package com.twitter.visibility.builder.media

import com.twitter.finagle.stats.StatsReceiver

import com.twitter.mediaservices.media\_util.GenericMediaKey

import com.twitter.visibility.builder.FeatureMapBuilder

import com.twitter.visibility.common.MediaMetadataSource

import com.twitter.visibility.features.HasDmcaMediaFeature

import com.twitter.visibility.features.MediaGeoRestrictionsAllowList

import com.twitter.visibility.features.MediaGeoRestrictionsDenyList

import com.twitter.visibility.features.AuthorId

class MediaMetadataFeatures(

mediaMetadataSource: MediaMetadataSource,

statsReceiver: StatsReceiver) {

private[this] val scopedStatsReceiver = statsReceiver.scope("media\_metadata\_features")

private[this] val requests = scopedStatsReceiver.counter("requests")

private[this] val hasDmcaMedia =

scopedStatsReceiver.scope(HasDmcaMediaFeature.name).counter("requests")

private[this] val mediaGeoAllowList =

scopedStatsReceiver.scope(MediaGeoRestrictionsAllowList.name).counter("requests")

private[this] val mediaGeoDenyList =

scopedStatsReceiver.scope(MediaGeoRestrictionsDenyList.name).counter("requests")

private[this] val uploaderId =

scopedStatsReceiver.scope(AuthorId.name).counter("requests")

def forGenericMediaKey(

genericMediaKey: GenericMediaKey

): FeatureMapBuilder => FeatureMapBuilder = { featureMapBuilder =>

requests.incr()

featureMapBuilder.withFeature(

HasDmcaMediaFeature,

mediaIsDmca(genericMediaKey)

)

featureMapBuilder.withFeature(

MediaGeoRestrictionsAllowList,

geoRestrictionsAllowList(genericMediaKey)

)

featureMapBuilder.withFeature(

MediaGeoRestrictionsDenyList,

geoRestrictionsDenyList(genericMediaKey)

)

featureMapBuilder.withFeature(

AuthorId,

mediaUploaderId(genericMediaKey)

)

}

private def mediaIsDmca(genericMediaKey: GenericMediaKey) = {

hasDmcaMedia.incr()

mediaMetadataSource.getMediaIsDmca(genericMediaKey)

}

private def geoRestrictionsAllowList(genericMediaKey: GenericMediaKey) = {

mediaGeoAllowList.incr()

mediaMetadataSource.getGeoRestrictionsAllowList(genericMediaKey).map { allowListOpt =>

allowListOpt.getOrElse(Nil)

}

}

private def geoRestrictionsDenyList(genericMediaKey: GenericMediaKey) = {

mediaGeoDenyList.incr()

mediaMetadataSource.getGeoRestrictionsDenyList(genericMediaKey).map { denyListOpt =>

denyListOpt.getOrElse(Nil)

}

}

private def mediaUploaderId(genericMediaKey: GenericMediaKey) = {

uploaderId.incr()

mediaMetadataSource.getMediaUploaderId(genericMediaKey).map { uploaderIdOpt =>

uploaderIdOpt.map(Set(\_)).getOrElse(Set.empty)

}

}

}