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

import com.twitter.finagle.tracing.Trace

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

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

import com.twitter.product\_mixer.core.model.common.UniversalNoun

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

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

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

import com.twitter.suggests.controller\_data.home\_tweets.v1.{thriftscala => v1ht}

import com.twitter.suggests.controller\_data.home\_tweets.{thriftscala => ht}

import com.twitter.suggests.controller\_data.thriftscala.ControllerData

import com.twitter.suggests.controller\_data.v2.thriftscala.{ControllerData => ControllerDataV2}

case class HomeAdsClientEventDetailsBuilder(injectionType: Option[String])

extends BaseClientEventDetailsBuilder[PipelineQuery, UniversalNoun[Any]] {

override def apply(

query: PipelineQuery,

candidate: UniversalNoun[Any],

candidateFeatures: FeatureMap

): Option[ClientEventDetails] = {

val homeTweetsControllerDataV1 = v1ht.HomeTweetsControllerData(

tweetTypesBitmap = 0L,

traceId = Some(Trace.id.traceId.toLong),

requestJoinId = None)

val serializedControllerData = HomeClientEventDetailsBuilder.ControllerDataSerializer(

ControllerData.V2(

ControllerDataV2.HomeTweets(ht.HomeTweetsControllerData.V1(homeTweetsControllerDataV1))))

val clientEventDetails = ClientEventDetails(

conversationDetails = None,

timelinesDetails = Some(

TimelinesDetails(

injectionType = injectionType,

controllerData = Some(serializedControllerData),

sourceData = None)),

articleDetails = None,

liveEventDetails = None,

commerceDetails = None

)

Some(clientEventDetails)

}

}