package com.twitter.home\_mixer.functional\_component.feature\_hydrator

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

import com.twitter.product\_mixer.core.feature.Feature

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

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

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

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

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

import com.twitter.stitch.Stitch

import com.twitter.timelinemixer.clients.feedback.FeedbackHistoryManhattanClient

import javax.inject.Inject

import javax.inject.Singleton

@Singleton

case class FeedbackHistoryQueryFeatureHydrator @Inject() (

feedbackHistoryClient: FeedbackHistoryManhattanClient)

extends QueryFeatureHydrator[PipelineQuery] {

override val identifier: FeatureHydratorIdentifier = FeatureHydratorIdentifier("FeedbackHistory")

override val features: Set[Feature[\_, \_]] = Set(FeedbackHistoryFeature)

override def hydrate(

query: PipelineQuery

): Stitch[FeatureMap] =

Stitch

.callFuture(feedbackHistoryClient.get(query.getRequiredUserId))

.map { feedbackHistory =>

FeatureMapBuilder().add(FeedbackHistoryFeature, feedbackHistory).build()

}

}