package com.twitter.home\_mixer.product.for\_you

import com.twitter.home\_mixer.candidate\_pipeline.ConversationServiceResponseFeatureTransformer

import com.twitter.home\_mixer.functional\_component.decorator.HomeConversationServiceCandidateDecorator

import com.twitter.home\_mixer.functional\_component.decorator.urt.builder.HomeFeedbackActionInfoBuilder

import com.twitter.home\_mixer.functional\_component.feature\_hydrator.InNetworkFeatureHydrator

import com.twitter.home\_mixer.functional\_component.feature\_hydrator.NamesFeatureHydrator

import com.twitter.home\_mixer.functional\_component.feature\_hydrator.TweetypieFeatureHydrator

import com.twitter.home\_mixer.functional\_component.filter.InvalidConversationModuleFilter

import com.twitter.home\_mixer.functional\_component.filter.InvalidSubscriptionTweetFilter

import com.twitter.home\_mixer.functional\_component.filter.PreviouslyServedTweetsFilter

import com.twitter.home\_mixer.functional\_component.filter.RetweetDeduplicationFilter

import com.twitter.home\_mixer.model.HomeFeatures.IsHydratedFeature

import com.twitter.home\_mixer.model.HomeFeatures.QuotedTweetDroppedFeature

import com.twitter.home\_mixer.model.HomeFeatures.TimelineServiceTweetsFeature

import com.twitter.home\_mixer.product.for\_you.model.ForYouQuery

import com.twitter.home\_mixer.service.HomeMixerAlertConfig

import com.twitter.product\_mixer.component\_library.candidate\_source.tweetconvosvc.ConversationServiceCandidateSource

import com.twitter.product\_mixer.component\_library.candidate\_source.tweetconvosvc.ConversationServiceCandidateSourceRequest

import com.twitter.product\_mixer.component\_library.candidate\_source.tweetconvosvc.TweetWithConversationMetadata

import com.twitter.product\_mixer.component\_library.filter.FeatureFilter

import com.twitter.product\_mixer.component\_library.filter.PredicateFeatureFilter

import com.twitter.product\_mixer.component\_library.gate.NoCandidatesGate

import com.twitter.product\_mixer.component\_library.gate.NonEmptySeqFeatureGate

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.common.SpecificPipelines

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

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

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

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

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

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

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

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

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

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

import javax.inject.Inject

import javax.inject.Singleton

/\*\*

\* Candidate Pipeline Config that fetches Tweet ancestors from Conversation Service Candidate Source

\*/

@Singleton

class ForYouConversationServiceCandidatePipelineConfig @Inject() (

forYouScoredTweetsCandidatePipelineConfig: ForYouScoredTweetsCandidatePipelineConfig,

forYouTimelineScorerCandidatePipelineConfig: ForYouTimelineScorerCandidatePipelineConfig,

conversationServiceCandidateSource: ConversationServiceCandidateSource,

tweetypieFeatureHydrator: TweetypieFeatureHydrator,

namesFeatureHydrator: NamesFeatureHydrator,

invalidSubscriptionTweetFilter: InvalidSubscriptionTweetFilter,

homeFeedbackActionInfoBuilder: HomeFeedbackActionInfoBuilder)

extends DependentCandidatePipelineConfig[

ForYouQuery,

ConversationServiceCandidateSourceRequest,

TweetWithConversationMetadata,

TweetCandidate

] {

override val identifier: CandidatePipelineIdentifier =

CandidatePipelineIdentifier("ForYouConversationService")

override val gates: Seq[BaseGate[ForYouQuery]] = Seq(

NoCandidatesGate(

SpecificPipelines(

forYouTimelineScorerCandidatePipelineConfig.identifier,

forYouScoredTweetsCandidatePipelineConfig.identifier

)

),

NonEmptySeqFeatureGate(TimelineServiceTweetsFeature)

)

override val candidateSource: BaseCandidateSource[

ConversationServiceCandidateSourceRequest,

TweetWithConversationMetadata

] = conversationServiceCandidateSource

override val queryTransformer: DependentCandidatePipelineQueryTransformer[

ForYouQuery,

ConversationServiceCandidateSourceRequest

] = { (query, candidates) =>

val timelineServiceTweets = query.features

.map(\_.getOrElse(TimelineServiceTweetsFeature, Seq.empty)).getOrElse(Seq.empty)

val tweetsWithConversationMetadata = timelineServiceTweets.map { id =>

TweetWithConversationMetadata(

tweetId = id,

userId = None,

sourceTweetId = None,

sourceUserId = None,

inReplyToTweetId = None,

conversationId = None,

ancestors = Seq.empty

)

}

ConversationServiceCandidateSourceRequest(tweetsWithConversationMetadata)

}

override val featuresFromCandidateSourceTransformers: Seq[

CandidateFeatureTransformer[TweetWithConversationMetadata]

] = Seq(ConversationServiceResponseFeatureTransformer)

override val resultTransformer: CandidatePipelineResultsTransformer[

TweetWithConversationMetadata,

TweetCandidate

] = { sourceResult => TweetCandidate(id = sourceResult.tweetId) }

override val preFilterFeatureHydrationPhase1: Seq[

BaseCandidateFeatureHydrator[ForYouQuery, TweetCandidate, \_]

] = Seq(

InNetworkFeatureHydrator,

tweetypieFeatureHydrator

)

override def filters: Seq[Filter[ForYouQuery, TweetCandidate]] = Seq(

PreviouslyServedTweetsFilter,

RetweetDeduplicationFilter,

FeatureFilter.fromFeature(FilterIdentifier("TweetypieHydrated"), IsHydratedFeature),

PredicateFeatureFilter.fromPredicate(

FilterIdentifier("QuotedTweetDropped"),

shouldKeepCandidate = { features => !features.getOrElse(QuotedTweetDroppedFeature, false) }

),

invalidSubscriptionTweetFilter,

InvalidConversationModuleFilter

)

override val postFilterFeatureHydration: Seq[

BaseCandidateFeatureHydrator[ForYouQuery, TweetCandidate, \_]

] = Seq(namesFeatureHydrator)

override val decorator: Option[CandidateDecorator[ForYouQuery, TweetCandidate]] =

HomeConversationServiceCandidateDecorator(homeFeedbackActionInfoBuilder)

override val alerts = Seq(

HomeMixerAlertConfig.BusinessHours.defaultSuccessRateAlert(),

HomeMixerAlertConfig.BusinessHours.defaultEmptyResponseRateAlert()

)

}