package com.twitter.home\_mixer.functional\_component.decorator.urt.builder

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

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

import com.twitter.home\_mixer.product.following.model.HomeMixerExternalStrings

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

import com.twitter.product\_mixer.core.feature.featuremap.FeatureMap

import com.twitter.product\_mixer.core.functional\_component.decorator.urt.builder.social\_context.BaseSocialContextBuilder

import com.twitter.product\_mixer.core.model.marshalling.response.urt.metadata.LikeGeneralContextType

import com.twitter.product\_mixer.core.model.marshalling.response.urt.metadata.SocialContext

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

import com.twitter.product\_mixer.core.product.guice.scope.ProductScoped

import com.twitter.stringcenter.client.StringCenter

import javax.inject.Inject

import javax.inject.Provider

import javax.inject.Singleton

@Singleton

case class LikedBySocialContextBuilder @Inject() (

externalStrings: HomeMixerExternalStrings,

@ProductScoped stringCenterProvider: Provider[StringCenter])

extends BaseSocialContextBuilder[PipelineQuery, TweetCandidate] {

private val stringCenter = stringCenterProvider.get()

private val engagerSocialContextBuilder = EngagerSocialContextBuilder(

contextType = LikeGeneralContextType,

stringCenter = stringCenter,

oneUserString = externalStrings.socialContextOneUserLikedString,

twoUsersString = externalStrings.socialContextTwoUsersLikedString,

moreUsersString = externalStrings.socialContextMoreUsersLikedString,

timelineTitle = externalStrings.socialContextLikedByTimelineTitle

)

def apply(

query: PipelineQuery,

candidate: TweetCandidate,

candidateFeatures: FeatureMap

): Option[SocialContext] = {

// Liked by users are valid only if they pass both the SGS and Perspective filters.

val validLikedByUserIds =

candidateFeatures

.getOrElse(SGSValidLikedByUserIdsFeature, Nil)

.filter(

candidateFeatures.getOrElse(PerspectiveFilteredLikedByUserIdsFeature, Nil).toSet.contains)

engagerSocialContextBuilder(

socialContextIds = validLikedByUserIds,

query = query,

candidateFeatures = candidateFeatures

)

}

}