBuilder[PipelineQuery, UserCandidate]

],

displayLocationParam: Param[String],

supportedLayoutsParam: Param[Seq[String]],

layoutVersionParam: Param[Int],

excludedUserIdsFeature: Option[Feature[PipelineQuery, Seq[Long]]],

) extends CandidatePipelineConfig[

Query,

t.GetModuleRequest,

t.RecommendedUser,

UserCandidate

] {

override val candidateSource: BaseCandidateSource[t.GetModuleRequest, t.RecommendedUser] =

whoToFollowCandidateSource

override val queryTransformer: CandidatePipelineQueryTransformer[

PipelineQuery,

t.GetModuleRequest

] = WhoToFollowCandidatePipelineQueryTransformer(

displayLocationParam,

supportedLayoutsParam,

layoutVersionParam,

excludedUserIdsFeature)

override val featuresFromCandidateSourceTransformers: Seq[

CandidateFeatureTransformer[t.RecommendedUser]

] = Seq(WhoToFollowResponseFeatureTransformer)

override val resultTransformer: CandidatePipelineResultsTransformer[

t.RecommendedUser,

UserCandidate

] = { user => UserCandidate(user.userId) }

override val decorator: Option[CandidateDecorator[Query, UserCandidate]] =

Some(WhoToFollowCandidateDecorator(moduleDisplayTypeBuilder, feedbackActionInfoBuilder))

}