package com.twitter.visibility.builder.tweets

import com.twitter.finagle.stats.StatsReceiver

import com.twitter.stitch.Stitch

import com.twitter.tweetypie.thriftscala.EscherbirdEntityAnnotations

import com.twitter.tweetypie.thriftscala.Tweet

import com.twitter.visibility.builder.FeatureMapBuilder

import com.twitter.visibility.common.MisinformationPolicySource

import com.twitter.visibility.features.\_

import com.twitter.visibility.models.MisinformationPolicy

import com.twitter.visibility.models.SemanticCoreMisinformation

import com.twitter.visibility.models.ViewerContext

class MisinformationPolicyFeatures(

misinformationPolicySource: MisinformationPolicySource,

statsReceiver: StatsReceiver) {

private[this] val scopedStatsReceiver =

statsReceiver.scope("misinformation\_policy\_features")

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

private[this] val tweetMisinformationPolicies =

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

def forTweet(

tweet: Tweet,

viewerContext: ViewerContext

): FeatureMapBuilder => FeatureMapBuilder = {

requests.incr()

tweetMisinformationPolicies.incr()

\_.withFeature(

TweetMisinformationPolicies,

misinformationPolicy(tweet.escherbirdEntityAnnotations, viewerContext))

.withFeature(

TweetEnglishMisinformationPolicies,

misinformationPolicyEnglishOnly(tweet.escherbirdEntityAnnotations))

}

def misinformationPolicyEnglishOnly(

escherbirdEntityAnnotations: Option[EscherbirdEntityAnnotations],

): Stitch[Seq[MisinformationPolicy]] = {

val locale = Some(

MisinformationPolicySource.LanguageAndCountry(

language = Some("en"),

country = Some("us")

))

fetchMisinformationPolicy(escherbirdEntityAnnotations, locale)

}

def misinformationPolicy(

escherbirdEntityAnnotations: Option[EscherbirdEntityAnnotations],

viewerContext: ViewerContext

): Stitch[Seq[MisinformationPolicy]] = {

val locale = viewerContext.requestLanguageCode.map { language =>

MisinformationPolicySource.LanguageAndCountry(

language = Some(language),

country = viewerContext.requestCountryCode

)

}

fetchMisinformationPolicy(escherbirdEntityAnnotations, locale)

}

def fetchMisinformationPolicy(

escherbirdEntityAnnotations: Option[EscherbirdEntityAnnotations],

locale: Option[MisinformationPolicySource.LanguageAndCountry]

): Stitch[Seq[MisinformationPolicy]] = {

Stitch.collect(

escherbirdEntityAnnotations

.map(\_.entityAnnotations)

.getOrElse(Seq.empty)

.filter(\_.domainId == SemanticCoreMisinformation.domainId)

.map(annotation =>

misinformationPolicySource

.fetch(

annotation,

locale

)

.map(misinformation =>

MisinformationPolicy(

annotation = annotation,

misinformation = misinformation

)))

)

}

}