package com.twitter.home\_mixer.product.scored\_tweets.candidate\_pipeline

import com.twitter.home\_mixer.product.scored\_tweets.feature\_hydrator.FrsSeedUsersQueryFeatureHydrator

import com.twitter.home\_mixer.product.scored\_tweets.gate.MinCachedTweetsGate

import com.twitter.home\_mixer.product.scored\_tweets.model.ScoredTweetsQuery

import com.twitter.home\_mixer.product.scored\_tweets.param.ScoredTweetsParam.CachedScoredTweets

import com.twitter.home\_mixer.product.scored\_tweets.param.ScoredTweetsParam.CandidatePipeline

import com.twitter.home\_mixer.product.scored\_tweets.query\_transformer.TimelineRankerFrsQueryTransformer

import com.twitter.home\_mixer.product.scored\_tweets.response\_transformer.ScoredTweetsFrsResponseFeatureTransformer

import com.twitter.product\_mixer.component\_library.candidate\_source.timeline\_ranker.TimelineRankerRecapCandidateSource

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

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

import com.twitter.product\_mixer.core.functional\_component.feature\_hydrator.BaseQueryFeatureHydrator

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.candidate.CandidatePipelineConfig

import com.twitter.timelineranker.{thriftscala => tlr}

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

import javax.inject.Inject

import javax.inject.Singleton

/\*\*

\* Candidate Pipeline Config that takes user recommendations from Follow Recommendation Service (FRS)

\* and makes a TimelineRanker->Earlybird query for tweet candidates from those users.

\* Additionally, the candidate pipeline hydrates followedByUserIds so that followed-by social proof

\* can be used.

\*/

@Singleton

class ScoredTweetsFrsCandidatePipelineConfig @Inject() (

timelineRankerRecapCandidateSource: TimelineRankerRecapCandidateSource,

frsSeedUsersQueryFeatureHydrator: FrsSeedUsersQueryFeatureHydrator)

extends CandidatePipelineConfig[

ScoredTweetsQuery,

tlr.RecapQuery,

tlr.CandidateTweet,

TweetCandidate

] {

override val identifier: CandidatePipelineIdentifier =

CandidatePipelineIdentifier("ScoredTweetsFrs")

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

Some(CandidatePipeline.EnableFrsParam)

override val gates: Seq[Gate[ScoredTweetsQuery]] = Seq(

MinCachedTweetsGate(identifier, CachedScoredTweets.MinCachedTweetsParam)

)

override val queryFeatureHydration: Seq[

BaseQueryFeatureHydrator[ScoredTweetsQuery, \_]

] = Seq(frsSeedUsersQueryFeatureHydrator)

override val candidateSource: BaseCandidateSource[tlr.RecapQuery, tlr.CandidateTweet] =

timelineRankerRecapCandidateSource

override val queryTransformer: CandidatePipelineQueryTransformer[

ScoredTweetsQuery,

tlr.RecapQuery

] = TimelineRankerFrsQueryTransformer(identifier)

override val featuresFromCandidateSourceTransformers: Seq[

CandidateFeatureTransformer[tlr.CandidateTweet]

] = Seq(ScoredTweetsFrsResponseFeatureTransformer)

override val resultTransformer: CandidatePipelineResultsTransformer[

tlr.CandidateTweet,

TweetCandidate

] = { candidate => TweetCandidate(candidate.tweet.get.id) }

}