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

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

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

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.functional\_component.decorator.urt.builder.metadata.BaseClientEventInfoBuilder

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

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

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

import com.twitter.timelines.injection.scribe.InjectionScribeUtil

/\*\*

\* Sets the [[ClientEventInfo]] with the `component` field set to the Suggest Type assigned to each candidate

\*/

case class HomeClientEventInfoBuilder[Query <: PipelineQuery, Candidate <: UniversalNoun[Any]](

detailsBuilder: Option[BaseClientEventDetailsBuilder[Query, Candidate]] = None)

extends BaseClientEventInfoBuilder[Query, Candidate] {

override def apply(

query: Query,

candidate: Candidate,

candidateFeatures: FeatureMap,

element: Option[String]

): Option[ClientEventInfo] = {

val suggestType = candidateFeatures

.getOrElse(SuggestTypeFeature, None)

.getOrElse(throw new UnsupportedOperationException(s"No SuggestType was set"))

Some(

ClientEventInfo(

component = InjectionScribeUtil.scribeComponent(suggestType),

element = element,

details = detailsBuilder.flatMap(\_.apply(query, candidate, candidateFeatures)),

action = None,

/\*\*

\* A backend entity encoded by the Client Entities Encoding Library.

\* Placeholder string for now

\*/

entityToken = candidateFeatures.getOrElse(EntityTokenFeature, None)

)

)

}

}