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

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

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

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

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

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

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

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

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

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

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

import com.twitter.stringcenter.client.StringCenter

import javax.inject.Inject

import javax.inject.Singleton

@Singleton

case class UnfollowUserChildFeedbackActionBuilder @Inject() (

@ProductScoped stringCenter: StringCenter,

externalStrings: HomeMixerExternalStrings) {

def apply(candidateFeatures: FeatureMap): Option[ChildFeedbackAction] = {

val isInNetwork = candidateFeatures.getOrElse(InNetworkFeature, false)

val userIdOpt = candidateFeatures.getOrElse(AuthorIdFeature, None)

if (isInNetwork) {

userIdOpt.flatMap { userId =>

val screenNamesMap =

candidateFeatures.getOrElse(ScreenNamesFeature, Map.empty[Long, String])

val userScreenNameOpt = screenNamesMap.get(userId)

userScreenNameOpt.map { userScreenName =>

val prompt = stringCenter.prepare(

externalStrings.unfollowUserString,

Map("username" -> userScreenName)

)

val confirmation = stringCenter.prepare(

externalStrings.unfollowUserConfirmationString,

Map("username" -> userScreenName)

)

ChildFeedbackAction(

feedbackType = RichBehavior,

prompt = Some(prompt),

confirmation = Some(confirmation),

feedbackUrl = None,

hasUndoAction = Some(true),

confirmationDisplayType = None,

clientEventInfo = None,

icon = Some(icon.Unfollow),

richBehavior = Some(RichFeedbackBehaviorToggleFollowUser(userId)),

subprompt = None

)

}

}

} else None

}

}