package com.twitter.visibility.models

import com.twitter.context.TwitterContext

import com.twitter.context.thriftscala.Viewer

import com.twitter.featureswitches.{UserAgent => FSUserAgent}

import com.twitter.finatra.request.util.AddressUtils

case class ViewerContext(

userId: Option[Long] = None,

guestId: Option[Long] = None,

userAgentStr: Option[String] = None,

clientApplicationId: Option[Long] = None,

auditIp: String = "0.0.0.0",

requestCountryCode: Option[String] = None,

requestLanguageCode: Option[String] = None,

deviceId: Option[String] = None,

ipTags: Set[String] = Set.empty,

isVerifiedCrawler: Boolean = false,

userRoles: Option[Set[String]] = None) {

val fsUserAgent: Option[FSUserAgent] = userAgentStr.flatMap(ua => FSUserAgent(userAgent = ua))

val isTwOffice: Boolean = ipTags.contains(AddressUtils.TwofficeIpTag)

}

object ViewerContext {

def fromContext: ViewerContext = viewerContext.getOrElse(ViewerContext())

def fromContextWithViewerIdFallback(viewerId: Option[Long]): ViewerContext =

viewerContext

.map { viewer =>

if (viewer.userId.isEmpty) {

viewer.copy(userId = viewerId)

} else {

viewer

}

}.getOrElse(ViewerContext(viewerId))

private def viewerContext: Option[ViewerContext] =

TwitterContext(TwitterContextPermit)().map(apply)

def apply(viewer: Viewer): ViewerContext = new ViewerContext(

userId = viewer.userId,

guestId = viewer.guestId,

userAgentStr = viewer.userAgent,

clientApplicationId = viewer.clientApplicationId,

auditIp = viewer.auditIp.getOrElse("0.0.0.0"),

requestCountryCode = viewer.requestCountryCode collect { case value => value.toLowerCase },

requestLanguageCode = viewer.requestLanguageCode collect { case value => value.toLowerCase },

deviceId = viewer.deviceId,

ipTags = viewer.ipTags.toSet,

isVerifiedCrawler = viewer.isVerifiedCrawler.getOrElse(false)

)

}